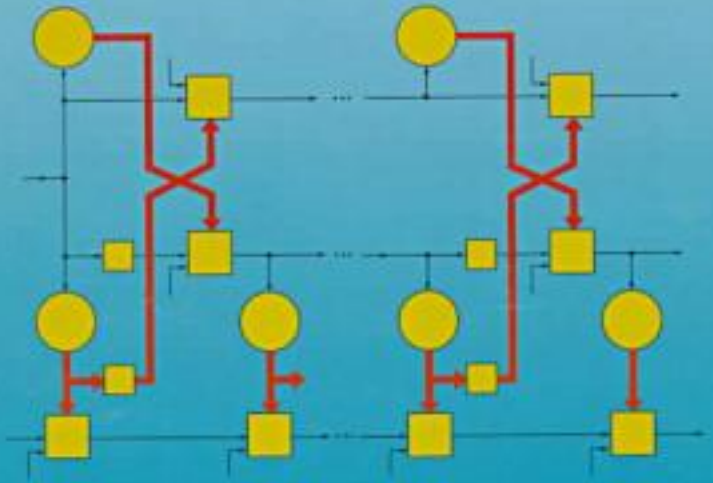
Adaptive Filter Theory

THIRD EDITION



Simon Haykin

PRENTICE HALL INFORMATION AND SYSTEM SCIENCES SERIES
Thomas Kailath, Series Editor

Contents

Introduction

1. THE FILTERING PROBLEM

The term *filter* is often used to describe a device in the form of a piece of physical hardware or software that is applied to a set of noisy data in order to extract information about a prescribed quantity of interest. The noise may arise from a variety of sources. For example, the data may have been derived by means of noisy sensors or may represent a useful signal component that has been corrupted by transmission through a communication channel. In any event, we may use a filter to perform three basic information-processing tasks:

1. *Filtering*, which means the extraction of information about a quantity of interest at time t by using data measured up to and including time t.
2. *Smoothing*, which differs from filtering in that information about the quantity of interest need not be available at time t, and data measured later than time t can be used in obtaining this information. This means that in the case of smoothing there is a *delay* in producing the result of interest. Since in the smoothing process we are able to use data obtained not only up to time t but also data obtained after time t, we would expect smoothing to be more accurate in some sense than filtering.
3. *Prediction*, which is the forecasting side of information processing. The aim here is to derive information about what the quantity of interest will be like at some time $t + \tau$ in the future, for some $\tau > 0$, by using data measured up to and including time t.

We may classify filters into linear and nonlinear. A filter is said to be *linear* if the filtered, smoothed, or predicted quantity at the output of the device is a *linear function of the observations applied to the filter input.* Otherwise, the filter is *nonlinear.*

In the statistical approach to the solution of the *linear filtering problem* as classified above, we assume the availability of certain statistical parameters (i.e., *mean and correlation functions*) of the useful signal and unwanted additive noise, and the requirement is to design a linear filter with the noisy data as input so as to minimize the effects of noise at the filter output according to some statistical criterion. A useful approach to this filter-optimization problem is to minimize the mean-square value of the *error signal* that is defined as the difference between some desired response and the actual filter output. For stationary inputs, the resulting solution is *commonly* known as the *Wiener filter*, which is said to be *optimum in the mean-square sense*. A plot of the mean-square value of the error signal versus the adjustable parameters of a linear filter is referred to as the *error-performance surface*. The minimum point of this surface represents the Wiener solution.

The Wiener filter is inadequate for dealing with situations in which *nonstationarity* of the signal and/or noise is intrinsic to the problem. In such situations, the optimum filter has to assume a *time-varying form*. A highly successful solution to this more difficult problem is found in the *Kalman filter*, a powerful device with a wide variety of engineering applications.

Linear filter theory, encompassing both Wiener and Kalman filters, has been developed fully in the literature for *continuous-time* as well as *discrete-time* signals. However, for technical reasons influenced by the wide availability of digital computers and the ever-increasing use of digital signal-processing devices, we find in practice that the discrete-time representation is often the preferred method. Accordingly, in subsequent chapters, we only consider the discrete-time version of Wiener and Kalman filters. In this method of representation, the input and output signals, as well as the characteristics of the filters themselves, are all defined at discrete instants of time. In any case, a continuous-time signal may always be represented by a *sequence of samples* that are derived by observing the signal at uniformly spaced instants of time. No loss of information is incurred during this conversion process provided, of course, we satisfy the well-known *sampling theorem*, according to which the sampling rate has to be greater than twice the highest frequency component of the continuous-time signal. We may thus represent a continuous-time signal $u(t)$ by the sequence $u(n)$, $n = 0, \pm 1, = \pm 2, \ldots$, where for convenience we have normalized the sampling period to unity, a practice that we follow throughout the book.

2. ADAPTIVE FILTERS

The design of a Wiener filter requires *a priori* information about the statistics of the data to be processed. The filter is optimum only when the statistical characteristics of the input data match the *a priori* information on which the design of the filter is based. When this information is not known completely, however, it may not be possible to design the Wiener filter or else the design may no longer be optimum. A straightforward approach that we may use in such situations is the "estimate and plug" procedure. This is a two-stage process whereby the filter first "estimates" the statistical parameters of the relevant signals and then "plugs" the results so obtained into a *nonrecursive* formula for computing

the filter parameters. For *real-time* operation, this procedure has the disadvantage of requiring excessively elaborate and costly hardware. A more efficient method is to use an *adaptive filter*. By such a device we mean one that is *self-designing* in that the adaptive filter relies for its operation on a *recursive algorithm*, which makes it possible for the filter to perform satisfactorily in an environment where complete knowledge of the relevant signal characteristics is not available. The algorithm starts from some predetermined set of *initial conditions*, representing whatever we know about the environment. Yet, in a stationary environment, we find that after successive iterations of the algorithm it *converges* to the optimum Wiener solution in some statistical sense. In a nonstationary environment, the algorithm offers a *tracking* capability, in that it can track time variations in the statistics of the input data, provided that the variations are sufficiently slow.

As a direct consequence of the application of a recursive algorithm whereby the parameters of an adaptive filter are updated from one iteration to the next, the parameters become *data dependent*. This, therefore, means that an adaptive filter is in reality a *nonlinear device, in the sense that it does not obey the principle of superposition.* Notwithstanding this property, adaptive filters are commonly classified as linear or nonlinear. An adaptive filter is said to be *linear* if the estimate of a quantity of interest is computed adaptively (at the output of the filter) as a *linear combination of the available set of observations applied to the filter input.* Otherwise, the adaptive filter is said to be *nonlinear.*

A wide variety of recursive algorithms have been developed in the literature for the operation of linear adaptive filters. In the final analysis, the choice of one algorithm over another is determined by one or more of the following factors:

- *Rate of convergence.* This is defined as the number of iterations required for the algorithm, in response to stationary inputs, to converge "close enough" to the optimum Wiener solution in the mean-square sense. A fast rate of convergence allows the algorithm to adapt rapidly to a stationary environment of unknown statistics.
- *Misadjustment.* For an algorithm of interest, this parameter provides a quantitative measure of the amount by which the final value of the mean-squared error, averaged over an ensemble of adaptive filters, deviates from the minimum mean-squared error that is produced by the Wiener filter.
- *Tracking.* When an adaptive filtering algorithm operates in a nonstationary environment, the algorithm is required to *track* statistical variations in the environment. The tracking performance of the algorithm, however, is influenced by two contradictory features: (1) rate of convergence, and (b) steady-state fluctuation due to algorithm noise.
- *Robustness.* For an adaptive filter to be *robust*, small disturbances (i.e., disturbances with small energy) can only result in small estimation errors. The disturbances may arise from a variety of factors, internal or external to the filter.
- *Computational requirements.* Here the issues of concern include (a) the number of operations (i.e., multiplications, divisions, and additions/subtractions) required to make one complete iteration of the algorithm, (b) the size of memory locations

required to store the data and the program, and (c) the investment required to program the algorithm on a computer.

- *Structure.* This refers to the structure of information flow in the algorithm, determining the manner in which it is implemented in hardware form. For example, an algorithm whose structure exhibits high modularity, parallelism, or concurrency is well suited for implementation using very large-scale integration (VLSI).[1]
- *Numerical properties.* When an algorithm is implemented numerically, inaccuracies are produced due to *quantization errors*. The quantization errors are due to analog-to-digital conversion of the input data and digital representation of internal calculations. Ordinarily, it is the latter source of quantization errors that poses a serious design problem. In particular, there are two basic issues of concern: numerical stability and numerical accuracy. *Numerical stability* is an inherent characteristic of an adaptive filtering algorithm. *Numerical accuracy*, on the other hand, is determined by the number of *bits* (i.e., *bi*nary digi*ts*) used in the numerical representation of data samples and filter coefficients. An adaptive filtering algorithm is said to be numerically robust when it is insensitive to variations in the wordlength used in its digital implementation.

These factors, in their own ways, also enter into the design of nonlinear adaptive filters, except for the fact that we now no longer have a well-defined frame of reference in the form of a Wiener filter. Rather, we speak of a nonlinear filtering algorithm that may converge to a local minimum or, hopefully, a global minimum on the error-performance surface.

In the sections that follow, we shall first discuss various aspects of linear adaptive filters. Discussion of nonlinear adaptive filters is deferred to a later section in the chapter.

3. LINEAR FILTER STRUCTURES

The operation of a linear adaptive filtering algorithm involves two basic processes: (1) a *filtering* process designed to produce an output in response to a sequence of input data, and (2) an *adaptive* process, the purpose of which is to provide a mechanism for the *adaptive control* of an *adjustable* set of parameters used in the filtering process. These two processes work interactively with each other. Naturally, the choice of a structure for the filtering process has a profound effect on the operation of the algorithm as a whole.

[1] VLSI technology favors the implementation of algorithms that possess high modularity, parallelism, or concurrency. We say that a structure is *modular* when it consists of similar stages connected in cascade. By *parallelism* we mean a large number of operations being performed side by side. By *concurrency* we mean a large number of *similar* computations being performed at the same time.

For a discussion of VLSI implementation of adaptive filters, see Shanbhag and Parhi (1994). This book emphasizes the use of *pipelining*, an architectural technique used for increasing the throughput of an adaptive filtering algorithm.

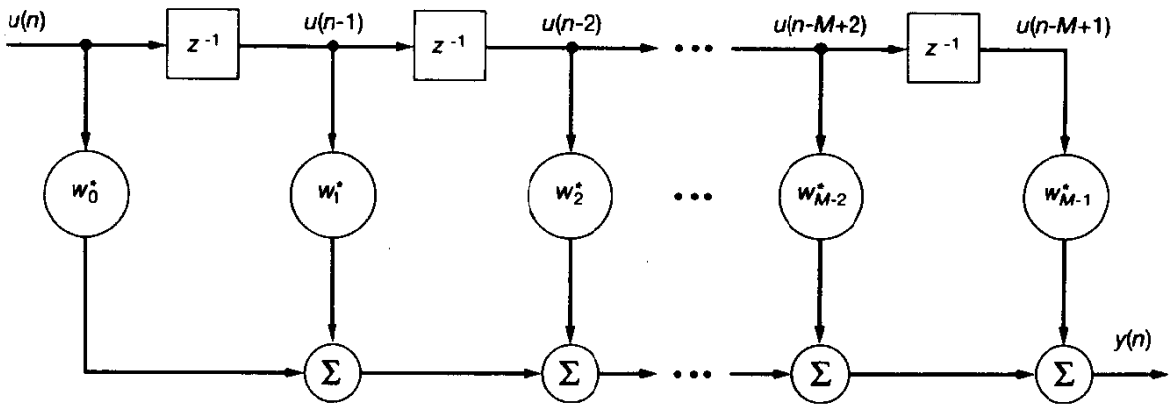


Figure 1 Transversal filter.

There are three types of filter structures that distinguish themselves in the context of an adaptive filter with *finite memory* or, equivalently, *finite-duration impulse response*. The three filter structures are as follows:

1. *Transversal filter.* The *transversal filter*,[2] also referred to as a *tapped-delay line filter*, consists of three basic elements, as depicted in Fig. 1: (a) *unit-delay element*, (b) *multiplier*, and (c) *adder*. The number of delay elements used in the filter determines the finite duration of its impulse response. The number of delay elements, shown as $M - 1$ in Fig. 1, is commonly referred to as the *filter order*. In this figure, the delay elements are each identified by the *unit-delay operator* z^{-1}. In particular, when z^{-1} operates on the input $u(n)$, the resulting output is $u(n - 1)$. The role of each multiplier in the filter is to multiply the *tap input* (to which it is connected) by a filter coefficient referred to as a *tap weight*. Thus a multiplier connected to the kth tap input $u(n - k)$ produces the scalar version of the *inner product*, $w_k^* u(n - k)$, where w_k is the respective tap weight and $k = 0, 1, \ldots, M - 1$. The asterisk denotes *complex conjugation*, which assumes that the tap inputs and therefore the tap weights are all *complex valued*. The combined role of the adders in the filter is to sum the individual multiplier outputs and produce an overall filter output. For the transversal filter described in Fig. 1, the filter output is given by

$$y(n) = \sum_{k=0}^{m-1} w_k^* u(n-k) \tag{1}$$

[2]The transversal filter was first described by Kallmann as a continuous-time device whose output is formed as a linear combination of voltages taken from uniformly spaced taps in a nondispersive delay line (Kallmann, 1940). In recent years, the transversal filter has been implemented using digital circuitry, charge-coupled devices, or surface-acoustic wave devices. Owing to its versatility and ease of implementation, the transversal filter has emerged as an essential signal-processing structure in a wide variety of applications.

Equation (1) is called a finite *convolution sum* in the sense that it *convolves* the finite-duration impulse response of the filter, w_n^*, with the filter input $u(n)$ to produce the filter output $y(n)$.

2. *Lattice predictor.* A *lattice predictor*[3] is *modular* in structure in that it consists of a number of individual stages, each of which has the appearance of a lattice, hence the name "lattice" as a structural descriptor. Figure 2 depicts a lattice predictor consisting of $M - 1$ stages; the number $M - 1$ is referred to as the *predictor order*. The mth stage of the lattice predictor in Fig. 2 is described by the pair of input–output relations (assuming the use of complex-valued, wide-sense stationary input data):

$$f_m(n) = f_{m-1}(n) + \kappa_m^* b_{m-1}(n-1) \tag{2}$$

$$b_m(n) = b_{m-1}(n-1) + \kappa_m f_{m-1}(n) \tag{3}$$

where $m = 1, 2, \ldots, M - 1$, and $M - 1$ is the *final* predictor order. The variable $f_m(n)$ is the mth *forward prediction error*, and $b_m(n)$ is the mth *backward prediction error*. The coefficient κ_m is called the mth *reflection coefficient*. The forward prediction error $f_m(n)$ is defined as the difference between the input $u(n)$ and its *one-step predicted* value; the latter is based on the set of m *past inputs* $u(n-1), \ldots, u(n-m)$. Correspondingly, the backward prediction error $b_m(n)$ is defined as the difference between the input $u(n-m)$ and its "backward" prediction based on the set of m "future" inputs $u(n), \ldots, u(n-m+1)$. Considering the conditions at the input of stage 1 in Fig. 2, we have

$$f_0(n) = b_0(n) = u(n) \tag{4}$$

where $u(n)$ is the lattice predictor input at time n. Thus, starting with the *initial conditions* of Eq. (4) and given the set of reflection coefficients $\kappa_1, \kappa_2, \ldots, \kappa_{M-1}$, we may determine the final pair of outputs $f_{M-1}(n)$ and $b_{M-1}(n)$ by moving through the lattice predictor, stage by stage.

For a *correlated* input sequence $u(n), u(n-1), \ldots, u(n-M+1)$ drawn from a stationary process, the backward prediction errors $b_0, b_1(n), \ldots, b_{M-1}(n)$ form a sequence of *uncorrelated* random variables. Moreover, there is a one-to-one correspondence between these two sequences of random variables in the sense that if we are given one of them, we may uniquely determine the other, and vice versa. Accordingly, a linear combination of the backward prediction errors $b_0(n), b_1(n), \ldots, b_{M-1}(n)$ may be used to provide an *estimate* of some desired response $d(n)$, as depicted in the lower half of Fig. 2. The arithmetic difference between $d(n)$ and the estimate so produced represents the estimation error $e(n)$. The process described herein is referred to as a *joint-process estimation*. Naturally, we may use the original input sequence $u(n), u(n-1), \ldots, u(n-M+1)$ to produce an estimate of the desired response $d(n)$ directly. The indirect method depicted in Fig. 2, however, has the advantage of simplifying the computation of the tap weights $h_0, h_1, \ldots, h_{M-1}$

[3]The development of the lattice predictor is credited to Itakura and Saito (1972).

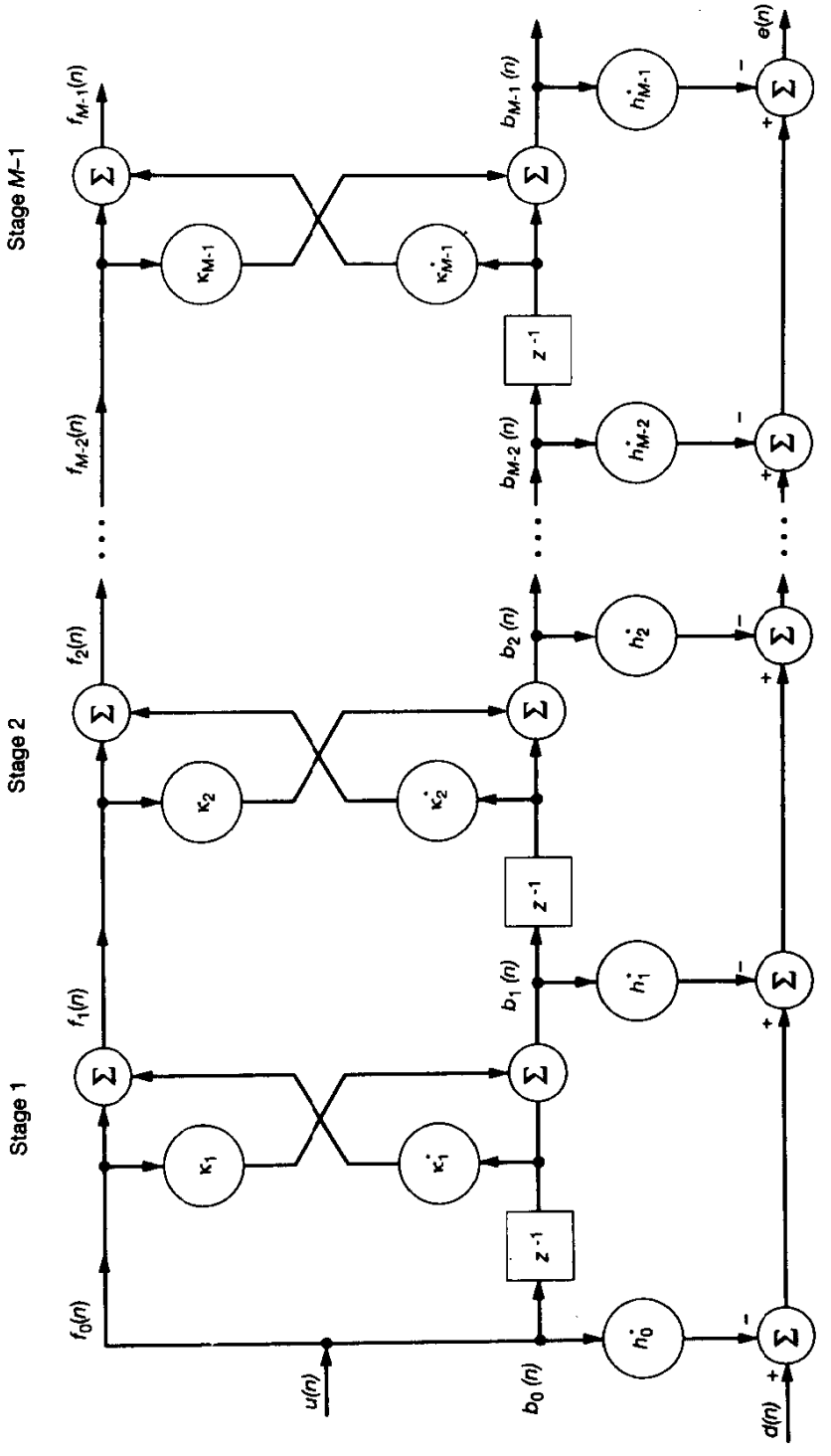


Figure 2 Multistage lattice filter.

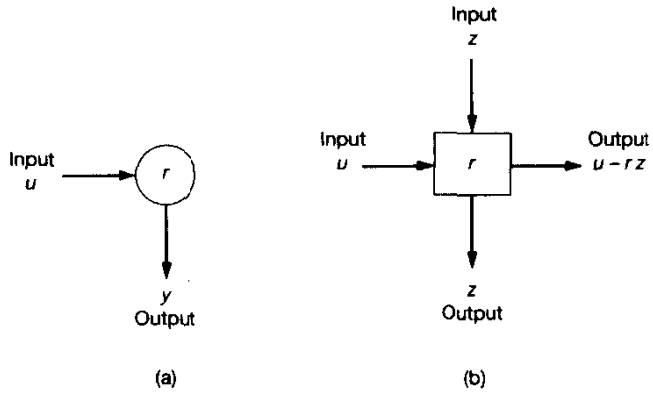


Figure 3 Two basic cells of a systolic array: (a) boundary cell; (b) internal cell.

by exploiting the uncorrelated nature of the corresponding backward prediction errors used in the estimation.

3. *Systolic array.* A *systolic array*[4] represents a *parallel computing* network ideally suited for *mapping* a number of important linear algebra computations, such as *matrix multiplication, triangularization,* and *back substitution.* Two basic types of processing elements may be distinguished in a systolic array: *boundary cells* and *internal cells.* Their functions are depicted in Figs. 3(a) and 3(b), respectively. In each case, the parameter r represents a value *stored* within the cell. The function of the boundary cell is to produce an output equal to the input u divided by the number r stored in the cell. The function of the internal cell is twofold: (a) to multiply the input z (coming in from the top) by the number r stored in the cell, subtract the product rz from the second input (coming in from the left), and thereby produce the difference $u - rz$ as an output from the right-hand side of the cell, and (b) to transmit the first input z downward without alteration.

Consider, for example, the 3-by-3 triangular array shown in Fig. 4. This systolic array involves a combination of boundary and internal cells. In this case, the triangular array computes an output vector $\mathbf{y}$ related to the input vector $\mathbf{u}$ as follows:

$$\mathbf{y} = \mathbf{R}^{-T}\mathbf{u} \tag{5}$$

where the $\mathbf{R}^{-\mathrm{T}}$ is the *inverse* of the transposed matrix $\mathbf{R}^T$. The elements of $\mathbf{R}^T$ are the respective cell contents of the triangular array. The zeros added to the inputs of the array in Fig. 4 are intended to provide the delays necessary for pipelining the computation described in Eq. (5).

A systolic array architecture, as described herein, offers the desirable features of *modularity, local interconnections,* and highly *pipelined* and *synchronized* parallel processing; the synchronization is achieved by means of a global *clock.*

We note that the transversal filter of Fig. 1, the joint-process estimator of Fig. 2 based on a lattice predictor, and the triangular systolic array of Fig. 4 have a common

[4]The systolic array was pioneered by Kung and Leiserson (1978). In particular, the use of systolic arrays has made it possible to achieve a high throughput, which is required for many advanced signal processing algorithms to operate in *real time.*

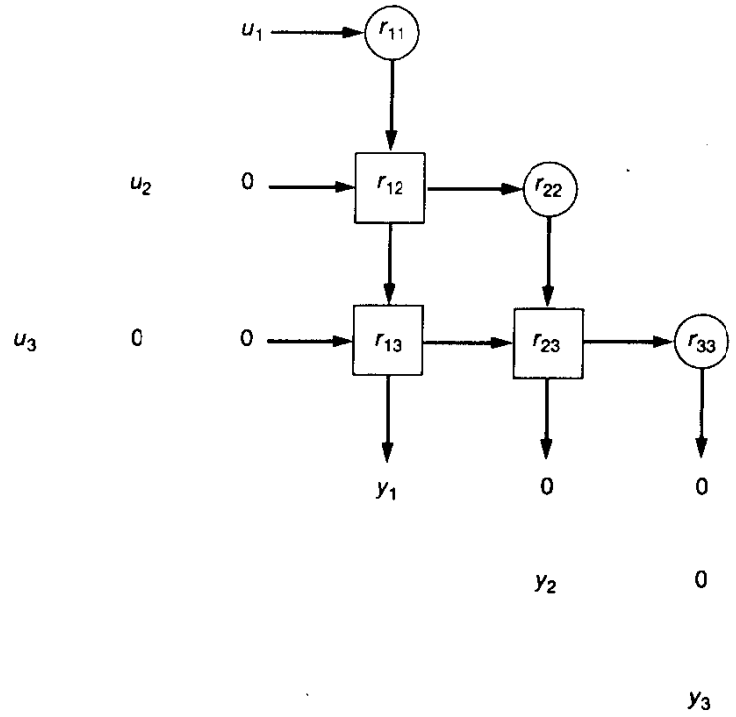


Figure 4 Triangular systolic array.

property: all three of them are characterized by an impulse response of finite duration. In other words, they are examples of a *finite-duration impulse response (FIR) filter*, whose structures contain *feedforward* paths only. On the other hand, the filter structure shown in Fig. 5 is an example of an *infinite-duration impulse response (IIR) filter*. The feature that distinguishes an IIR filter from an FIR filter is the inclusion of *feedback* paths. Indeed, it is the presence of feedback that makes the duration of the impulse response of an IIR filter infinitely long. Furthermore, the presence of feedback introduces a new problem, namely, that of *stability*. In particular, it is possible for an IIR filter to become unstable (i.e., break into oscillation), unless special precaution is taken in the choice of feedback coefficients. By contrast, an FIR filter is inherently *stable*. This explains the reason for the popular use of FIR filters, in one form or another, as the structural basis for the design of linear adaptive filters.

4. APPROACHES TO THE DEVELOPMENT OF LINEAR ADAPTIVE FILTERING ALGORITHMS

There is no unique solution to the linear adaptive filtering problem. Rather, we have a "kit of tools" represented by a variety of recursive algorithms, each of which offers desirable features of its own. The challenge facing the user of adaptive filtering is, first, to understand the capabilities and limitations of various adaptive filtering algorithms and, second,

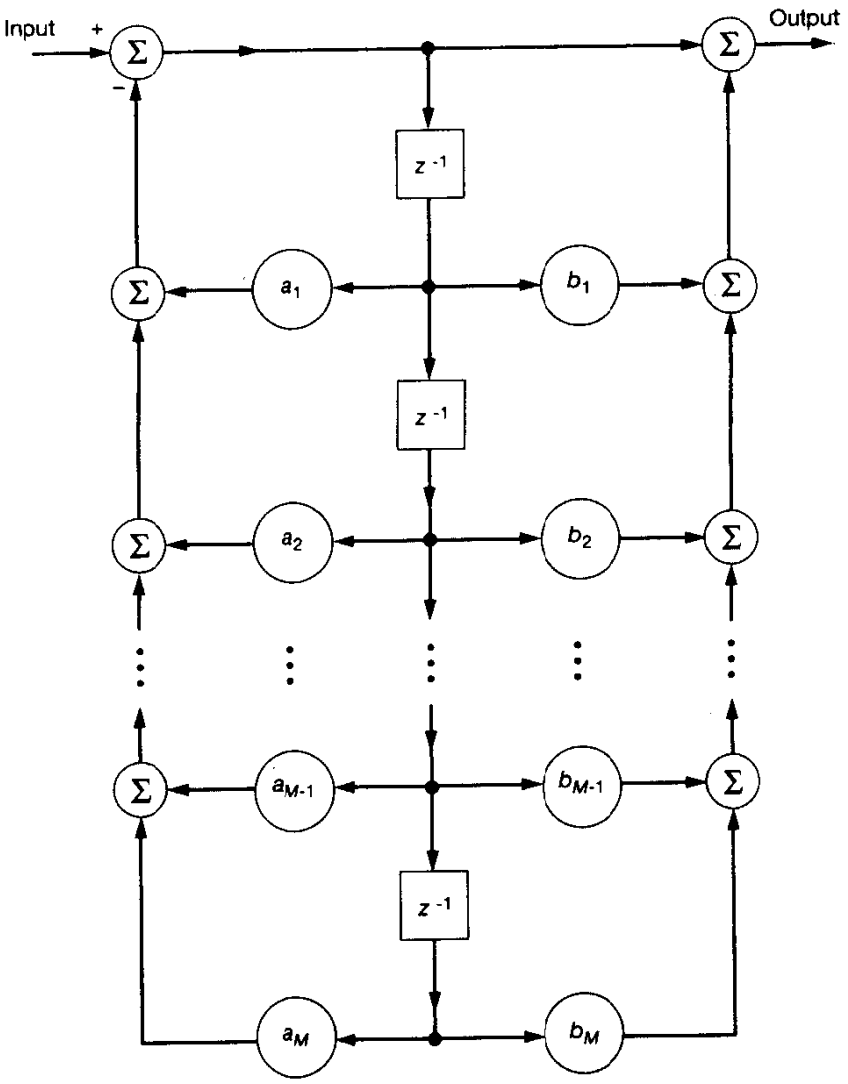


Figure 5 IIR filter.

to use this understanding in the selection of the appropriate algorithm for the application at hand.

Basically, we may identify two distinct approaches for deriving recursive algorithms for the operation of linear adaptive filters, as discussed next.

Stochastic Gradient Approach

Here we may use a tapped-delay line or transversal filter as the structural basis for implementing the linear adaptive filter. For the case of stationary inputs, the *cost function*,[5] also referred to as the *index of performance*, is defined as the *mean-squared error* (i.e., the mean-square value of the difference between the desired response and the transversal filter output). This cost function is precisely a second-order function of the tap weights in the transversal filter. The dependence of the mean-squared error on the unknown tap weights may be viewed to be in the form of a *multidimensional paraboloid* (i.e., punch bowl) with a uniquely defined bottom or *minimum point*. As mentioned previously, we refer to this paraboloid as the *error-performance surface*; the tap weights corresponding to the minimum point of the surface define the optimum Wiener solution.

To develop a recursive algorithm for updating the tap weights of the adaptive transversal filter, we proceed in two stages. We first modify the system of *Wiener—Hopf equations* (i.e., the matrix equation defining the optimum Wiener solution) through the use of the *method of steepest descent*, a well-known technique in optimization theory. This modification requires the use of a *gradient vector*, the value of which depends on two parameters: the *correlation matrix* of the tap inputs in the transversal filter, and the *cross-correlation vector* between the desired response and the same tap inputs. Next, we use instantaneous values for these correlations so as to derive an *estimate* for the gradient vector, making it assume a *stochastic* character in general. The resulting algorithm is widely known as the *least-mean-square (LMS) algorithm*, the essence of which may be described in words as follows for the case of a transversal filter operating on real-valued data:

$$\begin{pmatrix}\text{updated value}\\ \text{of tap-weight}\\ \text{vector}\end{pmatrix} = \begin{pmatrix}\text{old value}\\ \text{of tap-weight}\\ \text{vector}\end{pmatrix} + \begin{pmatrix}\text{learning-}\\ \text{rate}\\ \text{parameter}\end{pmatrix}\begin{pmatrix}\text{tap-}\\ \text{input}\\ \text{vector}\end{pmatrix}\begin{pmatrix}\text{error}\\ \text{signal}\end{pmatrix}$$

where the error signal is defined as the difference between some desired response and the actual response of the transversal filter produced by the tap-input vector.

The LMS algorithm is simple and yet capable of achieving satisfactory performance under the right conditions. Its major limitations are a relatively slow rate of convergence and a sensitivity to variations in the condition number of the correlation matrix of the tap inputs; the *condition number* of a Hermitian matrix is defined as the ratio of its largest

[5] In the general definition of a function, we speak of a transformation from a vector space into the space of real (or complex) scalars (Luenberger, 1969; Dorny, 1975). A cost function provides a quantitative measure for assessing the quality of performance; hence the restriction of it to a real scalar.

eigenvalue to its smallest eigenvalue. Nevertheless, the LMS algorithm is highly popular and widely used in a variety of applications.

In a nonstationary environment, the orientation of the error-performance surface varies continuously with time. In this case, the LMS algorithm has the added task of continually *tracking* the bottom of the error-performance surface. Indeed, tracking will occur provided that the input data vary slowly compared to the *learning rate* of the LMS algorithm.

The stochastic gradient approach may also be pursued in the context of a lattice structure. The resulting adaptive filtering algorithm is called the *gradient adaptive lattice (GAL) algorithm*. In their own individual ways, the LMS and GAL algorithms are just two members of the *stochastic gradient family* of linear adaptive filters, although it must be said that the LMS algorithm is by far the most popular member of this family.

Least-squares Estimation

The second approach to the development of linear adaptive filtering algorithms is based on the *method of least squares*. According to this method we minimize a cost function or index of performance that is defined as the *sum of weighted error squares*, where the *error* or *residual* is itself defined as the difference between some desired response and the actual filter output. The method of least squares may be formulated with *block estimation* or *recursive estimation* in mind. In block estimation the input data stream is arranged in the form of blocks of equal length (duration), and the filtering of input data proceeds on a block-by-block basis. In recursive estimation, on the other hand, the estimates of interest (e.g., tap weights of a transversal filter) are *updated* on a sample-by-sample basis. Ordinarily, a recursive estimator requires less storage than a block estimator, which is the reason for its much wider use in practice.

Recursive least-squares (RLS) estimation may be viewed as a special case of Kalman filtering. A distinguishing feature of the Kalman filter is the notion of *state*, which provides a measure of all the inputs applied to the filter up to a specific instant of time. Thus, at the heart of the Kalman filtering algorithm we have a recursion that may be described in words as follows:

$$\begin{pmatrix}\text{updated value}\\ \text{of the}\\ \text{state}\end{pmatrix} = \begin{pmatrix}\text{old value}\\ \text{of the}\\ \text{state}\end{pmatrix} + \begin{pmatrix}\text{Kalman}\\ \text{gain}\end{pmatrix}\begin{pmatrix}\text{innovation}\\ \text{vector}\end{pmatrix}$$

where the *innovation vector* represents new information put into the filtering process at the time of the computation. For the present, it suffices to say that there is indeed a one-to-one correspondence between the Kalman variables and RLS variables. This correspondence means that we can tap the vast literature on Kalman filters for the design of linear adaptive filters based on recursive least-squares estimation. Moreover, we may classify the *recursive least-squares family* of linear adaptive filtering algorithms into three distinct categories, depending on the approach taken:

1. *Standard RLS algorithm*, which assumes the use of a transversal filter as the structural basis of the linear adaptive filter. Derivation of the standard RLS algorithm relies on a basic result in linear algebra known as the *matrix inversion lemma*. Most importantly, it enjoys the same virtues and suffers from the same limitations as the standard Kalman filtering algorithm. The limitations include lack of numerical robustness and excessive computational complexity. Indeed, it is these two limitations that have prompted the development of the other two categories of RLS algorithms, described next.
2. *Square-root RLS algorithms*, which are based on *QR-decomposition* of the incoming data matrix. Two well-known techniques for performing this decomposition are the *Householder transformation* and the *Givens rotation*, both of which are data-adaptive transformations. At this point in the discussion, we need to merely say that RLS algorithms based on the Householder transformation or Givens rotation are numerically stable and robust. The resulting linear adaptive filters are referred to as *square-root adaptive filters*, because in a matrix sense they represent the square-foot forms of the standard RLS algorithm.
3. *Fast RLS algorithms*. The standard RLS algorithm and square-root RLS algorithms have a computational complexity that increases as the square of M, where M is the number of adjustable weights (i.e., the number of degrees of freedom) in the algorithm. Such algorithms are often referred to as $O(M^2)$ *algorithms*, where $O(\cdot)$ denotes "order of." By contrast, the LMS algorithm is an $O(M)$ algorithm, in that its computational complexity increases linearly with M. When M is large, the computational complexity of $O(M^2)$ algorithms may become objectionable from a hardware implementation point of view. There is therefore a strong motivation to modify the formulation of the RLS algorithm in such a way that the computational complexity assumes an $O(M)$ form. This objective is indeed achievable, in the case of temporal processing, first by virtue of the inherent *redundancy* in the *Toeplitz structure* of the input data matrix and, second, by exploiting this redundancy through the use of *linear least-squares prediction in both the forward and backward directions*. The resulting algorithms are known collectively as *fast RLS algorithms*; they combine the desirable characteristics of recursive linear least-squares estimation with an $O(M)$ computational complexity. Two types of fast RLS algorithms may be identified, depending on the filtering structure employed:

   - *Order-recursive adaptive filters*, which are based on a latticelike structure for making linear forward and backward predictions.
   - *Fast transversal filters*, in which the linear forward and backward predictions are performed using separate transversal filters.

   Certain (but not all) realizations of order-recursive adaptive filters are known to be numerically stable, whereas fast transversal filters suffer from a numerical sta-

bility problem and therefore require some form of stabilization for them to be of practical use.

An introductory discussion of linear adaptive filters would be incomplete without saying something about their tracking behavior. In this context, we note that stochastic gradient algorithms such as the LMS algorithm are *model-independent*; generally speaking, we would expect them to exhibit good tracking behavior, which indeed they do. In contrast, RLS algorithms are *model-dependent*; this, in turn, means that their tracking behavior may be inferior to that of a member of the stochastic gradient family, unless care is taken to minimize the mismatch between the mathematical model on which they are based and the underlying physical process responsible for generating the input data.

How to Choose an Adaptive Filter

Given the wide variety of adaptive filters available to a system designer, how can a choice be made for an application of interest? Clearly, whatever the choice, it has to be *cost-effective*. With this goal in mind, we may identify three important issues that require attention: *computational cost, performance*, and *robustness*. The use of computer simulation provides a good first step in undertaking a detailed investigation of these issues. We may begin by using the LMS algorithm as an adaptive filtering tool for the study. The LMS algorithm is relatively simple to implement. Yet it is powerful enough to evaluate the practical benefits that may result from the application of adaptivity to the problem at hand. Moreover, it provides a practical frame of reference for assessing any further improvement that may be attained through the use of more sophisticated adaptive filtering algorithms. Finally, the study must include tests with real-life data, for which there is no substitute.

Practical applications of adaptive filtering are very diverse, with each application having peculiarities of its own. The solution for one application may not be suitable for another. Nevertheless, to be successful we have to develop a physical understanding of the environment in which the filter has to operate and thereby relate to the realities of the application of interest.

5. REAL AND COMPLEX FORMS OF ADAPTIVE FILTERS

In the development of adaptive filtering algorithms, regardless of their origin, it is customary to assume that the input data are in baseband form. The term "baseband" is used to designate the band of frequencies representing the original (message) signal as generated by the source of information.

In such applications as communications, radar, and sonar, the information-bearing signal component of the receiver input typically consists of a message signal *modulated* onto a carrier wave. The bandwidth of the message signal is usually small compared to the carrier frequency, which means that the modulated signal is a *narrow-band signal*. To obtain the baseband representation of a narrow-band signal, the signal is translated down

in frequency in such a way that the effect of the carrier wave is completely removed, yet the information content of the message signal is fully preserved. In general, the baseband signal so obtained is *complex*. In other words, a sample $u(n)$ of the signal may be written as

$$u(n) = u_I(n) + ju_Q(n) \tag{6}$$

where $u_I(n)$ is the *in-phase* (real) *component*, and $u_Q(n)$ is the *quadrature* (imaginary) *component*. Equivalently, we may express $u(n)$ as

$$u(n) = |u(n)|e^{j\phi(n)} \tag{7}$$

where $|u(n)|$ is the *magnitude* and $\phi(n)$ is the *phase angle*.

Accordingly, the theory of adaptive filters (both linear and nonlinear) developed in subsequent chapters of the book assumes the use of complex signals. An adaptive filtering algorithm so developed is said to be in *complex form*. The important virtue of complex adaptive filters is that they preserve the mathematical formulation and elegant structure of complex signals encountered in the aforementioned areas of application.

If the signals to be processed are *real*, we naturally use the *real form* of the adaptive-filtering algorithm of interest. Given the complex form of an adaptive filtering algorithm, it is straightforward to deduce the corresponding real form of the algorithm. Specifically, we do two things:

1. The operation of *complex conjugation*, wherever in the algorithm, is simply removed.
2. The operation of *Hermitian transposition* (i.e., conjugate transposition) of a matrix, wherever in the algorithm, is replaced by ordinary transposition.

Simply put, complex adaptive filters include real adaptive filters as special cases.

6. NONLINEAR ADAPTIVE FILTERS

The theory of linear optimum filters is based on the mean-square error criterion. The Wiener filter that results from the minimization of such a criterion, and which represents the goal of linear adaptive filtering for a stationary environment, can only relate to second-order statistics of the input data and no higher. This constraint limits the ability of a linear adaptive filter to extract information from input data that are non-Gaussian. Despite its theoretical importance, the existence of Gaussian noise is open to question (Johnson and Rao, 1990). Moreover, non-Gaussian processes are quite common in many signal processing applications encountered in practice. The use of a Wiener filter or a linear adaptive filter to extract signals of interest in the presence of such non-Gaussian processes will therefore yield suboptimal solutions. We may overcome this limitation by incorporating some form of *nonlinearity* in the structure of the adaptive filter to take care of higher-order statistics. Although by so doing, we no longer have the Wiener filter as a frame of refer-

ence and so complicate the mathematical analysis, we would expect to benefit in two significant ways: improving learning efficiency and a broadening of application areas.

Fundamentally, there are two types of nonlinear adaptive filters, as described next.

Volterra-based Nonlinear Adaptive Filters

In this type of a nonlinear adaptive filter, the nonlinearity is localized at the front end of the filter. It relies on the use of a *Volterra* series[6] that provides an attractive method for describing the input–output relationship of a nonlinear device with memory. This special form of a series derives its name from the fact that it was first studied by Vito Volterra around 1880 as a generalization of the Taylor series of a function. But Norbert Wiener (1958) was the first to use the Volterra series to model the input–output relationship of a nonlinear system.

Let the time series x_n denote the input of a nonlinear discrete-time system. We may then combine these input samples to define a set of *discrete Volterra kernels* as follows:

$$H_0 = \text{zero-order (dc) term}$$

$$H_1[x_n] = \text{first-order (linear) term}$$

$$= \sum_i h_i x_i$$

$$H_2[x_n] = \text{second-order (quadratic) term}$$

$$= \sum_i \sum_j h_{ij} x_i x_j$$

$$H_3[x_n] = \text{third-order (cubic) term}$$

$$= \sum_i \sum_j \sum_k h_{ijk} x_i x_j x_k$$

and so on for higher-order terms. Ordinarily, the nonlinear model coefficients, the h's, are fixed by analytical methods. We may thus decompose a nonlinear adaptive filter as follows:[7]

- A *nonlinear Volterra state expander* that combines the set of input values x_0, $x_1, \ldots, x_n$ to produce a larger set of outputs $u_0, u_1, \ldots, u_q$ for which q is larger than n. For example, the extension vector for a (3,2) system has the form

$$\mathbf{u} = [1, x_0, x_1, x_2, x_0^2, x_0x_1, x_0x_2, x_1x_0, x_1^2, x_1x_2, x_2x_0, x_2x_1, x_2^2]^T$$

- A *linear FIR adaptive filter* that operates on the u_k (i.e., elements of $\mathbf{u}$) as inputs to produce an estimate $\hat{d}_n$ of some desired response d_n.

[6]For a discussion of Volterra series, see the book by Schetzen (1981).

[7]The idea described herein is discussed in Rayner and Lynch (1989) and Lynch and Rayner (1989).

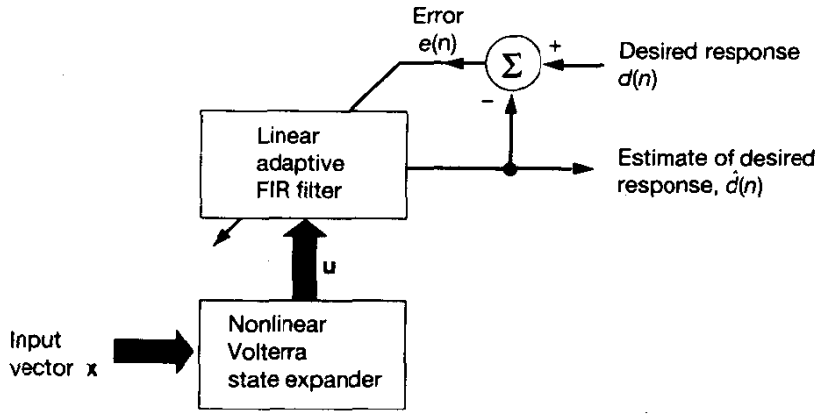


Figure 6 Volterra-based nonlinear adaptive filter.

The important thing to note here is that by using a scheme similar to that described in Fig. 6, we may expand the use of linear adaptive filters to include Volterra filters.

Neural Networks

An *artificial neural network*, or a *neural network* as it is commonly called, consists of the interconnection of a large number of nonlinear processing units called *neurons*; that is, the nonlinearity is distributed throughout the network. The development of neural networks, right from their inception, has been motivated by the way the human brain performs its operations; hence their name.

In this book, we are interested in a particular class of neural networks that *learn* about their environment in a *supervised manner*. In other words, as with the conventional form of a linear adaptive filter, we have a desired response that provides a target signal, which the neural network tries to approximate during the learning process. The approximation is achieved by adjusting a set of free parameters, called *synaptic weights*, in a systematic manner. In effect, the synaptic weights provide a mechanism for storing the information content of the input data.

In the context of adaptive signal processing applications, neural networks offer the following advantages:

- *Nonlinearity*, which makes it possible to account for the nonlinear behavior of physical phenomena responsible for generating the input data
- The ability to *approximate any prescribed input–output mapping* of a continuous nature
- *Weak statistical assumptions* about the environment, in which the network is embedded
- *Learning* capability, which is accomplished by undertaking a training session with input–output examples that are representative of the environment

- *Generalization*, which refers to the ability of the neural network to provide a satisfactory performance in response to *test data* never seen by the network before
- *Fault tolerance*, which means that the network continues to provide an acceptable performance despite the failure of some neurons in the network
- *VLSI implementability*, which exploits the massive parallelism built into the design of a neural network.

This is indeed an impressive list of attributes, which accounts for the widespread interest in the use of neural networks to solve signal-processing tasks that are too difficult for conventional (linear) adaptive filters.

7. APPLICATIONS

The ability of an adaptive filter to operate satisfactorily in an unknown environment and track time variations of input statistics make the adaptive filter a powerful device for signal-processing and control applications. Indeed, adaptive filters have been successfully applied in such diverse fields as communications, radar, sonar, seismology, and biomedical engineering. Although these applications are indeed quite different in nature, nevertheless, they have one basic common feature: an input vector and a desired response are used to compute an estimation error, which is in turn used to control the values of a set of adjustable filter coefficients. The adjustable coefficients may take the form of tap weights, reflection coefficients, rotation parameters, or synaptic weights, depending on the filter structure employed. However, the essential difference between the various applications of adaptive filtering arises in the manner in which the desired response is extracted. In this context, we may distinguish four basic classes of adaptive filtering applications, as depicted in Fig. 7. For convenience of presentation, the following notations are used in this figure:

u = input applied to the adaptive filter

y = output of the adaptive filter

d = desired response

$e = d - y$ = estimation error.

The functions of the four basic classes of adaptive filtering applications depicted herein are as follows:

I. *Identification* [Fig. 7(a)]. The notion of a *mathematical model* is fundamental to sciences and engineering. In the class of applications dealing with identification, an adaptive filter is used to provide a linear model that represents the best fit (in some sense) to an *unknown plant*. The plant and the adaptive filter are driven by the same input. The plant output supplies the desired response for the adaptive filter. If the plant is dynamic in nature, the model will be time varying.

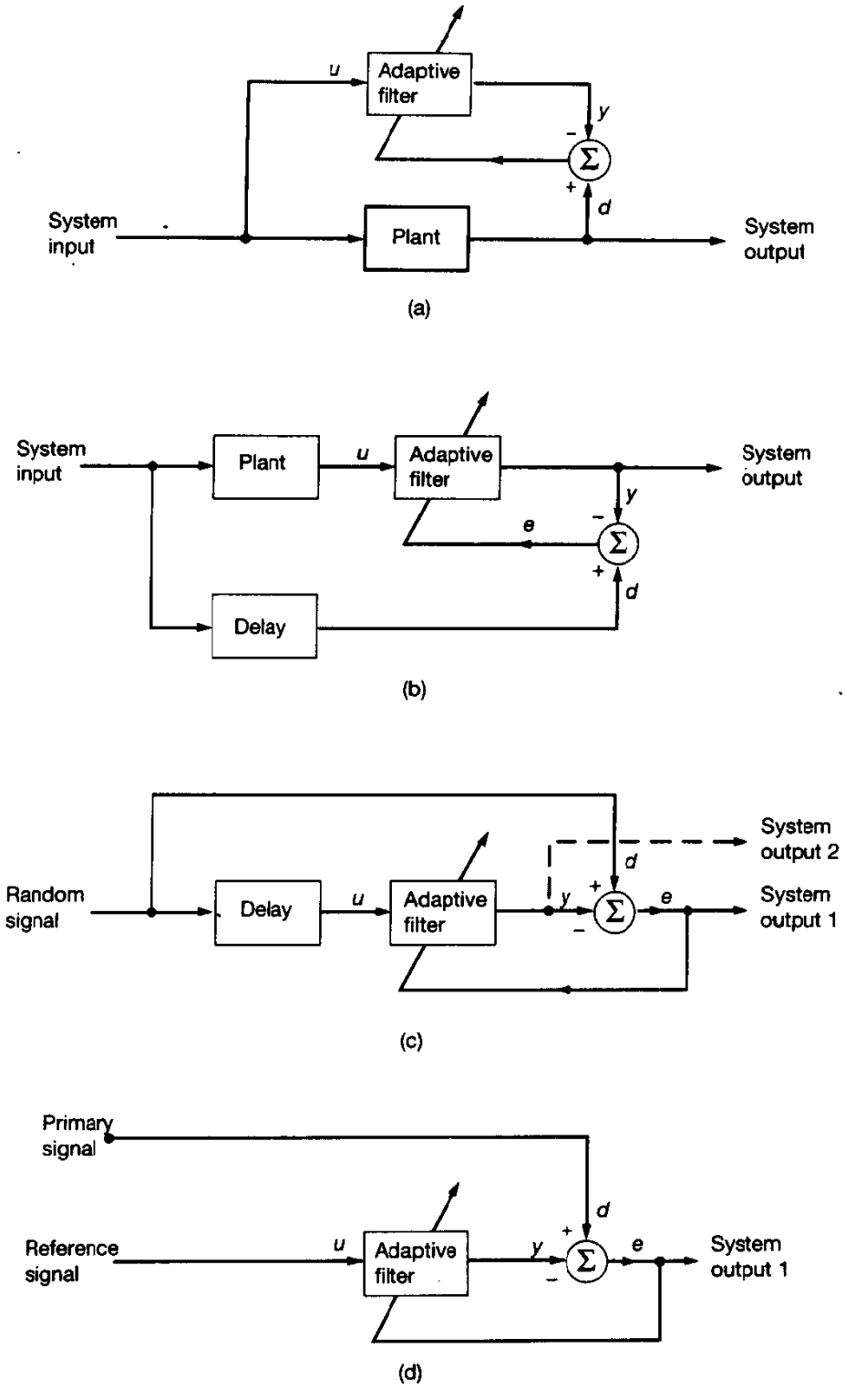


Figure 7 Four basic classes of adaptive filtering applications: (a) class I: identification; (b) class II: inverse modeling; (c) class III: prediction; (d) class IV: interference canceling.

II. *Inverse modeling* [Fig. 7(b)]. In this second class of applications, the function of the adaptive filter is to provide an *inverse model* that represents the best fit (in some sense) to an *unknown noisy plant*. Ideally, in the case of a linear system, the inverse model has a transfer function equal to the *reciprocal* (*inverse*) of the plant's transfer function, such that the combination of the two constitutes an ideal transmission medium. A delayed version of the plant (system) input constitutes the desired response for the adaptive filter. In some applications, the plant input is used without delay as the desired response.

III. *Prediction* [Fig. 7(c)]. Here the function of the adaptive filter is to provide the best *prediction* (in some sense) of the present value of a random signal. The present value of the signal thus serves the purpose of a desired response for the adaptive filter. Past values of the signal supply the input applied to the adaptive filter. Depending on the application of interest, the adaptive filter output or the estimation (prediction) error may serve as the system output. In the first case, the system operates as a *predictor*; in the latter case, it operates as a *prediction-error filter*.

IV. *Interference canceling* [Fig. 7(d)]. In this final class of applications, the adaptive filter is used to cancel *unknown interference* contained (alongside an information-bearing signal component) in a *primary signal*, with the cancelation being optimized in some sense. The primary signal serves as the desired response for the adaptive filter. A *reference (auxiliary) signal* is employed as the input to the adaptive filter. The reference signal is derived from a sensor or set of sensors located in relation to the sensor(s) supplying the primary signal in such a way that the information-bearing signal component is weak or essentially undetectable.

In Table 1 we have listed some applications that are illustrative of the four basic classes of adaptive filtering applications. These applications, totaling twelve, are drawn from the fields of control systems, seismology, electrocardiography, communications, and radar. They are described individually in the remainder of this section.

System Identification

System identification is the experimental approach to the modeling of a process or a plant (Goodwin and Payne, 1977; Ljung and Söderström, 1983; Ljung, 1987; Söderström and Stoica, 1988; Åström and Wittenmark, 1990). It involves the following steps: experimental planning, the selection of a model structure, parameter estimation, and model validation. The procedure of system identification, as pursued in practice, is iterative in nature in that we may have to go back and forth between these steps until a satisfactory model is built. Here we discuss briefly the idea of adaptive filtering algorithms for estimating the parameters of an unknown plant modeled as a transversal filter.

Suppose we have an unknown dynamic plant that is linear and time varying. The plant is characterized by a *real-valued* set of discrete-time measurements that describe the

TABLE 1 APPLICATIONS OF ADAPTIVE FILTERS

Class of adaptive filtering	Application
I. Identification	System identification Layered earth modeling
II. Inverse modeling	Predictive deconvolution Adaptive equalization Blind equalization
III. Prediction	Linear predictive coding Adaptive differential pulse-code modulation Autoregressive spectrum analysis Signal detection
IV. Interference canceling	Adaptive noise canceling Echo cancelation Adaptive beamforming

variation of the plant output in response to a known stationary input. The requirement is to develop an *on-line transversal filter model* for this plant, as illustrated in Fig. 8. The model consists of a finite number of unit-delay elements and a corresponding set of adjustable parameters (tap weights).

Let the available input signal at time n be denoted by the set of samples: $u(n)$, $u(n-1), \ldots, u(n-M+1)$, where M is the number of adjustable parameters in the model. This input signal is applied simultaneously to the plant and the model. Let their respective outputs be denoted by $d(n)$ and $y(n)$. The plant output $d(n)$ serves the purpose of a desired response for the adaptive filtering algorithm employed to adjust the model parameters. The model output is given by

$$y(n) = \sum_{k=0}^{M-1} \hat{w}_k(n)u(n-k) \tag{8}$$

where $\hat{w}_0(n)$, $\hat{w}_1(n)$, . . . , and $\hat{w}_{M-1}(n)$ are the estimated model parameters. The model output $y(n)$ is compared with the plant output $d(n)$. The difference between them, $d(n) - y(n)$, defines the *modeling (estimation) error*. Let this error be denoted by $e(n)$.

Typically, at time n, the modeling error $e(n)$ is nonzero, implying that the model deviates from the plant. In an attempt to account for this deviation, the error $e(n)$ is applied to an *adaptive control algorithm*. The samples of the input signal, $u(n)$, $u(n-1), \ldots, u(n-M+1)$, are also applied to the algorithm. The combination of the transversal filter and the adaptive control algorithm constitutes the adaptive filtering algorithm. The algorithm is designed to control the adjustments made in the values of the model parameters. As a result, the model parameters assume a new set of values for use on the next iteration. Thus, at time $n+1$, a new model output is computed, and with it a new value for the modeling error. The operation described is then repeated. This process is continued for a sufficiently large number of iterations (starting from time $n = 0$), until the deviation of the

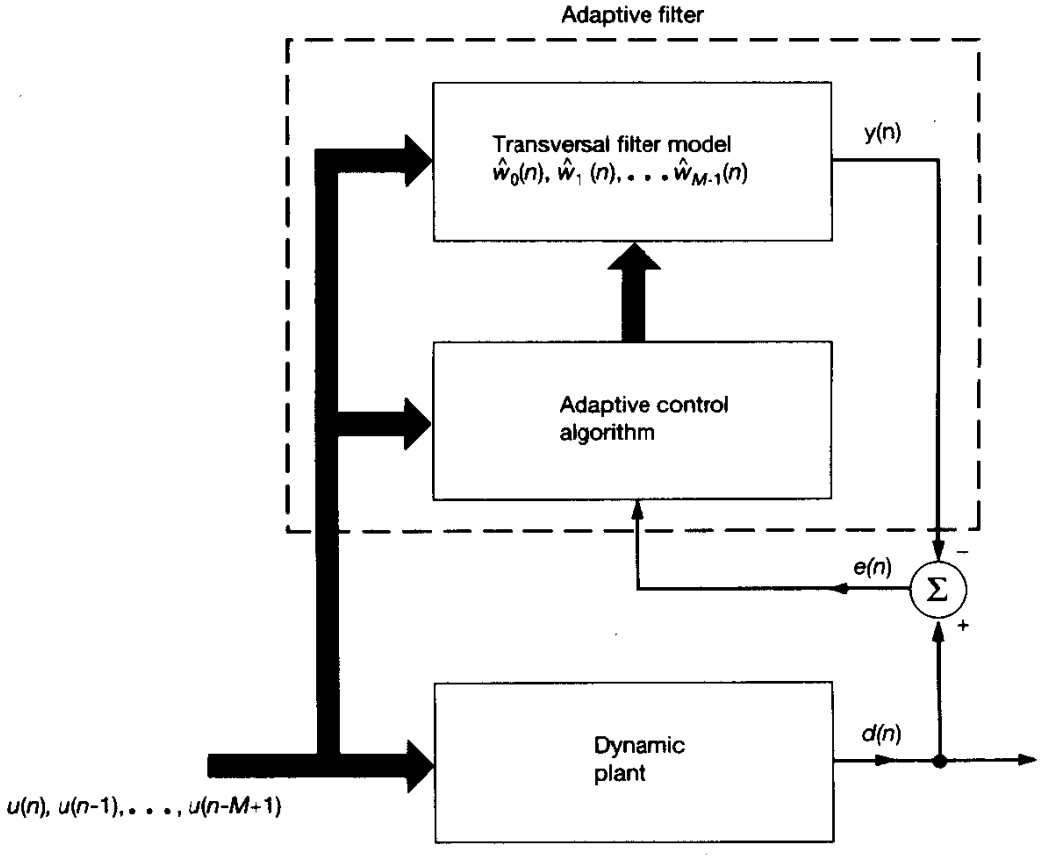


Figure 8 System identification.

model from the plant, measured by the magnitude of the modeling error $e(n)$, becomes sufficiently small in some statistical sense.

When the plant is time varying, the plant output is *nonstationary*, and so is the desired response presented to the adaptive filtering algorithm. In such a situation, the adaptive filtering algorithm has the task of not only keeping the modeling error small but also continually tracking the time variations in the dynamics of the plant.

Layered Earth Modeling

In *exploration seismology*, we usually think of a layered model of the earth (Robinson and Treitel, 1980; Justice, 1985; Mendel, 1986; Robinson and Durrani, 1986). In order to collect (record) seismic data for the purpose of characterizing such a model and thereby unraveling the complexities of the earth's surface, it is customary to use the *method of reflection seismology* that involves the following:

1. A *source of seismic energy* (e.g., dynamite, air gun) that is typically activated on the surface of the earth.

2. *Propagation* of the seismic signal away from the source and deep into the earth's crust.
3. *Reflection* of seismic waves from the interfaces between the earth's geological layers.
4. *Picking up* and *recording* the seismic returns (i.e., reflections of seismic waves from the interfaces) that carry information about the subsurface structure. On land, *geophones* (consisting of small sensors implanted into the earth) are used to pick up the seismic returns.

The method of reflection seismology, combined with a lot of signal processing, is capable of supplying a two- or three-dimensional "picture" of the earth's subsurface, down to about 20,000 to 30,000 feet and with high enough accuracy and resolution. This picture is then examined by an "interpreter" to see if it is likely that the part of the earth's subsurface (under exploration) contains hydrocarbon (petroleum) reservoirs. Accordingly, a decision is made whether or not to drill a well, which (in the final analysis) is the only way of knowing if petroleum is actually present.

A seismic wave is similar in nature to an acoustic wave, except that the earth permits the propagation of shear waves as well as compressional waves. (In an acoustic medium, only compressional waves are supported.) The earth tends to act like an *elastic medium* for the propagation of seismic waves. The property of elasticity means that a fluid or solid body resists changes in size and shape due to the applications of an external force, and that the body is restored to its original size and shape upon removal of the force. It is this property that permits the propagation of seismic waves through the earth.

An important issue in exploration seismology is the interpretation of seismic returns from the different geological layers of the earth. This interpretation is fundamental to the *identification* of crusted regions such as depth rocks, sand layers, or sedimentary layers. The sedimentary layers are of particular interest because they may contain hydrocarbon reservoirs. The idea of a layered earth model plays a key role here.

The *layered-earth model* is based on the physical fact that seismic-wave motion in each layer is characterized by two components propagating in opposite directions (Robinson and Durrani, 1986). This phenomenon is illustrated in Fig. 9. To understand the interaction between downgoing and upgoing waves, we have reproduced a portion of this diagram in Fig. 10(a), which pertains to the kth interface. The picture shown in Fig. 10(a) is decomposed into two parts, as depicted in Fig. 10(b) and 10(c). We thus observe the following:

- In layer k, there is an *upgoing* wave that consists of the superposition of the reflection of a downgoing wave incident on the kth interface (i.e., boundary) and the transmission of an upgoing (incident) wave from layer $k + 1$.
- In layer $k + 1$, there is a *downgoing* wave that consists of the superposition of the transmission of a downgoing (incident) wave from layer k and the reflection of an upgoing wave incident on the kth interface.

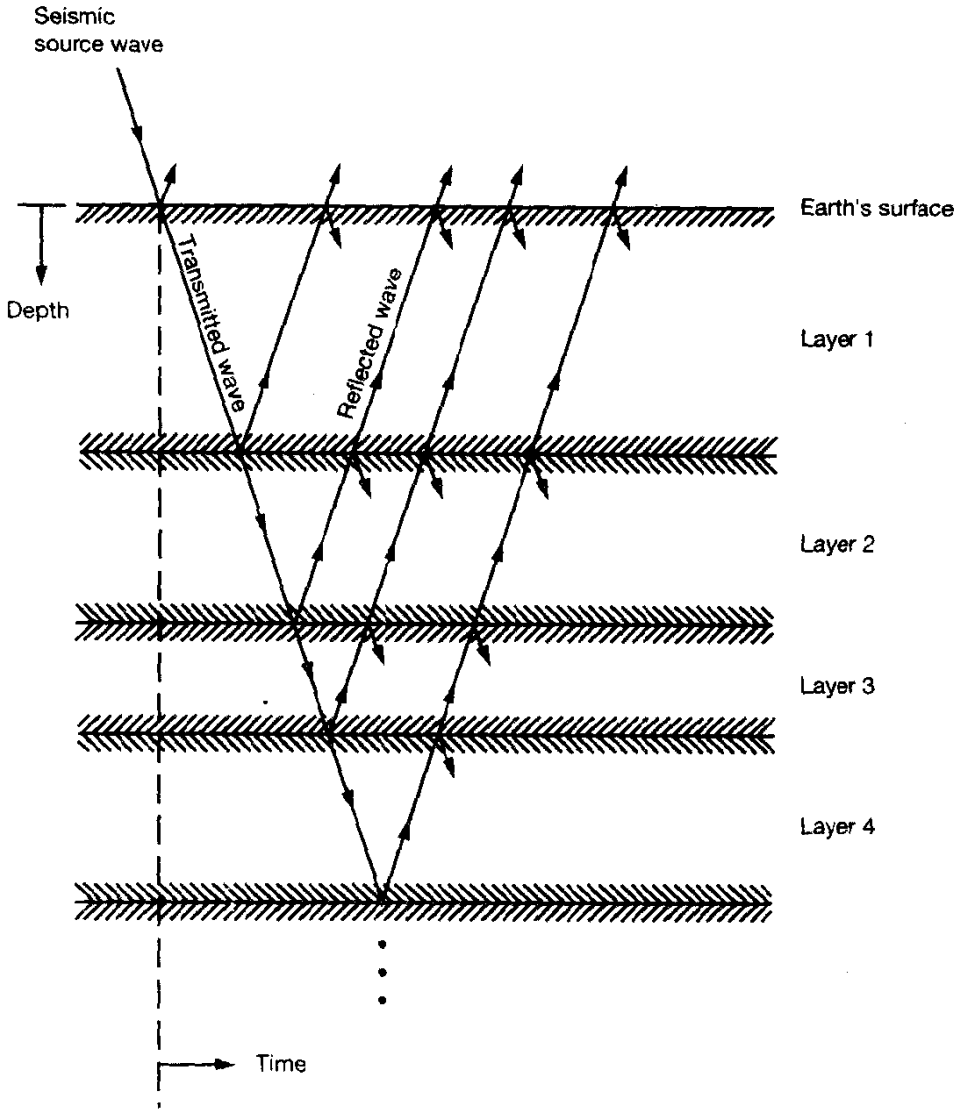


Figure 9 Upgoing and downgoing waves in different layers of the layered earth model. *Note*: The layers are unevenly spaced to add a sense of realism.

Lattice Model. Let c_k denote the upward *reflection coefficient* of the kth interface [see Fig. 10(b)]. Let $d_k(n)$ and $u_k(n)$ denote the downgoing and upgoing waves, respectively, at the *top* of layer k, and let $d_k'(n)$ and $u_k'(n)$ denote the downgoing and upgoing waves, respectively, at the *bottom* of layer k, as depicted in Fig. 10(a). The index n denotes discrete time. Ideally, the waves propagate through the medium without distortion, or absorption. Accordingly, we have from Fig. 10(a),

$$d_k'(n) = d_k(n - \tfrac{1}{2}) \tag{9}$$

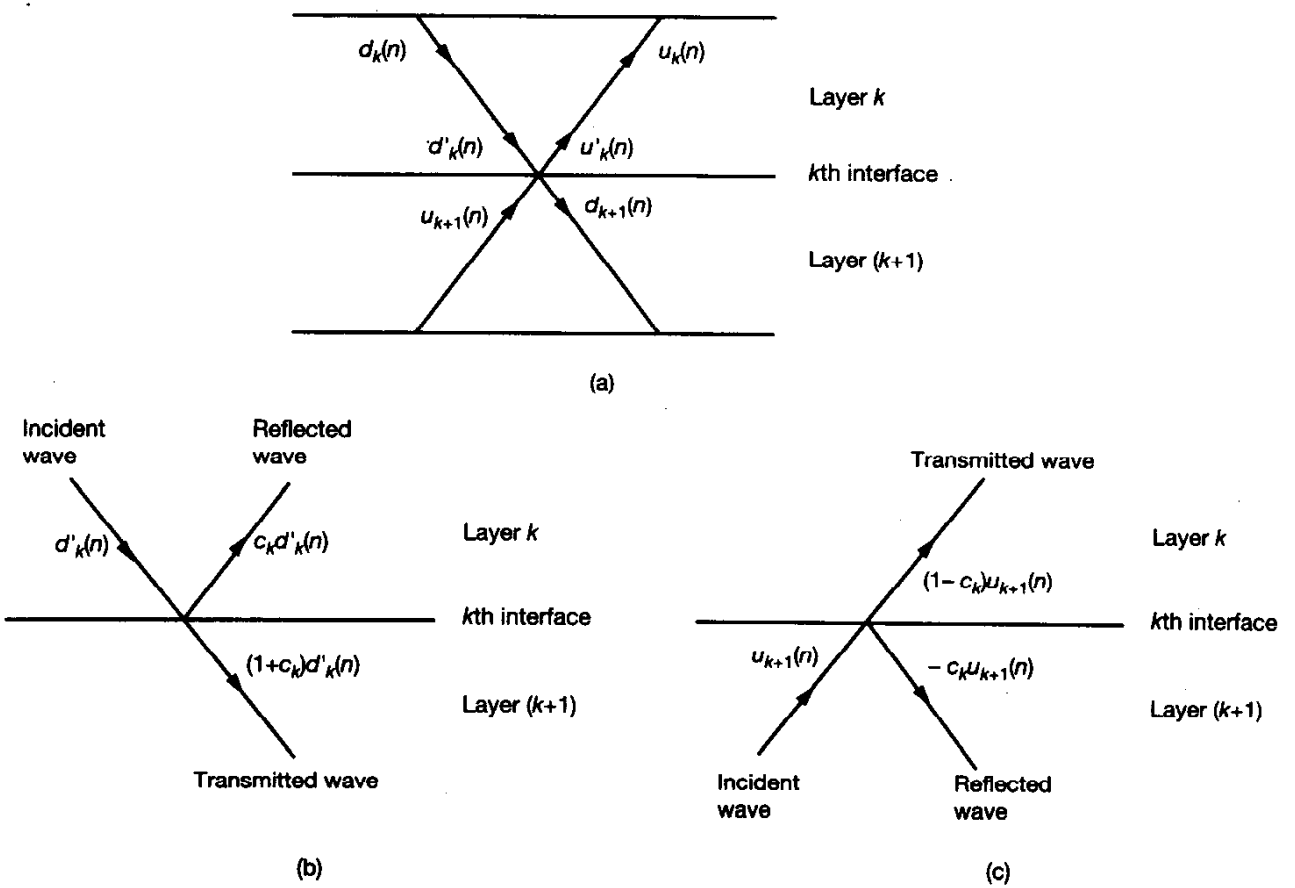


Figure 10 (a) Propagation of seismic waves through a pair of adjacent layers; (b) effects of downgoing incident wave; (c) effects of upgoing incident wave.

and

$$u_k'(n) = u_k(n + \tfrac{1}{2}) \tag{10}$$

where the travel time from the top of a layer to its bottom (or vice versa) is assumed to be one-half of a time unit. The superposition of the pictures depicted in parts (b) and (c) of Fig. 10 and comparison with that of part (a) yields the following interactions between the downgoing and upgoing waves:

$$d_{k+1}(n) = -c_k u_{k+1}(n) + (1 + c_k) d_k'(n) \tag{11}$$

and

$$u_k'(n) = c_k d_k'(n) + (1 - c_k) u_{k+1}(n) \tag{12}$$

The upward *transmission coefficient*[8] of the kth interface is defined by [see Fig. 10(c)]

$$\tau_k' = 1 - c_k \tag{13}$$

Thus, using this definition in Eq. (12), and also using this equation to eliminate $u_{k+1}(n)$ from Eq. (11), we obtain

$$u_k'(n) = c_k d_k'(n) + \tau_k' u_{k+1}(n) \tag{14}$$

and

$$d_{k+1}(n) = \frac{1}{\tau_k'} d_k'(n) - \frac{c_k}{\tau_k'} u_k'(n) \tag{15}$$

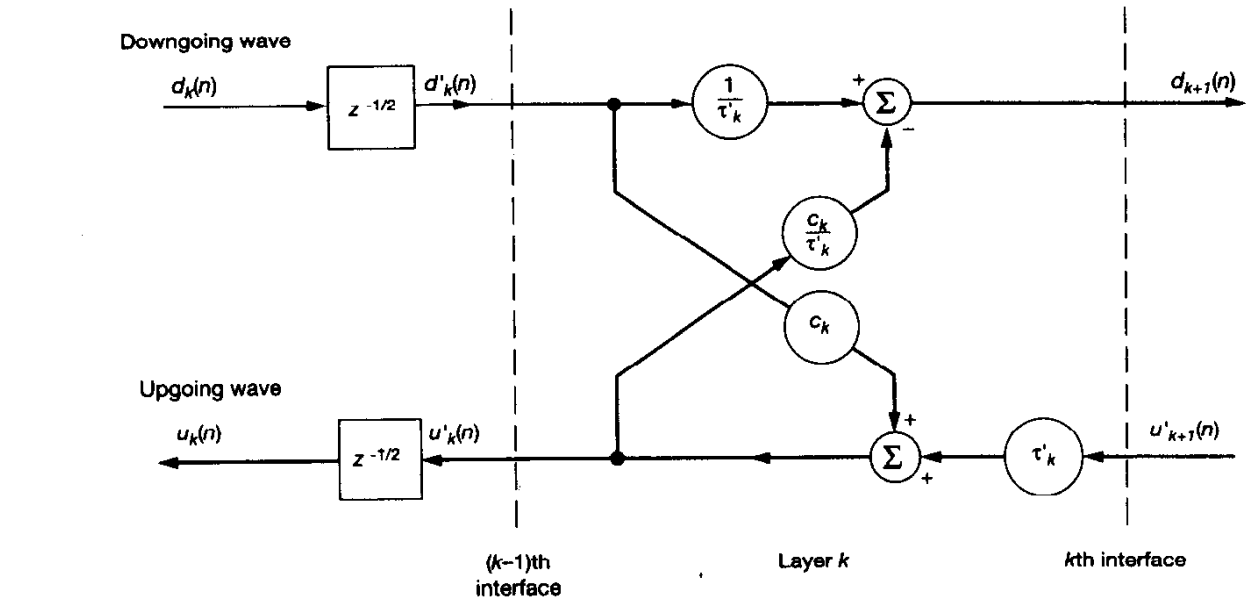
Using this pair of equations, we may construct a *lattice model* for layer k, as shown in Fig. 11(a) (Robinson and Durrani, 1986). Moreover, we may extend this idea to develop a *multistage lattice model*, shown in block diagram form in Fig. 11(b), which depicts the propagation of waves through several layers of the medium. The lattice model for each layer has the details given in Fig. 11(a). The combined use of these two figures provides a great deal of physical insight into the interaction of downgoing and upgoing waves as they propagate from one layer to the next.

Examination of Eq. (14) reveals that the evaluation of $u_k'(n)$ at the bottom of layer k requires knowledge of $u_{k+1}(n)$ at the top of layer $k + 1$. But $u_{k+1}(n)$ is not available until the layer $k + 1$ has been dealt with. The lattice model of Fig. 11 is therefore of limited practical use. To overcome this limitation, we may use the z-transform to modify this model. Specifically, applying the z-transform to Eqs. (9), (10), (14), and (15) and manipulating them into matrix form, we get the so-called *scattering equation*:

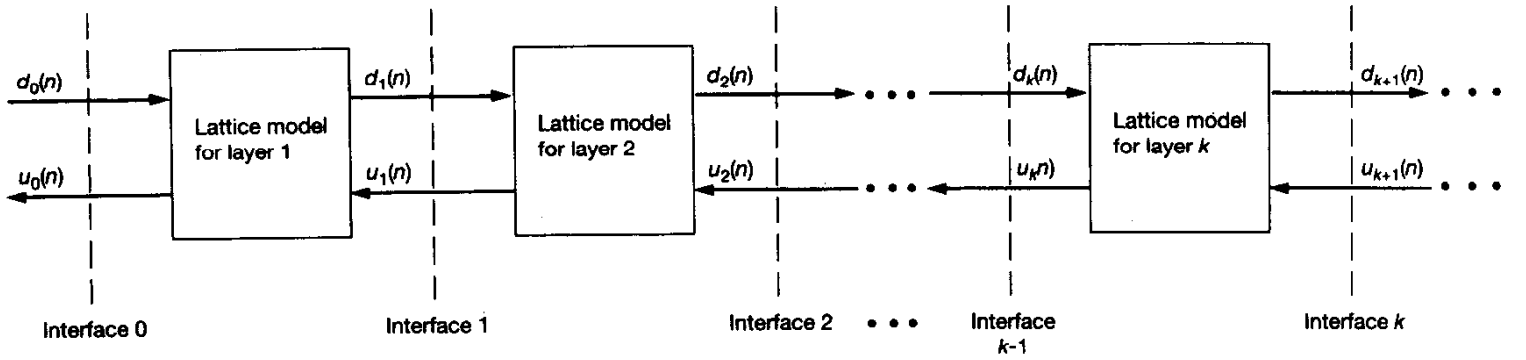
$$\begin{bmatrix} D_{k+1}(z) \\ U_{k+1}(z) \end{bmatrix} = \frac{z^{1/2}}{\tau_k'} \begin{bmatrix} z^{-1} & -c_k \\ -c_k z^{-1} & 1 \end{bmatrix} \begin{bmatrix} D_k(z) \\ U_k(z) \end{bmatrix} \tag{16}$$

[8]The prime in the upward transmission coefficient τ_k' is used to distinguish it from the *downward* transmission coefficient [see Fig. 11(a)], given by

$$\tau_k = 1 + c_k$$



(a)



(b)

Figure 11 (a) Lattice model for layer k; (b) multistage lattice model of the layered earth model, with each lattice configuration having the form given in part (a).

where z^{-1} is the unit-delay operator. The 2-by-2 matrix on the right-hand side of Eq. (16) is called the *scattering matrix*. Thus, on the basis of Eq. (16), we may construct the *modified lattice model* for layer k, shown in Fig. 12(a) (Robinson and Durrani, 1986). Correspondingly, the multistage version of the modified lattice model is as shown in Fig. 12(b).

The following points are noteworthy in the context of the modified lattice model of Fig. 12 for the propagation of compressional seismic waves in the subsurface of the earth:

1. The lattice structure of the model has *physical* significance, since it follows naturally from the notion of a layered earth.
2. The structure for each layer (state) of the model is *symmetric*.
3. The reciprocal of the transmission coefficient for each layer merely plays the role of a *scaling factor* insofar as input–output relations are concerned. Specifically, for layer k, we may remove $1/\tau_k'$ from the top path of the model in Fig. 12(a) simply by absorbing it in $D_{k+1}(z)$. Similarly, we may remove $1/\tau_k'$ from the bottom path by absorbing it in $U_{k+1}(z)$. Moreover, the values of the transmission coefficients $\tau_1', \tau_2', \ldots, \tau_k', \ldots$ are determined from the respective values of the reflection coefficients $c_1, c_2, \ldots, c_k, \ldots$ by using Eq. (13).
4. The overall model for layers $1, 2, \ldots, k, \ldots$ is *uniquely determined by the sequence of reflection coefficients* $c_1, c_2, \ldots, c_k, \ldots$.

A case of special interest arises when

$$u_{k+1}(n) = 0, \qquad k \text{ is the deepest layer} \tag{17}$$

This case corresponds to the case when the *final* interface [i.e., the $(k+1)$th interface] acts as a *perfect absorber*. In other words, there is no outgoing wave from the deepest layer, so Eq. (17) follows. This equation thus represents the *boundary condition* on the lattice model of Fig. 11. The corresponding boundary condition for the modified lattice model of Fig. 12 is

$$U_{k+1}(z) = 0, \quad k \text{ is the deepest layer} \tag{18}$$

Given this boundary condition and the sequence of reflection coefficients $c_1, c_2, \ldots, c_k, \ldots$, we may then use the modified lattice model of Fig. 12(b) (in a stage-by-stage fashion) to determine $U_0(z)$, the z-transform of the output (outgoing) seismic wave $u_0(n)$ at the earth's surface, in terms of $D_0(z)$, the z-transform of the input (downgoing) seismic wave $d_0(n)$.

Tapped-Delay-Line (Transversal Model). Figure 13 depicts a *tapped-delay-line model* for a layered earth. It provides a local parameterization of the propagation (scattering) phenomenon in the earth's subsurface. According to the alternative model, the input (downgoing) seismic wave $d_0(n)$ and the output (upgoing) seismic wave $u_0(n)$ are, in general, linearly related by the *infinite convolution sum*

$$u_0(n) = \sum_{k=0}^{\infty} w_k d_0(n-k) \tag{19}$$

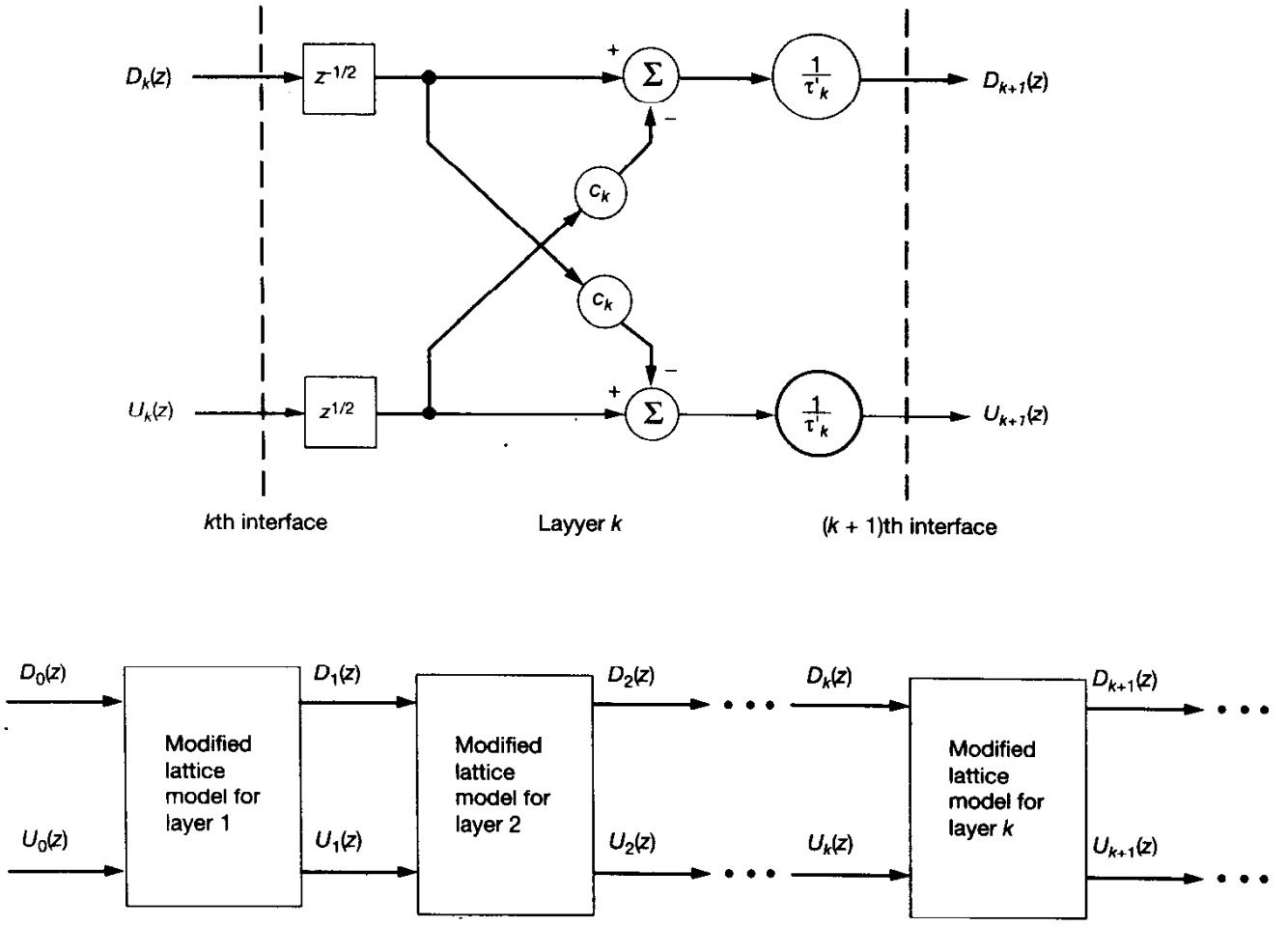


Figure 12 (a) Modified lattice model for layer k; (b) multistage version of modified lattice model, with each stage having the representation shown in part (a).

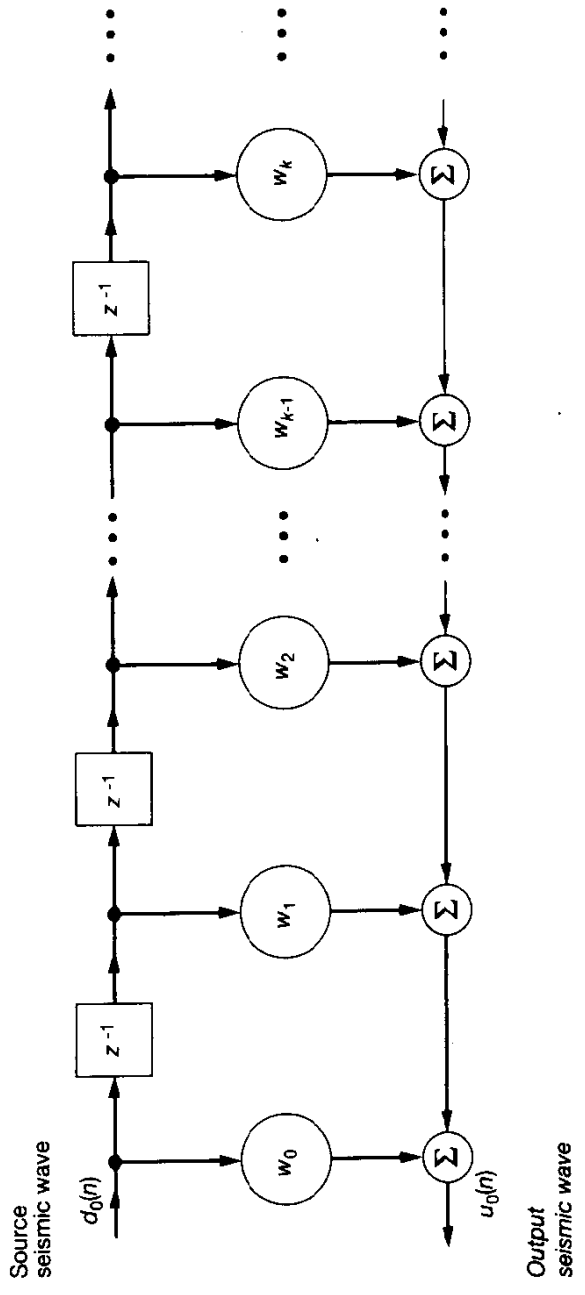


Figure 13 Tapped-delay-line model of layered earth.

where the infinite sequence of tap weights w_n represents the *spatial mapping* of the medium's weighting or the *impulse response* of the medium. Equation (19) states that the output $u_0(n)$ is an infinite series of time-delayed and scaled replicas of the input $d_0(n)$.

There is a *one-to-one correspondence* between the impulse response w_n that characterizes the tapped-delay-line model of Fig. 13 and the sequence of reflection coefficients c_n that characterizes the lattice model of Fig. 11:

$$\{w_n\} \rightleftharpoons \{c_n\} \tag{20}$$

In other words, given the c_n we may uniquely determine the w_n, and vice versa.

In reflection seismology, the model of Fig. 13 is referred to as the *convolutional model*, in view of the convolution of the impulse response of the medium with the input. This model is the starting point of seismic deconvolution (described in the next application).

Parameter Estimation.[9] The seismic wave $d_0(n)$ generated by the source of energy acts as a "probing" wave that is transmitted into the earth. Correspondingly, the seismic wave $u_0(n)$ is the output evoked by the propagation of $d_0(n)$ in the earth's subsurface. A *recorded* trace of the output $u_0(n)$ for varying time n is called a *seismogram*. Thus, given digital recordings of the probing wave $d_0(n)$ and the resulting seismogram $u_0(n)$, we may apply an adaptive filtering algorithm to estimate the impulse response w_n of the layered earth. This computation is performed *off-line*, with the probing wave $d_0(n)$ used as input to the adaptive filtering algorithm and the seismogram $u_0(n)$ serving the role of desired response for the algorithm.

Predictive Deconvolution

Convolution is fundamental to the analysis of linear time-invariant systems. Specifically, the output of a linear time-invariant system is the convolution of the input with the impulse response of the system. Convolution is *commutative*. We may therefore also say that the output of the system is the convolution of the impulse response of the system with the input. Moreover, convolution is a linear operation; it therefore holds regardless of the type of signal used as the system input.

Consider the convolutional model for reflection seismology depicted in Fig. 13. We may express the input–output relation of this model simply as

$$u_0(n) = w_n * d_0(n) \tag{21}$$

[9]For a survey of different parameter estimation procedures applicable to reflection seismology, see Mendel (1986). This paper also discusses other related issues, namely, *representation* (i.e., how something should be modeled), *measurement* (which physical parameters should be measured and how they should be measured), and *validation* (i.e., demonstrating confidence in the model). For a deterministic approach applicable to reflection seismology, see Bruckstein and Kailath (1987). The approach taken here is based on an *inverse scattering* framework for determining the parameters of a layered wave propagation medium from measurements taken at the boundary.

where $d_0(n)$ is the input, w_n is the impulse response, and $u_0(n)$ is the output. The symbol * is shorthand for convolution. The important point to note here is that given the values of w_n and $d_0(n)$ for varying n, we may determine the corresponding values of $u_0(n)$.

Deconvolution is a linear operation that *removes* the effect of some previous convolution performed on a given data record (time series). Suppose that we are given the input $d_0(n)$ and the output $u_0(n)$. We may then use deconvolution to determine the impulse response w_n. In symbolic form we may thus write

$$w_n = u_0(n) * d_0^{-1}(n) \tag{22}$$

where $d_0^{-1}(n)$ denotes the *inverse* of $d_0(n)$. Note, however, that $d_0^{-1}(n)$ is *not* the reciprocal of $d_0(n)$; rather, the use of the superscript -1 is merely a flag indicating "inverse."

In *seismic deconvolution*, we are given the seismogram $u_0(n)$ and the requirement is to unravel it so as to obtain an estimate of the impulse response w_n of a layered earth model. The problem, however, is complicated by the fact that in the general case of reflection seismology we do not have an estimate of the input seismic wave (also referred to as the seismic wavelet) $d_0(n)$. To overcome this practical uncertainty, we may use an elegant statistical procedure known as *predictive deconvolution* (Robinson, 1954; Robinson and Durrani, 1986). The term "predictive" arises from the fact that the procedure relies on the use of linear prediction. The derivation of predictive deconvolution rests on two simplifying hypotheses for seismic wave propagation with normal incidence:

1. The input wave $d_0(n)$, generated by the source of seismic energy, is the *impulse response of an all-pole feedback system*, and is thus minimum phase.
2. The impulse response w_n of the layered earth model has the properties of a *white-noise process*.

Condition 1 is referred to as the *feedback hypothesis*, and condition 2 is referred to as the *random hypothesis*. Geophysical experience over three decades has shown that it is indeed possible to satisfy these two hypotheses (Robinson, 1984). As a result, predictive deconvolution is used routinely on all seismic records in every exploration program.

The implication of the feedback hypothesis is that we may express the present value $d_0(n)$ of the input wave as a *linear combination of the past values*, as shown by

$$d_0(n) = -\sum_{k=1}^{M} a_k d_0(n-k) \tag{23}$$

where the a_k are the *feedback coefficients*, and M is the *order* of the all-pole feedback system. The order M may be fixed in advance; alternatively, it may be determined by a mean-square-error criterion.

According to the random hypothesis, the impulse response w_n has the properties of a white-noise process. We therefore expect the estimate $\hat{w}_n$ produced by the deconvolution filter in Fig. 14 to have similar properties. In other words, the deconvolution filter acts as a *whitening filter*. Furthermore, the deconvolution filter is an *all-zero* filter with a transfer

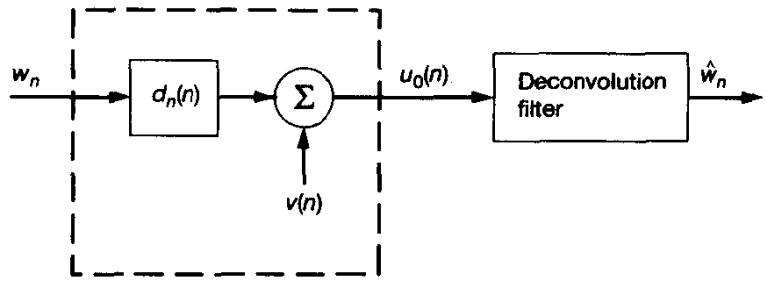


Figure 14 Block diagram illustrating seismic deconvolution.

function equal to the reciprocal of the transfer function of the all-pole feedback system used to model $d_0(n)$. This means that if we express the transfer function of the feedback system [i.e., the z-transform of $d_0(n)$] as:

$$D_0(z) = \frac{1}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_M z^{-M}} \tag{24}$$

where $a_1, a_2, \ldots, a_M$ are the *feedback coefficients*, and ignore the additive noise $v(n)$ in the model of Fig. 14, then the transfer function of the deconvolution filter is

$$\begin{aligned} A(z) &= \frac{1}{D_0(z)} \\ &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_M z^{-M} \end{aligned} \tag{25}$$

To evaluate $A(z)$, we may use a block processing approached based on the *augmented matrix form of the Wiener–Hopf equations for linear prediction*. This relation consists of a system of $(M + 1)$ simultaneous equations that involve the following:

1. A set of $(M + 1)$ known quantities represented by the *estimates* $\hat{r}(0), \hat{r}(1), \ldots, \hat{r}(M)$ of the autocorrelation function of the seismogram $u_0(n)$ for varying lags $0, 1, \ldots, M$, respectively. To get these values, we may use the formula for a *biased estimate* of the autocorrelation function:

$$\hat{r}(l) = \frac{1}{N} \sum_{n=l+1}^{N} d_0(n) d_0(n-l), \qquad l = 0, 1, \ldots, M \tag{26}$$

   where N is the *record length* of the seismogram. Typically, N is very large compared to M.

2. A set of $(M + 1)$ unknowns, made up of the feedback coefficients $a_1, a_2, \ldots, a_M$ and the variance σ^2 of the white-noise process assumed to model w_n.

Given the seismogram $u_0(n)$, we may therefore uniquely determine the feedback coefficients $a_1, a_2, \ldots, a_M$ and the variance σ^2 by solving this system of equations.

From Eq. (25), we see that the impulse response of the deconvolution filter consists of the sequence a_k, $k = 1, 2, \ldots, M$. Accordingly, the convolution of this impulse response with $u_0(n)$ yields the desired estimate $\hat{w}_n$, as shown by (see Fig. 14)

$$\hat{w}_n = \sum_{k=0}^{M} a_k u_0(n-k) \tag{27}$$

where $a_0 = 1$. Equation (27) is a description of the deconvolution process. Note, however, the wave $d_0(n)$ generated by the source of seismic energy does not enter this description directly as in the idealized representation of Eq. (23). Rather, the physical nature of $d_0(n)$ influences the deconvolution process by modeling $d_0(n)$ as the impulse response of an all-pole feedback system.

An alternative procedure for constructing the deconvolution filter is to use an adaptive filtering algorithm, as illustrated in Fig. 15. In this application, the present value $u_0(n)$ of the seismic output serves the purpose of a desired response for the algorithm, and the past values $u_0(n-1), u_0(n-2), \ldots, u_0(n-M)$ are used as elements of the input vector. The prediction error controls the adaptation of the M tap weights of the transversal filter component of the algorithm. When the algorithm has converged, the tap weights of the transversal filter provide estimates of the feedback coefficients $a_1, a_2, \ldots, a_M$.

Adaptive Equalization

In digital communications a considerable effort has been devoted to the study of data-transmission systems that utilize the available channel bandwidth efficiently. The objective here is to design a system that accommodates the highest possible rate of data transmission, subject to a specified reliability that is usually measured in terms of the error rate or average probability of symbol error. The transmission of digital data through a linear communication channel is limited by two factors:

1. *Intersymbol interference (ISI).* This is caused by dispersion in the transmit filter, the transmission medium, and the receive filter.
2. *Thermal noise.* This is generated by the receiver at its front end.

For bandwidth-limited channels (e.g., voice-grade telephone channels), we usually find that intersymbol interference is the chief determining factor in the design of high-data-rate transmission systems.

Figure 16 shows the equivalent baseband model of a binary *pulse-amplitude modulation (PAM) system.* The signal applied to the input of the transmitter part of the system consists of a *binary data sequence* b_k, in which each symbol consists of 1 or 0. This sequence is applied to a pulse generator, the output of which is filtered first in the transmitter, then by the medium, and finally in the receiver. Let $u(k)$ denote the sampled output of the receive filter in Fig. 16; the sampling is performed in synchronism with the pulse generator in the transmitter. This output is compared to a *threshold* by means of a *decision device.* If the threshold is exceeded, the receiver makes a decision in favor of symbol 1.

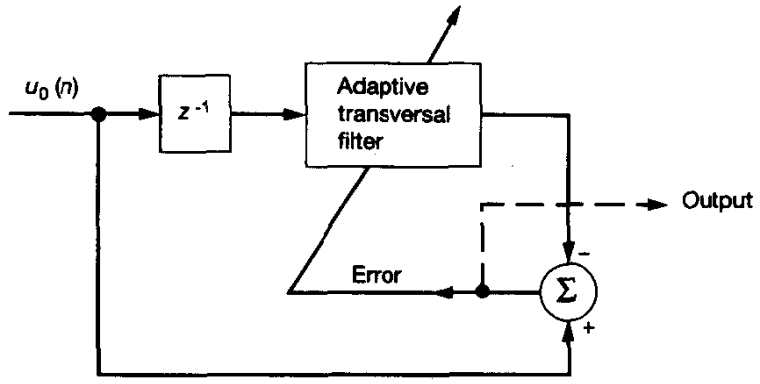


Figure 15 Adaptive filtering scheme for estimating the impulse response of the deconvolution filter.

Otherwise, it decides in favor of symbol 0.

Let a scaling factor a_k be defined by

$$a_k = \begin{cases} +1 & \text{if the input bit } b_k \text{ consists of symbol 1} \\ -1 & \text{if the input bit } b_k \text{ consists of symbol 0} \end{cases} \tag{28}$$

Then, in the absence of thermal noise, we may express $u(k)$ as

$$\begin{aligned} u(k) &= \sum_n a_n p(k-n) \\ &= a_k p(0) + \sum_{\substack{n \\ n \neq k}} a_n p(k-n) \end{aligned} \tag{29}$$

where $p(n)$ is the sampled version of the impulse response of the cascade connection of the transmit filter, the transmission medium, and the receive filter. The first term on the right-hand side of Eq. (29) defines the desired symbol, whereas the remaining series represents the intersymbol interference caused by the *channel* (i.e., the combination of the transmit filter, the medium, and the receive filter). This intersymbol interference, if left unchecked, can result in erroneous decisions when the sampled signal at the channel output is compared with some preassigned threshold by means of a decision device.

To overcome the intersymbol interference problem, control of the time-sampled function $p(n)$ is required. In principle, if the characteristics of the transmission medium are known precisely, then it is virtually always possible to design a pair of transmit and receive filters that will make the effect of intersymbol interference (at sampling times) arbitrarily small. This is achieved by proper shaping of the overall response of the channel in accordance with Nyquist's classic work on telegraph transmission theory. The overall frequency response consists of a *flat portion* and a *roll-off portion* that has a cosine form (Haykin, 1994). Correspondingly, the overall impulse response attains its maximum value at time $n = 0$ and is zero at all other sampling instants; the intersymbol interference is therefore zero. In practice we find that the channel is *time varying*, due to variations in the transmission medium, which makes the received signal *nonstationary*. Accordingly, the

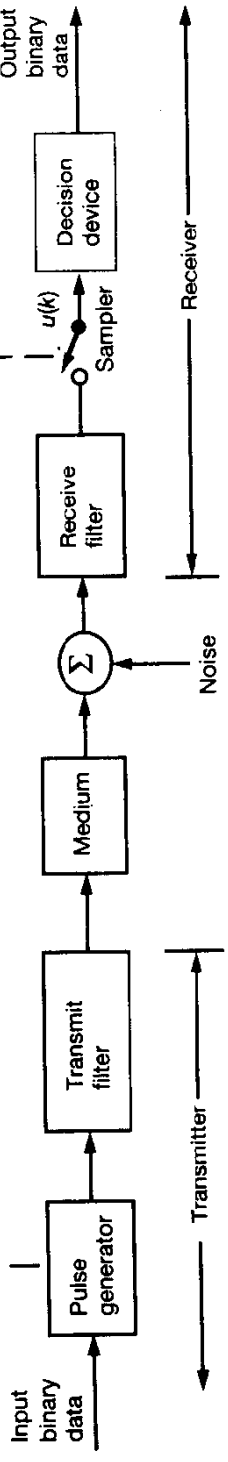


Figure 16 Block diagram of a baseband data transmission system (without equalization).

use of a fixed pair of transmit and receive filters, designed on the basis of average channel characteristics, may not adequately reduce intersymbol interference. This suggests the need for an *adaptive equalizer* that provides precise control over the time response of the channel (Lucky, 1965, 1966; Lucky et al., 1968; Proakis, 1975; Quereshi, 1985).

Among the basic philosophies for equalization of datatransmission systems are preequalization at the transmitter and postequalization at the receiver. Since the former technique requires the use of a feedback path, we will only consider equalization at the receiver, where the adaptive equalizer is placed after the receive filter–sampler combination in Fig. 16. In theory, the effect of intersymbol interference may be made arbitrarily small by making the number of adjustable coefficients (tap weights) in the adaptive equalizer infinitely large.

An adaptive filtering algorithm requires knowledge of the "desired" response so as to form the error signal needed for the adaptive process to function. In theory, the transmitted sequence (originating at the transmitter output) is the "desired" response for adaptive equalization. In practice, however, with the adaptive equalizer located in the receiver, the equalizer is physically separated from the origin of its ideal desired response. There are two methods in which a *replica* (*facsimile*) of the desired response may be generated locally in the receiver:

1. *Training method.* In the first method, a replica of the desired response is *stored* in the receiver. Naturally, the generator of this stored reference has to be electronically *synchronized* with the known transmitted sequence. A widely used *test (probing) signal* consists of a *pseudonoise (PN) sequence* (also known as a *maximal-length sequence*) with a broad and even power spectrum. The PN sequence has noiselike properties. Yet it has a deterministic waveform that repeats periodically. For the generation of a PN sequence, we may use a *linear feedback shift register* that consists of a number of consecutive two-state memory stages (flip-flops) regulated by a single timing clock (Golomb, 1964). A *feedback* term, consisting of the modulo-2 sum of the outputs of various memory stages, is applied to the first memory stage of the shift register and thereby prevents it from emptying.
2. *Decision-directed method.* Under normal operating conditions, a good facsimile of the transmitted sequence is being produced at the output of the *decision device* in the receiver. Accordingly, if this output were the correct transmitted sequence, it may be used as the "desired" response for the purpose of adaptive equalization. Such a method of learning is said to be *decision directed*, because the receiver attempts to learn by employing its own decisions (Lucky et al., 1968). If the average probability of symbol error is small (less than 10 percent, say), the decisions made by the receiver are correct enough for the *estimates of the error signal* (used in the adaptive process) to be *accurate most of the time*. This means that, in general, the adaptive equalizer is able to improve the tap-weight settings by virtue of the correlation procedure built into its feedback control loop. The improved tap-weight settings will, in turn, result in a lower average probability of

symbol error and therefore more accurate estimates of the error signal for adaptation, and so it goes on. However, it is also possible for the reverse effect to occur, in which case the tap-weight settings of the equalizer lose acquisition of the channel.

With a known training sequence, as in the first method, the adaptive filtering algorithm used to adjust the equalizer coefficients corresponds mathematically to searching for the unique minimum of a quadratic error-performance surface. The *unimodal* nature of this surface assures convergence of the algorithm. In the decision-directed method, on the other hand, the use of estimated and unreliable data modifies the error performance into a *multimodal* one, in which case complex behavior may result (Mazo, 1980). Specifically, the error performance surface now exhibits two types of local minima:

1. *Desired local minima*, whose positions correspond to coefficient (tap-weight) settings that yield the same performance as that obtained with a known training sequence
2. *Undesired (extraneous) local minima*, whose positions correspond to coefficient settings that yield inferior equalizer performance.

A poor choice of the initial coefficient settings may cause the adaptive equalizer to converge to an undesirable local minimum and stay there. The most significant point to note from this discussion is that, in general, a *linear* adaptive equalizer must be trained before it is switched to the decision-directed mode of operation if we are to be sure of delivering high performance.

A final comment pertaining to performance evaluation is in order. A popular experimental technique for assessing the performance of a data transmission system involves the use of an *eye pattern*. This pattern is obtained by applying (1) the received wave to the vertical deflection plates of an oscilloscope, and (2) a sawtooth wave at the transmitted symbol rate to the horizontal deflection plates. The resulting display is called an *eye pattern* because of its resemblance to the human eye for binary data. Thus, in a system using adaptive equalization, the equalizer attempts to correct for intersymbol interference in the system and thereby open the eye pattern as far as possible.

Thus far we have only discussed adaptive equalizers for baseband PAM systems. However, voice-band data transmission systems employ modulation–demodulation schemes that are commonly known as *modems*. Depending on the speed of operation, we may categorize modems as follows (Qureshi, 1985):

1. *Low-speed* (2400 to 4800 b/s) modems that use *phase-shift keying (PSK)*; PSK is a digital modulation scheme in which the phase of a sinusoidal carrier wave is shifted by $2\pi k/M$ radians in accordance with the input data, where M is the number of phase levels used and $k = 0, 1, \ldots, M - 1$. Specific values of M used in practice are $M = 2$ and 4, representing *binary phase-shift keying (BPSK)* and *quadriphase-shift keying (QPSK)*, respectively.

2. *High-speed* (4800 to 16,800 b/s or possibly even higher) modems that use combined *amplitude and phase modulation* or, equivalently, *quadrature amplitude modulation (QAM).*

The important point to note is that the baseband model for BPSK is real, whereas the baseband models for QPSK and QAM are complex, involving both in-phase and quadrature channels. Hence, the baseband adaptive equalizer for data transmission systems using BPSK (or its variation) is *real*, whereas the baseband adaptive equalizers for QPSK and QAM are *complex* (i.e., the tap weights of the transversal filter are complex). Note also that a real equalizer processes real inputs to produce a real equalized output, whereas a complex equalizer processes complex inputs to produce complex equalized outputs.

Blind Equalization

In the case of a highly nonstationary communications environment (e.g., digital mobile communications), it is impractical to consider the use of a training sequence. In such a situation, the adaptive filter has to equalize the communication channel in a self-organized (unsupervised) manner, and the resulting operation is referred to as *blind equalization.* Clearly, the design of a blind equalizer is a more challenging task than a conventional adaptive equalizer, because it has to make up for the absence of a training sequence by some practical means. Whereas a conventional adaptive equalizer relies on second-order statistics of the input data, a blind equalizer relies on additional information about the environment.

This additional information may take one of two basic forms:

- *Higher-order statistics (HOS),* the extraction of which is implicitly or explicitly built into the design of the blind equalizer. For this to be possible, the input data must be non-Gaussian, and the equalizer must include some form of nonlinearity.
- *Cyclostationarity,* which arises when the amplitude, phase, or frequency of a sinusoidal carrier is varied in accordance with an information-bearing signal. In this case, design of the blind equalizer is based on second-order cyclostationary statistics of the input data, and the use of nonlinearity is no longer a requirement.

An advantage of the latter type of blind equalizer is that it exhibits better convergence properties than an HOS-based blind equalizer.

Linear Predictive Coding

The coders used for the digital representation of speech signals fall into two broad classes: *source coders* and *waveform coders.* Source coders are *model dependent*, in that they use *a priori* knowledge about how the speech signal is generated at the source. Source coders for speech are generally referred to as *vocoders* (a contraction of voice coders). They can operate at low coding rates; however, they provide a synthetic quality, with the speech signal having lost substantial naturalness. Waveform coders, on the other hand, essentially

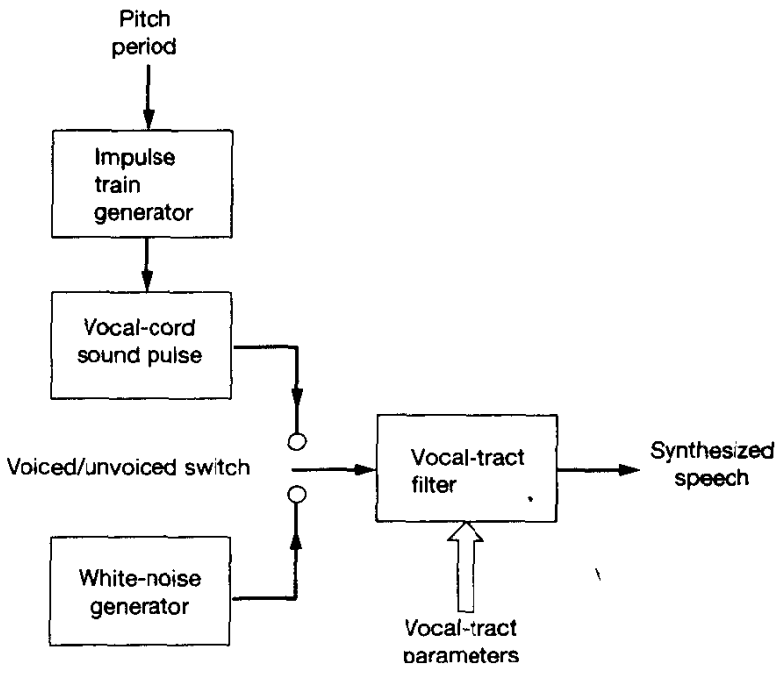


Figure 17 Block diagram of simplified model for the speech production process.

strive for facsimile reproduction of the speech waveform. In principle, these coders are *signal independent*. They may be designed to provide telephone-toll quality for speech at relatively high coding rates. In this subsection we describe a special form of source coder known as a linear predictive coder. Waveform coders are considered in the next subsection.

In the context of speech, *linear predictive coding* (LPC) strives to produce digitized voice data at low bit rates (as low as 2.4 kb/s), with two important motivations in mind. First, the use of linear predictive coding permits the transmission of digitized voice over a *narrow-band channel* (having a bandwidth of approximately 3 kHz). Second, the realization of a low-bit rate makes the *encryption* of voice signals easier and more reliable than would be the case otherwise; encryption is an essential requirement for *secure communications* (as in a military environment). Note that a bit rate of 2.4 kb/s is less than 5 percent of the 64 kb/s used typically for the standard pulse-code modulation (PCM); see the next subsection.

Linear predictive coding achieves a low bit rate for the digital representation of speech by exploiting the special properties of a classical model of the speech production process, which is described next.

Figure 17 shows a simplified block diagram of the classical model for the speech production process. It assumes that the sound-generating mechanism (i.e., the source of excitation) is linearly separable from the intelligence-modulating vocal-tract filter. The precise form of the excitation depends on whether the speech sound is voiced or unvoiced:

1. A *voiced* speech sound (such as[10] /i/ in eve) is generated from quasi-periodic vocal-cord sound. In the model of Fig. 17 the impulse-train generator produces a sequence of impulses (i.e., very short pulses), which are spaced by a fundamental period equal to the *pitch period*. This signal, in turn, excites a linear filter whose impulse response equals the vocal-cord sound pulse.
2. An *unvoiced* speech sound (such as /f/ in fish) is generated from random sound produced by turbulent airflow. In this case the excitation consists simply of a *white* (i.e., broad spectrum) noise source. The probability distribution of the noise samples does not appear to be critical.

The frequency response of the vocal-tract filter for unvoiced speech or that of the vocal tract multiplied by the spectrum of the vocal-cord sound pulses determines the short-time spectral envelope of the speech signal.

At first sight, it may appear that the speech production model falls under class I of adaptive filtering application (i.e., identification). In reality, however, this is not so. As may be seen in Fig. 17, there is *no* access to the input signal of the vocal tract.

The method of *linear predictive coding (LPC)* is an example of source coding. This method is important, because it provides not only a powerful technique for the digital transmission of speech at low bit rates but also accurate estimates of basic speech parameters.

The development of LPC relies on the model of Fig. 17 for the speech-production process. The frequency response of the vocal tract for unvoiced speech or that of the vocal tract multiplied by the spectrum of the vocal sound pulse for voiced speech is described by the *transfer function*

$$H(z) = \frac{G}{1 + \sum_{k=1}^{M} a_k z^{-k}} \tag{30}$$

where G is a gain parameter and z^{-1} is the unit-delay operator. The form of excitation applied to this filter is changed by switching between voiced and unvoiced sounds. Thus, the filter with transfer function $H(z)$ is excited by a sequence of impulses to generate voiced sounds or a white-noise sequence to generate unvoiced sounds. In this application, the input data are real valued; hence the filter coefficients, a_k, are likewise real valued.

In linear predictive coding, as the name implies, linear prediction is used to estimate the speech parameters. Given a set of past samples of a speech signal, $u(n-1), u(n-2), \ldots, u(n-M)$, a linear prediction of $u(n)$, the present sample value of the signal, is defined by

$$\hat{u}(n) = \sum_{k=1}^{M} \hat{w}_k u(n-k) \tag{31}$$

[10]The symbol / / is used to denote the *phenome*, a basic linguistic unit.

The predictor coefficients, $\hat{w}_1, \hat{w}_2, \ldots, \hat{w}_M$, are optimized by minimizing the mean-square value of the prediction error, $e(n)$, defined as the difference between $u(n)$ and $\hat{u}(n)$. The use of the minimum-mean-squared-error criterion for optimizing the predictor may be justified for two basic reasons:

1. If the speech signal satisfies the model described by Eq. (30) and if the mean-square value of the error signal $e(n)$ is minimized, then we find that $e(n)$ equals the excitation $u(n)$ multiplied by the gain parameter G in the model of Fig. 18 and $a_k = -\hat{w}_k$, $k = 1, 2, \ldots, M$. Thus, the estimation error $e(n)$ consists of quasi-periodic pulses in the case of voiced sounds or a white-noise sequence in the case of unvoiced sounds. In either case, the estimation error $e(n)$ would be small most of the time.
2. The use of the minimum-mean-squared-error criterion leads to tractable mathematics.

Figure 18 shows the block diagram of an LPC vocoder. It consists of a transmitter and a receiver. The transmitter first applies a *window* (typically 10 to 30 ms long) to the input speech signal, thereby identifying a block of speech samples for processing. This window is short enough for the vocal-tract shape to be nearly stationary, so the parameters of the speech-production model in Fig. 18 may be treated as essentially constant for the duration of the window. The transmitter then analyzes the input speech signal in an adaptive manner, block by block, by performing a linear prediction and pitch detection. Finally, it codes the parameters made up of (1) the set of predictor coefficients, (2) the pitch period, (3) the gain parameter, and (4) the voiced–unvoiced parameter, for transmission over the channel. The receiver performs the inverse operations, by first decoding the incoming parameters. In particular, it computes the values of the predictor coefficients, the pitch period, and the gain parameter, and determines whether the segment of interest represents voiced or unvoiced sound. Finally, the receiver uses these parameters to synthesize the speech signal by utilizing the model of Fig. 17.

Adaptive Differential Pulse-Code Modulation

In *pulse-code modulation*, which is the standard technique for waveform coding, three basic operation are performed on the speech signal. The three operations are *sampling* (time discretization), *quantization* (amplitude discretization), and *coding* (digital representation of discrete amplitudes). The operations of sampling and quantization are designed to preserve the shape of the speech signal. As for coding, it is merely a method of translating a discrete sequence of sample values into a more appropriate form of signal representation.

The rationale for sampling follows from a basic property of all speech signals: they are bandlimited. This means that a speech signal can be sampled in time at a finite rate in accordance with the sampling theorem. For example, commercial telephone networks designed to transmit speech signals occupy a bandwidth from 200 to 3200 Hz. To satisfy

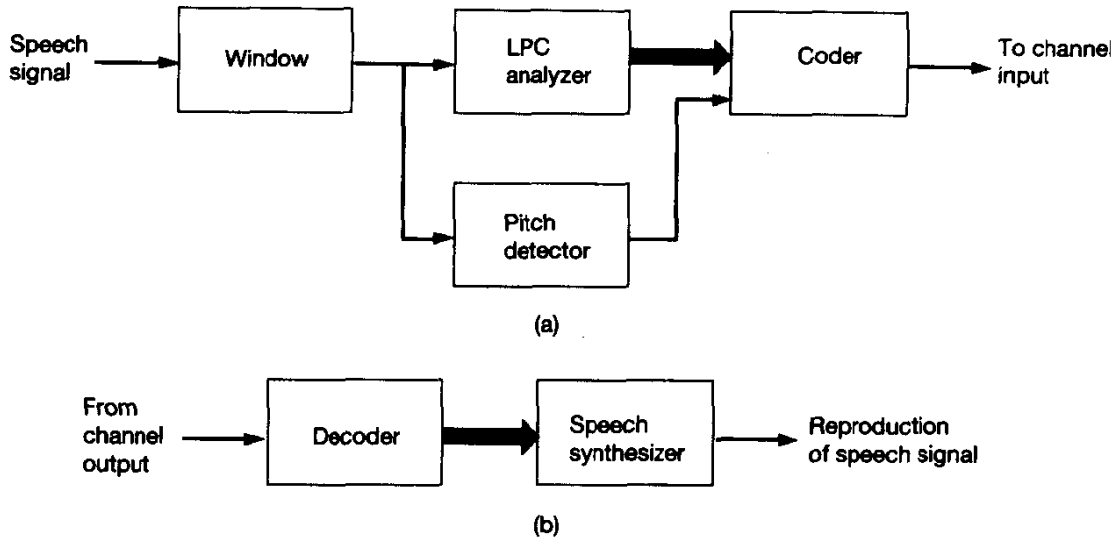


Figure 18 Block diagram of LPC vocoder: (a) transmitter, (b) receiver.

the sampling theorem, a conservative sampling rate of 8 kHz is commonly used in practice.

Quantization is justified on the following grounds. Although a speech signal has a continuous range of amplitudes (and therefore its samples also have a continuous amplitude range), it is not necessary to transmit the exact amplitudes of the samples. Basically, the human ear (as ultimate receiver) can only detect finite amplitude differences.

In PCM, as used in telephony, the speech signal (after low-pass filtering) is sampled at the rate of 8 kHz, nonlinearly (e.g., logarithmically) quantized, and then coded into 8-bit words; see Fig. 19(a). The result is a good signal-to-quantization-noise ratio over a wide dynamic range of input signal levels. This method requires a bit rate of 64 kb/s.

Differential pulse-code modulation (DPCM), another example of waveform coding, involves the use of a predictor as in Fig. 19(b). The predictor is designed to exploit the correlation that exists between adjacent samples of the speech signal, in order to realize a reduction in the number of bits required for the transmission of each sample of the speech signal and yet maintain a prescribed quality of performance. This is achieved by quantizing and then coding the prediction error that results from the subtraction of the predictor output from the input signal. If the prediction is optimized, the variance of the prediction error will be significantly smaller than that of the input signal, so a quantizer with a given number of levels can be adjusted to produce a quantizing error with a smaller variance than would be possible if the input signal were quantized directly as in a standard PCM system. Equivalently, for a quantizing error of prescribed variance, DPCM requires a smaller number of quantizing levels (and therefore a smaller bit rate) than PCM. Differential pulse-code modulation uses a fixed quantizer and a fixed predictor. A further reduction in the transmission rate can be achieved by using an adaptive quantizer together with an adaptive predictor of sufficiently high order, as in Fig. 19(c). This type of waveform coding is called *adaptive differential pulse-code modulation (ADPCM)*, where A denotes

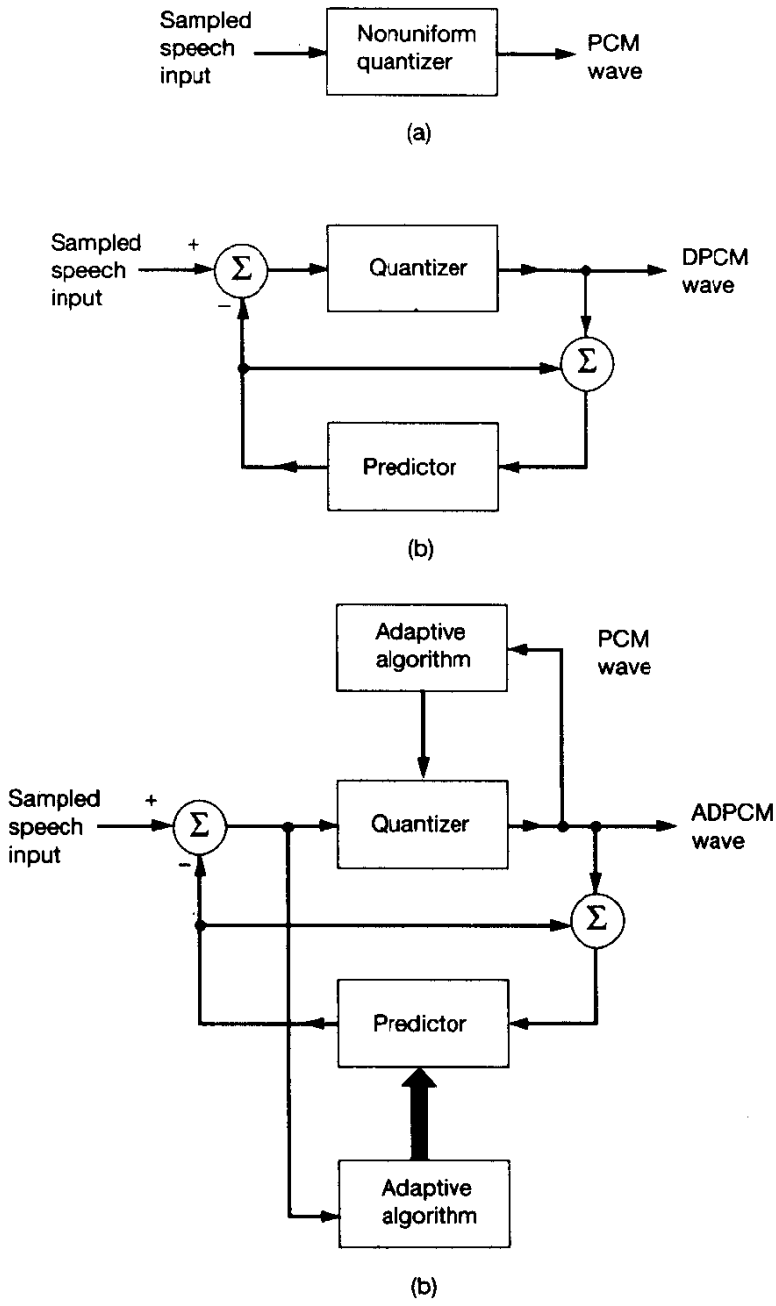


Figure 19 Waveform coders: (a) PCM, (b) DPCM, (c) ADPCM.

Figure 20 Black box representation of a stochastic model.

adaptation of both quantizer and predictor algorithms. An adaptive predictor is used in order to account for the nonstationary nature of speech signals. ADPCM can digitize speech with toll quality (8-bit PCM quality) at 32 kb/s. It can realize this level of quality with a 4-bit quantizer.[11]

Adaptive Spectrum Estimation

The *power spectrum* provides a quantitative measure of the second-order statistics of a discrete-time stochastic process as a function of frequency. In *parametric spectrum analysis*, we evaluate the power spectrum of the process by assuming a *model* for the process. In particular, the process is modeled as the output of a linear filter that is excited by a *white-noise process*, as in Fig. 20. By definition, a white-noise process has a constant power spectrum. A model that is of practical utility is the *autoregressive (AR) model*, in which the transfer function of the filter is assumed to consist of poles only. Let this transfer function be denoted by

$$H(e^{j\omega}) = \frac{1}{1 + a_1 e^{-j\omega} + \cdots + a_M e^{-jM\omega}} = \frac{1}{1 + \sum_{k=1}^{M} a_k e^{-jk\omega}} \tag{32}$$

where the a_k are called the *autoregressive (AR) parameters*, and M is the *model order*. Let σ_v^2 denote the constant power spectrum of the white-noise process $v(n)$ applied to the filter input. Accordingly, the power spectrum of the filter output $u(n)$ equals

$$S_{AR}(\omega) = \sigma_v^2 |H(e^{j\omega})|^2 \tag{33}$$

We refer to $S_{AR}(\omega)$ as the *autoregressive (AR) power spectrum*. Equation (32) assumes that the AR process $u(n)$ is real, in which case the AR parameters themselves assume real values.

[11]The International Telephone and Telegraph Consultative Committee (CCITT) has adopted the 32-kb/s ADPCM as an international standard. The adaptive predictor used herein has a transfer function consisting of two poles and six zeros. A two-pole configuration was chosen, because it permits control of decoder stability in the presence of transmission errors. Six zeros were combined with the two poles in order to improve performance. The eight coefficients of the predictor are adapted by using a simplified version of the LMS algorithm; for details, see Benvenuto et al. (1986) and Nishitani et al. (1987).

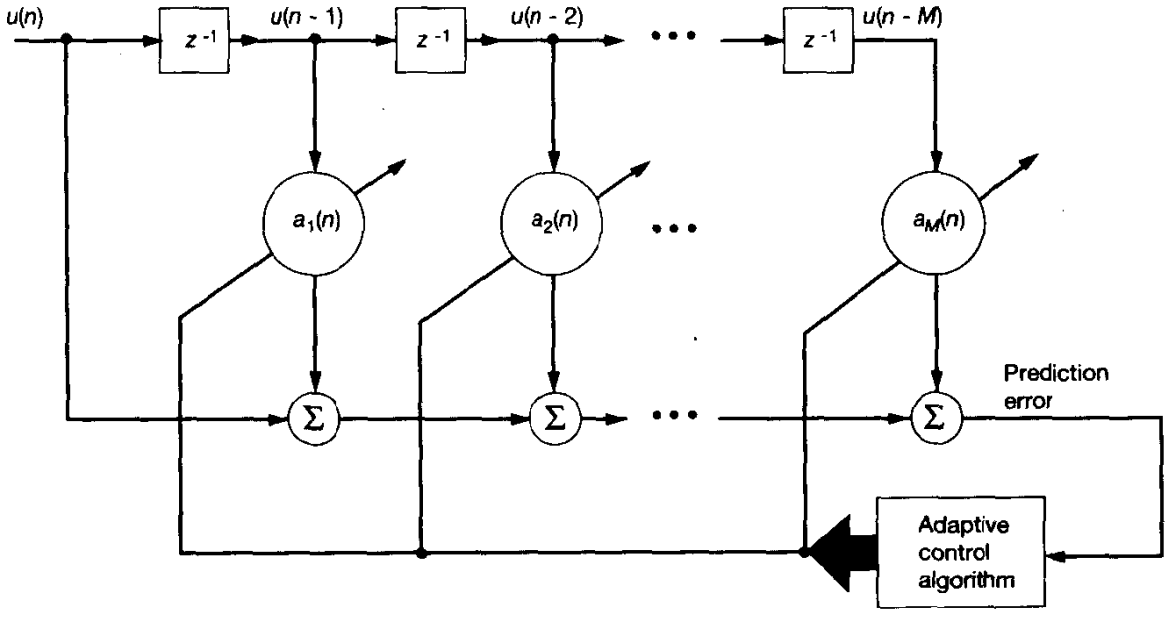


Figure 21 Adaptive prediction-error filter for real-valued data.

When the AR model is time varying, the model parameters become time dependent, as shown by $a_1(n), a_2(n), \ldots, a_M(n)$. In this case, we express the power spectrum of the time-varying AR process as

$$S_{AR}(\omega,n) = \frac{\sigma_v^2}{\left|1 + \sum_{k=1}^{M} a_k(n) e^{-jk\omega}\right|^2} \tag{34}$$

We may determine the AR parameters of the time-varying model by applying $u(n)$ to an *adaptive prediction-error filter*, as indicated in Fig. 21. The filter consists of a transversal filter with adjustable tap weights. In the adaptive scheme of Fig. 21, the prediction error produced at the output of the filter is used to control the adjustments applied to the tap weights of the filter.

The *adaptive AR model* provides a practical means for measuring the *instantaneous frequency* of a frequency-modulated process. In particular, we may do this by measuring the frequency at which the AR power spectrum $S_{AR}(\omega, n)$ attains its peak value for varying time n.

Signal Detection

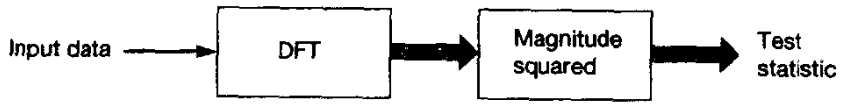
The *detection problem*, that is, the problem of detecting an information-bearing signal in noise, may be viewed as one of *hypothesis testing* with deep roots in *statistical decision*

theory (Van Trees, 1968): In the statistical formulation of hypothesis testing, there are two criteria of most interest: the *Bayes criterion* and the *Neyman–Pearson criterion*. In the Bayes test, we minimize the *average cost* or *risk* of the experiment of interest, which incorporates two sets of parameters: (1) *a priori probabilities* that represent the observer's information about the source of information before the experiment is conducted, and (2) a set of *costs* assigned to the various possible courses of action. As such, the Bayes criterion is directly applicable to digital communications. In the Neyman–Pearson test, on the other hand, we maximize the *probability of detection* subject to the constraint that the *probability of false alarm* does *not* exceed some preassigned value. Accordingly, the Neyman–Pearson criterion is directly applicable to radar or sonar. An idea of fundamental importance that emerges in hypothesis testing is that, for a Bayes criterion or Neyman–Pearson criterion, the optimum test consists of two distinct operations: (1) processing the observed data to compute a test statistic called the *likelihood ratio*, and (2) computing the likelihood ratio with a *threshold* to make a *decision* in favor of one of the two hypotheses. The choice of one criterion or the other merely affects the value assigned to the threshold. Let H_1 denote the hypothesis that the observed data consist of noise alone, and H_2 denote the hypothesis that the data consist of signal plus noise. The likelihood ratio is defined as the ratio of two maximum likelihood functions, the numerator assuming that hypothesis H_2 is true and the denominator assuming that hypothesis H_1 is true. If the likelihood ratio exceeds the threshold, the decision is made in favor of hypothesis H_2; otherwise, the decision is made in favor of hypothesis H_1.

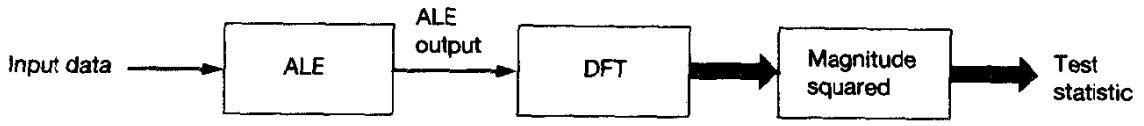
In simple binary hypothesis testing, it is assumed that the signal is known, and the noise is both white and Gaussian. In this case, the likelihood ratio test yields a *matched filter* (matched in the sense that its impulse response equals the time-reversed version of the known signal). When the additive noise is a *colored Gaussian noise* of known mean and correlation matrix, the likelihood ratio test yields a filter that consists of two sections: a *whitening filter* that transforms the colored noise component at the input into a white Gaussian noise process, and a *matched filter* that is matched to the new version of the known signal as modified by the whitening filter.

However, in some important operational environments such as *communications, radar*, and *active sonar*, there may be inadequate information on the signal and noise statistics to design a fixed optimum detector. For example, in a sonar environment it may be difficult to develop a precise *model* for the received sonar signal, one that would account for the following factors completely:

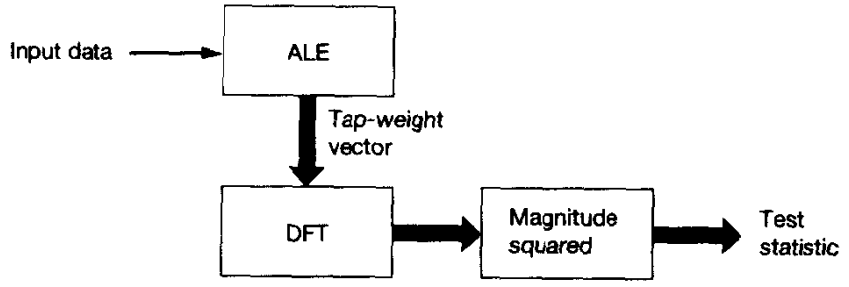
- Loss in the signal strength of a *target echo* from an object of interest (e.g., enemy vessel), due to oceanic propagation effects and reflection loss at the target
- Statistical variations in the additive *reverberation* component, produced by reflections of the transmitted signal from scatterers such as the ocean surface, ocean floor, biologies, and inhomogeneities within the ocean volume
- Potential sources of *noise* such as biological, shipping, oil drilling, seismic, and oceanographic phenomena.



(a) Conventional detector



(b) ALE output detector



(c) ALE weight detector

Figure 22 Fixed and adaptive detection schemes: (a) conventional detector. (b) ALE output detector. (c) ALE weight detector.

In situations of this kind, the use of adaptivity offers an attractive approach to solve the target (signal) detection problem. Typically, the design of an *adaptive detector* proceeds by exploiting some knowledge of general characteristics of the signal and noise, and designing the detector in such a way that its internal structure is adjustable in response to changes in the received signal. In general, the incorporation of this adjustment makes the performance analysis of an adaptive detector much more difficult to undertake than that of a fixed detector.

Fixed and adaptive detectors. Figure 22(a) shows the block diagram of a conventional detector based on the *discrete Fourier transform (DFT)* for the detection of narrow-band signals in white Gaussian noise (Williams and Ricker, 1972). The DFT may be viewed as a bank of nonoverlapping narrow-band filters whose passbands span the frequency range of interest. In the detector of Fig. 22(a) the magnitude of each complex output of the DFT is squared to form a *sufficient statistic*. This statistic is optimum (in the Neyman–Pearson sense) for detecting a sinusoid of known frequency (centered in the pertinent passband of the DFT) but unknown phase, and in the presence of white Gaussian noise. The detector output is compared to a threshold. If the threshold is exceeded, the

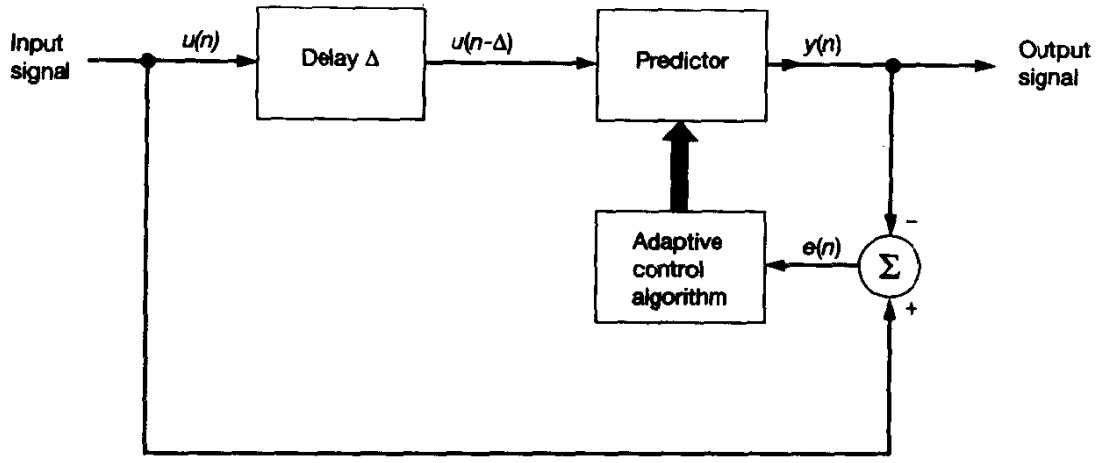


Figure 23 Adaptive line enhancer.

detector decides in favor of the narrow-band signal; otherwise, the detector declares the signal to be absent.

The performance of this conventional noncoherent detector may be improved by using an *adaptive line enhancer (ALE)* as a *prefilter* (preprocessor) to the detector (Widrow et al., 1975b). The ALE is a special form of adaptive noise canceler that is designed to suppress the wide-band noise component of the input, while passing the narrow-band signal component with little attenuation. Figure 23 depicts the block diagram of an ALE. It consists of the interconnection of a delay element and a linear predictor. The predictor output $y(n)$ is subtracted from the input signal $u(n)$ to produce the estimation error $e(n)$. This estimation error is, in turn, used to adaptively control the tap weights of the predictor. The predictor input equals $u(n - \Delta)$, where the delay Δ is equal to or greater than the sampling period. The main function of the *prediction depth* Δ is to remove the correlation between the noise component in the original input signal $u(n)$ and the delayed predictor input $u(n - \Delta)$. It is for this reason that the delay Δ is also called the *decorrelation parameter* of the ALE.

Two types of ALE detection structures have been proposed in the literature (Zeidler, 1990):

1. *ALE output detector.* In this adaptive detector shown in Fig. 22(b), the output of an ALE is applied to a DFT. The magnitude of the resulting DFT output is squared to produce the sufficient statistic for the detector.
2. *ALE weight detector.* In this second adaptive detector, shown in Fig. 22(c), the tap-weight vector of an ALE is applied to a DFT. The magnitude of the DFT output is squared as before to produce the sufficient statistic.

In both cases, the ALE processes N input data points, with the ALE length small compared to N. The real benefit of the ALE is realized in a nonstationary noise background (Zeidler, 1990).

The practical value of an ALE as a preprocessor to a conventional matched filter has been demonstrated by Nielson and Thomas (1988) as a means of improving the performance of the detector in the presence of Arctic ocean noise. This type of noise is known to have highly non-Gaussian and nonstationary characteristics; hence the benefit to be gained from the use of an ALE.

Adaptive Noise Canceling

As the name implies, adaptive noise canceling relies on the use of *noise canceling* by subtracting noise from a received signal, an operation controlled in an *adaptive* manner for the purpose of improved signal-to-noise ratio. Ordinarily, it is inadvisable to subtract noise from a received signal, because such an operation could produce disastrous results by causing an increase in the average power of the output noise. However, when proper provisions are made, and filtering and subtraction are controlled by an adaptive process, it is possible to achieve a superior system performance compared to direct filtering of the received signal (Widrow et al., 1975b; Widrow and Stearns, 1985).

Basically, an adaptive noise canceler is a *dual-input, closed-loop adaptive feedback system* as illustrated in Fig. 24. The two inputs of the system are derived from a pair of sensors: a *primary sensor* and a *reference (auxiliary) sensor*. Specifically, we have the following:

1. The primary sensor receives an *information-bearing signal* $s(n)$ corrupted by *additive noise* $v_0(n)$, as shown by

$$d(n) = s(n) + v_0(n) \tag{35}$$

The signal $s(n)$ and the noise $v_0(n)$ are uncorrelated with each other; that is,

$$E[s(n)v_0(n-k)] = 0 \qquad \text{for all } k \tag{36}$$

where $s(n)$ and $v_0(n)$ are assumed to be real valued.

2. The reference sensor receives a noise $v_1(n)$ that is *uncorrelated* with the signal $s(n)$ but *correlated* with the noise $v_0(n)$ in the primary sensor output in an *unknown* way; that is,

$$E[s(n)v_1(n-k)] = 0 \qquad \text{for all } k \tag{37}$$

and

$$E[v_0(n)v_1(n-k)] = p(k) \tag{38}$$

where, as before, the signals are real valued and $p(k)$ is an unknown cross-correlation for lag k.

The reference signal $v_1(n)$ is processed by an adaptive filter to produce the output signal:

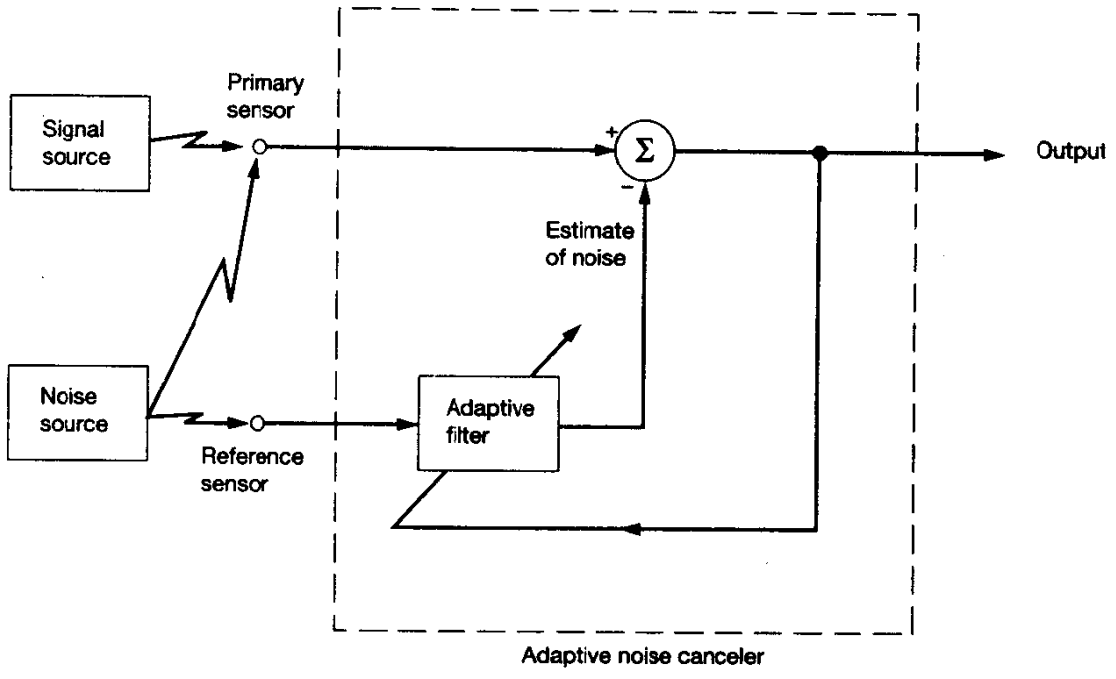


Figure 24 Adaptive noise cancelation.

$$y(n) = \sum_{k=0}^{M-1} \hat{w}_k(n) v_1(n-k) \tag{39}$$

where the $\hat{w}_k(n)$ are the adjustable (real) tap weights of the adaptive filter. The filter output $y(n)$ is subtracted from the primary signal $d(n)$, serving as the "desired" response for the adaptive filter. The error signal is defined by

$$e(n) = d(n) - y(n) \tag{40}$$

Thus, substituting Eq. (35) in (40), we get

$$e(n) = s(n) + v_0(n) - y(n) \tag{41}$$

The error signal is, in turn, used to adjust the tap weights of the adaptive filter, and the control loop around the operations of filtering and subtraction is thereby closed. Note that the information-bearing signal $s(n)$ is indeed part of the error signal $e(n)$, as indicated in Eq. (41).

The error signal $e(n)$ constitutes the overall *system output*. From Eq. (41) we see that the noise component in the system output is $v_0(n) - y(n)$. Now, the adaptive filter attempts to minimize the mean-square value (i.e., average power) of the error signal $e(n)$. The information-bearing signal $s(n)$ is essentially unaffected by the adaptive noise canceler.

Hence, minimizing the mean-square value of the error signal $e(n)$ is equivalent to minimizing the mean-square value of the output noise $v_0(n) - y(n)$. With the signal $s(n)$ remaining essentially constant, it follows that *the minimization of the mean-square value of the error signal is indeed the same as the maximization of the output signal-to-noise ratio of the system.*

The signal-processing operation described herein has two limiting cases that are noteworthy:

1. The adaptive filtering operation is *perfect* in the sense that

$$y(n) = v_0(n)$$

In this case, the system output is *noise free* and the noise cancelation is perfect. Correspondingly, the output signal-to-noise ratio is infinitely large.

2. The reference signal $v_1(n)$ is *completely uncorrelated* with both the signal and noise components of the primary signal $d(n)$; that is,

$$E[d(n)v_1(n-k)] = 0 \qquad \text{for all } k$$

In this case, the adaptive filter "switches itself off," resulting in a zero value for the output $y(n)$. Hence, the adaptive noise canceler has *no* effect on the primary signal $d(n)$, and the output signal-to-noise ratio remains unaltered.

The effective use of adaptive noise canceling therefore requires that we place the reference sensor in the noise field of the primary sensor with two specific objectives in mind. First, the information-bearing signal component of the primary sensor output is *undetectable* in the reference sensor output. Second, the reference sensor output is *highly correlated* with the noise component of the primary sensor output. Moreover, the adaptation of the adjustable filter coefficients must be near optimum.

In the remainder of this subsection, we describe three useful applications of the adaptive noise-canceling operation:

1. *Canceling 60-Hz interference in electrocardiography.* In *electrocardiography (ECG)*, commonly used to monitor heart patients, an *electrical discharge* radiates energy through a human *tissue* and the resulting output is received by an *electrode*. The electrode is usually positioned in such a way that the received energy is maximized. Typically, however, the electrical discharge involves very low potentials. Correspondingly, the received energy is very small. Hence extra care has to be exercised in minimizing signal degradation due to external *interference*. By far, the strongest form of interference is that of a 60-Hz periodic waveform picked up by the receiving electrode (acting like an antenna) from nearby electrical equipment (Huhta and Webster, 1973). Needless to say, this interference has undesirable effects in the interpretation of electrocardiograms. Widrow et al. ((1975b) have demonstrated the use of adaptive noise canceling (based on the LMS algorithm) as a method for reducing this form of interference. Specifically, the primary signal is taken from the ECG preamplifier, and the reference signal is taken from a wall outlet with proper attenuation. Figure 25 shows a block diagram of the adaptive noise canceler used

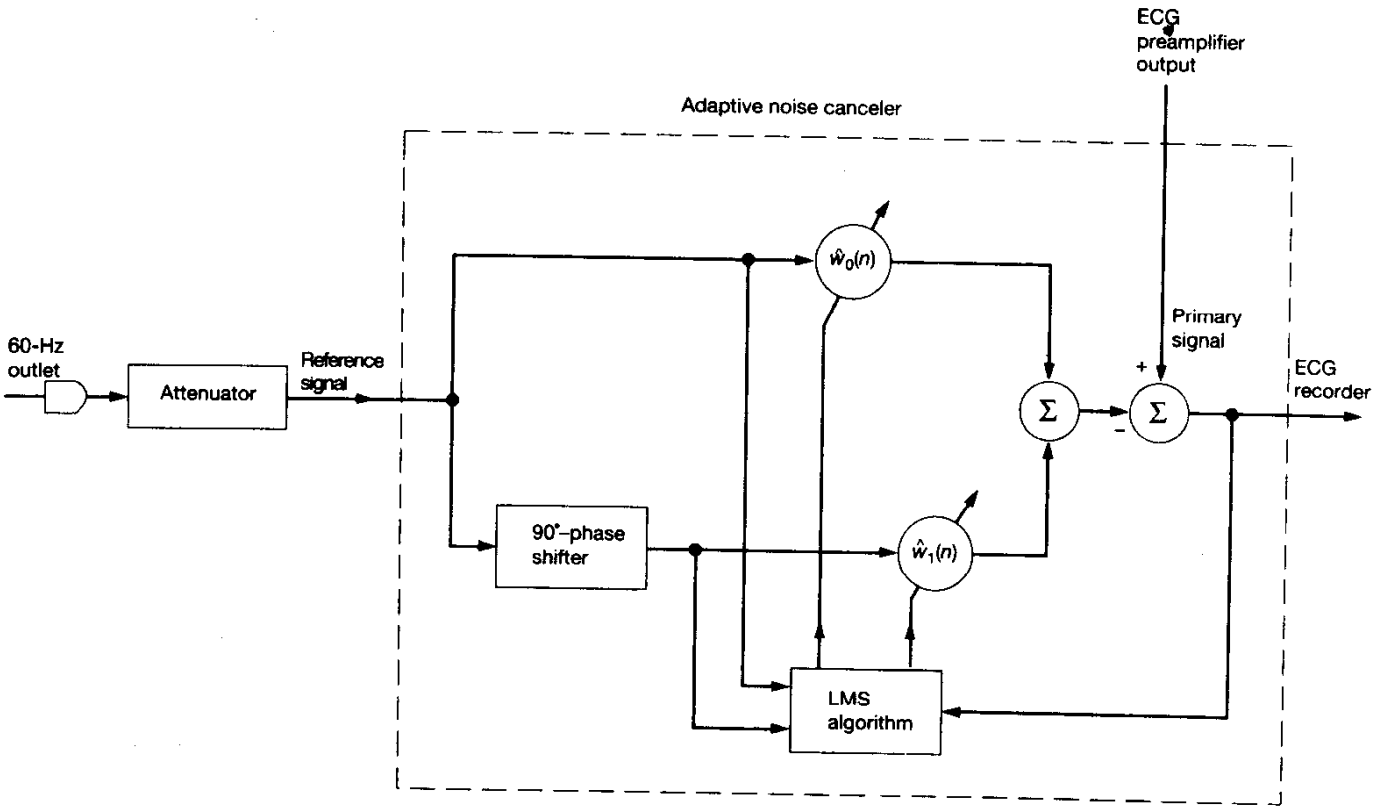


Figure 25 Adaptive noise canceler for suppressing 60-Hz interference in electrocardiography (After Widrow et al., 1975b).

by Widrow et al. (1975b). The adaptive filter has two adjustable weights, $\hat{w}_0(n)$ and $\hat{w}_1(n)$. One weight, $\hat{w}_0(n)$, is fed directly from the reference point. The second weight, $\hat{w}_1(n)$, is fed from a 90°-phase-shifted version of the reference input. The sum of the two weighted versions of the reference signal is then subtracted from the ECG output to produce an error signal. This error signal together with the weighted inputs are applied to the LMS algorithm, which, in turn, controls the adjustments applied to the two weights. In this application, the adaptive noise canceler acts as a variable "notch filter." The frequency of the sinusoidal interference in the ECG output is presumably the same as that of the sinusoidal reference signal. However, the amplitude and phase of the sinusoidal interference in the ECG output are unknown. The two weights $\hat{w}_0(n)$ and $\hat{w}_1(n)$ provide the two *degrees of freedom* required to control the amplitude and phase of the sinusoidal reference signal so as to cancel the 60-Hz interference contained in the ECG output.

2. *Reduction of acoustic noise in speech.* At a noisy site (e.g., the cockpit of a military aircraft), voice communication is affected by the presence of *acoustic noise*. This effect is particularly serious when linear predictive coding (LPC) is used for the digital representation of voice signals at low bit rates; LPC was discussed earlier. To be specific, high-frequency acoustic noise severely affects the estimated LPC spectrum in both the low- and high-frequency regions. Consequently, the intelligibility of digitized speech using LPC often falls below the minimum acceptable level. Kang and Fransen (1987) describe the use of an adaptive noise canceler, based on the LMS algorithm, for reducing acoustic noise in speech. The noise-corrupted speech is used as the primary signal. To provide the reference signal (noise only), a reference microphone is placed in a location where there is sufficient isolation from the source of speech (i.e., the known location of the speaker's mouth). In the experiments described by Kang and Fransen, a reduction of 10 to 15 dB in the acoustic noise floor is achieved, without degrading voice quality. Such a level of noise reduction is significant in improving voice quality, which may be unacceptable otherwise.

3. *Adaptive speech enhancement.* Consider the situation depicted in Fig. 26. The requirement is to listen to the voice of the desired speaker in the presence of background noise, which may be satisfied through the use of adaptive noise canceling. Specifically, *reference microphones* are added at locations far enough away from the desired speaker such that their outputs contain *only* noise. As indicated in Fig. 26, a weighted sum of the auxiliary microphone outputs is subtracted from the output of the desired speech-containing microphone, and an adaptive filtering algorithm (e.g., the LMS algorithm) is used to adjust the weights so as to minimize the average output power. A useful application of the idea described herein is in the adaptive noise cancelation for hearing aids[12] (Chazan et al., 1988). The so-called "cocktail party effect" severely limits the usefulness of hearing aids. The cocktail party phenomenon refers to the ability of a person with normal hearing to focus on a conversation taking place at a distant location in a crowded room. This ability

[12]This idea is similar to that of adaptive spatial filtering in the context of antennas, which is considered later in this section.

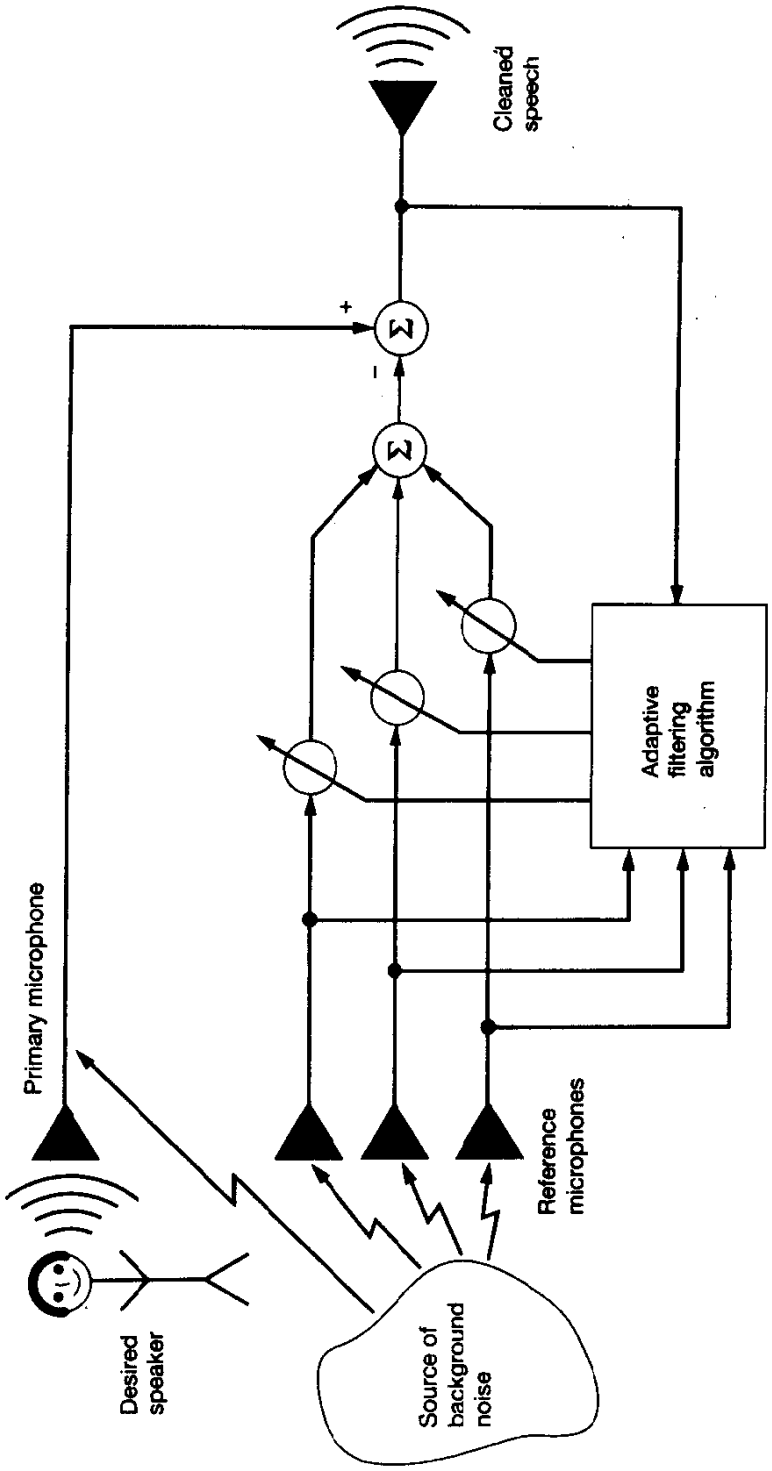


Figure 26 Block diagram of an adaptive noise canceler for speech.

is lacking in a person who wears hearing aids, because of extreme sensitivity to the presence of *background noise*. This sensitivity is attributed to two factors: (a) the loss of directional cues, and (b) the limited channel capacity of the ear caused by the reduction in both dynamic range and frequency response. Chazan et al. (1988) describe an adaptive noise canceling technique aimed at overcoming this problem. The technique involves the use of an *array of microphones* that exploit the difference in spatial characteristics between the desired signal and the noise in a crowded room. The approach taken by Chazan et al. is based on the fact that each microphone output may be viewed as the sum of the signals produced by the individual speakers engaged in conversations in the room. Each signal contribution in a particular microphone output is essentially the result of a speaker's speech signal having passed through the *room filter*. In other words, each speaker (including the desired speaker) produces a signal at the microphone output that is the sum of the direct transmission of his or her speech signal and its reflections from the walls of the room. The requirement is to reconstruct the desired speaker signal, including its room reverberations, while canceling out the source of noise. In general, the transformation undergone by the speech signal from the desired speaker is not known. Also, the characteristics of the background noise are variable. We thus have a signal-processing problem for which adaptive noise canceling offers a feasible solution.

Echo Cancelation

Almost all conversations are conducted in the presence of *echoes*. An echo may be nonnoticeable or distinct, depending on the time delay involved. If the delay between the speech and the echo is short, the echo is not noticeable but perceived as a form of spectral distortion or reverberation. If, on the other hand, the delay exceeds a few tens of milliseconds, the echo is distinctly noticeable. Distinct echoes are annoying.

Echoes may also be experienced on a telephone circuit (Sondhi and Berkley, 1980). When a speech signal encounters an *impedance mismatch* at any point on a telephone circuit, a portion of that signal is reflected (returned) as an echo. An echo represents an *impairment* that can be annoying subjectively as the more obvious impairments of low volume and noise.

To see how echoes occur, consider a long-distance telephone circuit depicted in Fig. 27. Every telephone set in a given geographical area is connected to a central office by a *two-wire line* called the *customer loop*; the two-wire line serves the need for communications in either direction. However, for circuits longer than about 35 miles, a separate path is necessary for each direction of transmission. Accordingly, there has to be provision for connecting the two-wire circuit to the four-wire circuit. This connection is accomplished by means of a *hybrid transformer*, commonly referred to as a *hybrid*. Basically, a hybrid is a bridge circuit with three ports (terminal pairs), as depicted in Fig. 28. If the bridge is *not* perfectly balanced, the "in" port of the hybrid becomes coupled to the "out" port, thereby giving rise to an echo.

Echoes are noticeable when a long-distance call is made on a telephone circuit, particularly one that includes a *geostationary satellite*. Due to the high altitude of such a satellite, there is a one-way travel time of about 300 ms between a ground station and the

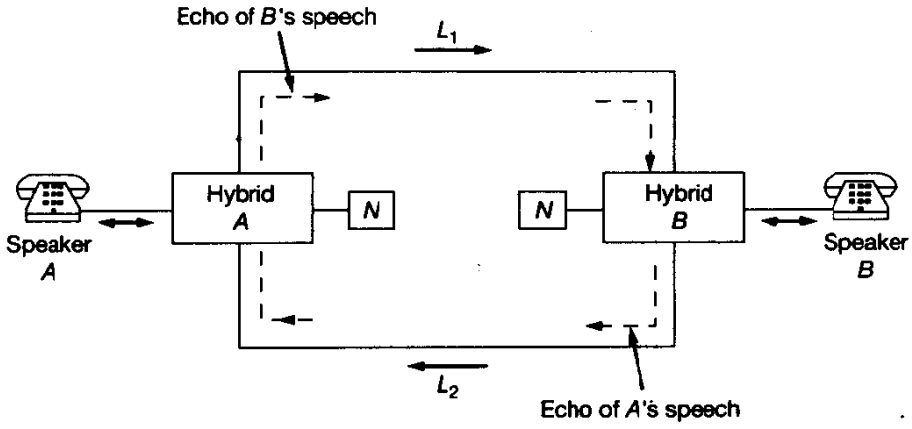


Figure 27 Long-distance telephone circuit; the boxes marked *N* are balancing impedances.

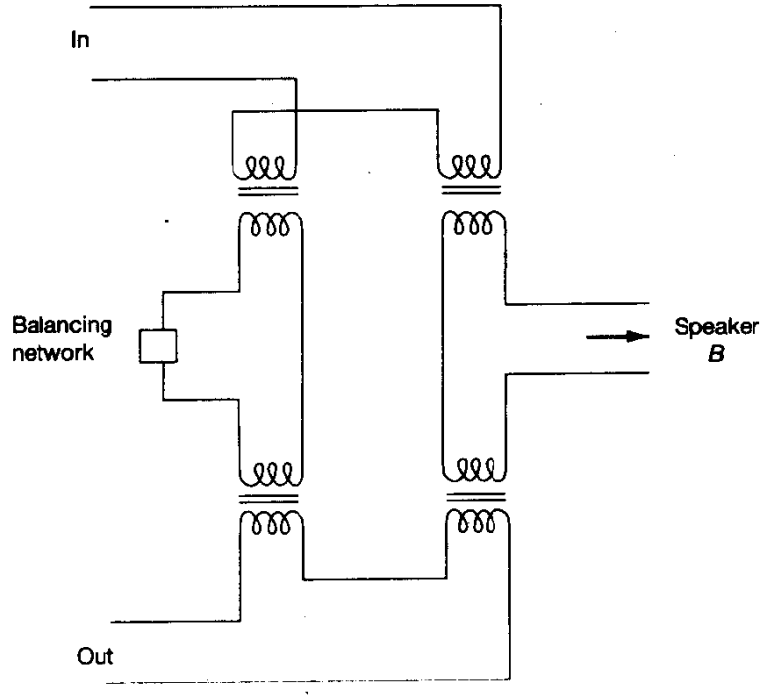


Figure 28 Hybrid circuit.

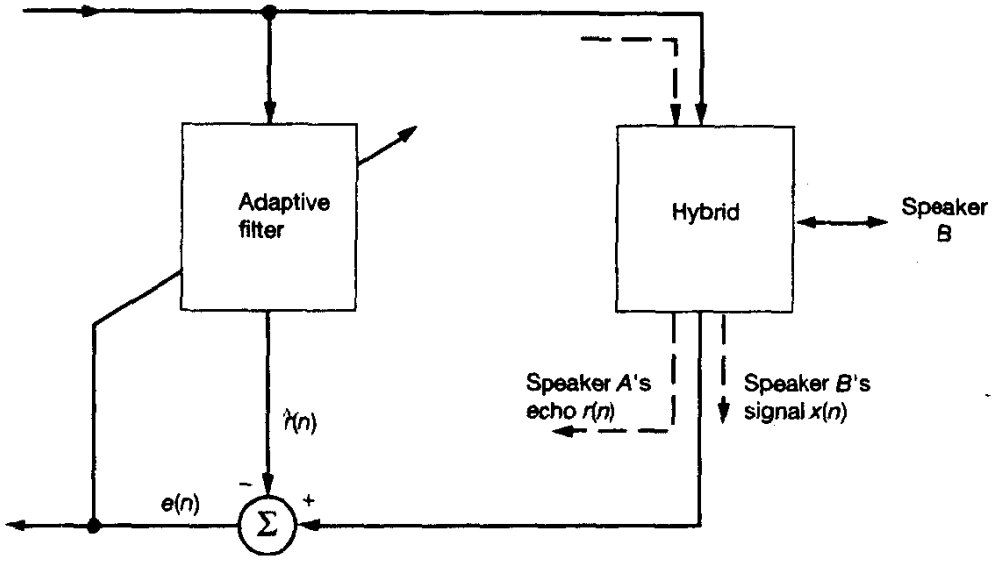


Figure 29 Signal definitions for echo cancelation.

satellite. Thus, the *round-trip delay* in a satellite link (including telephone circuits) can be as long as 600 ms. Generally speaking, the longer the echo delay, the more it must be attenuated before it becomes noticeable.

The question to be answered is: How do we exercise echo control? It appears that the idea with the greatest potential for echo control is that of *adaptive echo cancelation* (Sondhi and Prasti, 1966; Sondhi, 1967; Sondhi and Berkley, 1980; Messerschmitt, 1984; Murano et al., 1990). The basic principle of echo cancelation is to *synthesize a replica of the echo and subtract it from the returned signal.* This principle is illustrated in Fig. 29 for only one direction of transmission (from speaker A on the *far* left of the hybrid to speaker B on the right). The adaptive canceler is placed in the four-wire path *near* the origin of the echo. The synthetic echo, denoted by $\hat{r}(n)$, is generated by passing the speech signal from speaker A (i.e., the "reference" signal for the adaptive canceler) through an adaptive filter that ideally matches the transfer function of the echo path. The reference signal, passing through the hybrid, results in the echo signal $r(n)$. This echo, together with a near-end talker signal $x(n)$ (i.e., the speech signal from speaker B) constitutes the "desired" response for the adaptive canceler. The synthetic echo $\hat{r}(n)$ is subtracted from the desired response $r(n) + x(n)$ to yield the canceler error signal

$$e(n) = r(n) - \hat{r}(n) + x(n) \tag{42}$$

Note that the error signal $e(n)$ also contains the near-end talker signal $x(n)$. In any event, the error signal $e(n)$ is used to control the adjustments made in the coefficients (tap weights) of the adaptive filter. In practice, the echo path is highly variable, depending on the distance to the hybrid, the characteristics of the two-wire circuit, and so on. These variations are taken care of by the adaptive control loop built into the canceler. The control loop continuously adapts the filter coefficients to take care of fluctuations in the echo path.

For the adaptive echo cancelation circuit to operate satisfactorily, the impulse response of the adaptive filter should have a length greater than the longest echo path that needs to be accommodated. Let T_s be the sampling period of the digitized speech signal, M be the number of adjustable coefficients (tap weights) in the adaptive filter, and τ be the longest echo delay to be accommodated. We must then choose

$$MT_s > \tau \tag{43}$$

As mentioned previously (when discussing adaptive differential pulse-code modulation), the sampling rate for speech signals on the telephone network is conservatively chosen as 8 kHz, that is,

$$T_s = 125\ \mu s$$

Suppose, for example, that the echo delay $\tau = 30$ ms. Then we must choose

$$M > 240 \text{ taps}$$

Thus, the use of an echo canceler with $M = 256$ taps, say, is satisfactory for this situation.

Adaptive Beamforming

For our last application, we describe a *spatial* form of adaptive signal processing that finds practical use in radar, sonar, communications, geophysical exploration, astrophysical exploration, and biomedical signal processing.

In the particular type of spatial filtering of interest to us in this book, a number of independent *sensors* are placed at different points in space to "listen" to the received signal. In effect, the sensors provide a means of *sampling* the received signal *in space*. The set of sensor outputs collected at a particular instant of time constitutes a *snapshot*. Thus, a snapshot of data in spatial filtering (for the case when the sensors lie uniformly on a straight line) plays a role analogous to that of a set of consecutive tap inputs that exist in a transversal filter at a particular instant of time.[13]

In radar, the sensors consist of antenna elements (e.g., dipoles, horns, slotted waveguides) that respond to incident electromagnetic waves. In sonar, the sensors consist of hydrophones designed to respond to acoustic waves. In any event, spatial filtering, known as *beamforming*, is used in these systems to distinguish between the spatial properties of signal and noise. The device used to do the beamforming is called a *beamformer*. The term "beamformer" is derived from the fact that the early forms of antennas (spatial filters) were designed to form *pencil beams*, so as to receive a signal radiating from a specific direction and attenuate signals radiating from other directions of no interest (Van Veen and Buckley, 1988). Note that the beamforming applies to the radiation (transmission) or reception of energy.

[13]For a discussion of the analogies between time- and space-domain forms of signal processing, see **Bracewell (1986)** and Van Veen and Buckley (1988).

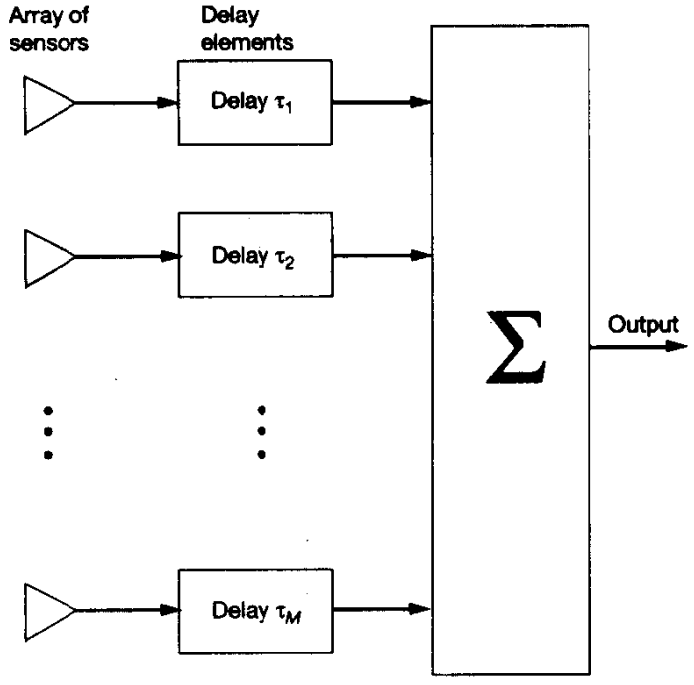


Figure 30 Delay-and-sum beamformer.

In a primitive type of spatial filtering, known as the *delay-and-sum-beamformer*, the various sensor outputs are delayed (by appropriate amounts to align signal components coming from the direction of a target) and then summed, as in Fig. 30. Thus, for a single target, the average power at the output of the delay-and-sum beamformer is maximized when it is steered toward the target. A major limitation of the delay-and-sum beamformer, however, is that it has no provisions for dealing with sources of *interference.*

In order to enable a beamformer to respond to an unknown interference environment, it has to be made *adaptive* in such a way that it places *nulls* in the direction(s) of the source(s) of interference automatically and in real time. By so doing, the output signal-to-noise ratio of the system is increased, and the *directional response* of the system is thereby improved. Below, we consider two examples of *adaptive beamformers* that are well suited for use with narrow-band signals in radar and sonar systems.

Adaptive beamformer with minimum-variance distortionless response. Consider an adaptive beamformer that uses a linear array of M identical sensors, as in Fig. 31. The individual sensor outputs, assumed to be in *baseband* form, are weighted and then summed. The beamformer has to satisfy two requirements: (1) a *steering* capability whereby the target signal is always protected, and (2) the effects of sources of interference are minimized. One method of providing for these two requirements is to

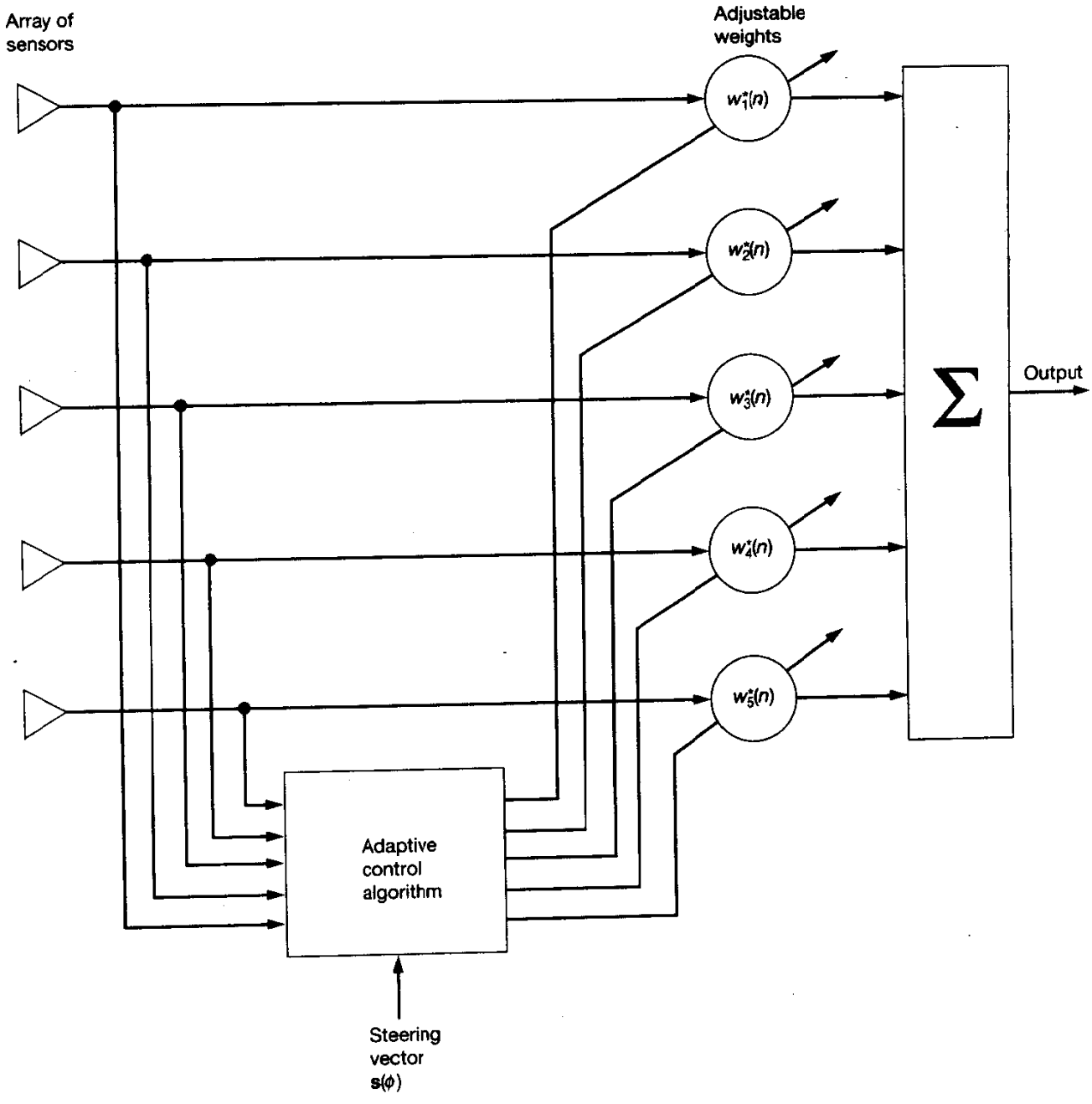


Figure 31 Adaptive beamformer for an array of 5 sensors. The sensor outputs (in baseband form) are complex valued; hence the weights are complex valued.

minimize the variance (i.e., average power) of the beamformer output, subject to the *constraint* that, during the process of adaptation, the weights satisfy the condition:

$$\mathbf{w}^H(n)\mathbf{s}(\phi) = 1 \qquad \text{for all } n, \text{ and } \phi = \phi_t \tag{44}$$

where $\mathbf{w}(n)$ is the M-by-1 weight vector and $\mathbf{s}(\phi)$ is an M-by-1 *steering vector*. The superscript H denotes Hermitian transposition (i.e., transposition combined with complex conjugation). In this application, the baseband data are complex valued; hence the need for complex conjugation. The value of *electrical angle* $\phi = \phi_t$ is determined by the direction of the target. The angle ϕ is itself measured with sensor 1 (at the top end of the array) treated as the point of reference.

The dependence of vector $\mathbf{s}(\phi)$ on the angle ϕ is defined by

$$\mathbf{s}(\phi) = [1, e^{-j\phi}, \ldots, e^{-j(M-1)\phi}]^T$$

The angle ϕ is itself related to incidence angle θ of a plane wave, measured with respect to the normal to the linear array, as follows[14]

$$\phi = \frac{2\pi d}{\lambda}\sin\theta \tag{45}$$

where d is the spacing between adjacent sensors of the array, and λ is the wavelength (see Fig. 32). The incidence angle θ lies inside the range $-\pi/2$ to $\pi/2$. The permissible values that the angle ϕ may assume lie inside the range $-\pi$ to π. This means that we must choose the spacing $d < \lambda/2$, so that there is a one-to-one correspondence between the values of θ and ϕ without ambiguity. The condition $d < \lambda/2$ may be viewed as the spatial analog of the sampling theorem.

The imposition of the *signal-protection constraint* in Eq. (44) ensures that, for a prescribed look direction, the response of the array is maintained constant (i.e., equal to 1), no matter what values are assigned to the weights. An algorithm that minimizes the variance of the beamformer output, subject to this constraint, is therefore referred to as the *minimum-variance distortionless response (MVDR) beamforming algorithm* (Capon, 1969; Owsley, 1985). The imposition of the constraint described in Eq. (44) reduces the number of "degrees of freedom" available to the MVDR algorithm to $M - 2$, where M is the number of sensors in the array. This means that the number of independent nulls produced by the MVDR algorithm (i.e., the number of independent interferences that can be canceled) is $M - 2$.

The MVDR beamforming is a special case of *linearly constrained minimum variance (LCMV) beamforming*. In the latter case, we minimize the variance of the beamformer output, subject to the constraint

$$\mathbf{w}^H(n)\mathbf{s}(\phi) = g \qquad \text{for all } n, \text{ and } \phi = \phi_t \tag{46}$$

[14]When a plane wave impinges on a linear array as in Fig. 32 there is a spatial delay of $d \sin\theta$ between the signals received at any pair of adjacent sensors. With a wavelength of λ, this spatial delay is translated into an electrical angular difference defined by $\phi = 2\pi(d \sin\theta/\lambda)$.

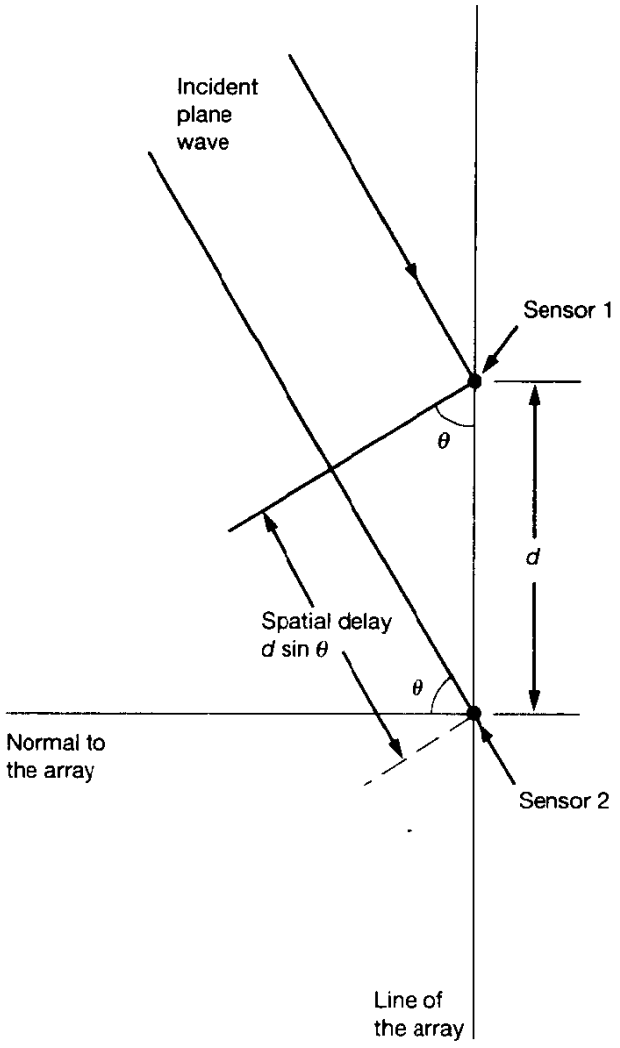


Figure 32 Spatial delay incurred when a plane wave impinges on a linear array.

where g is a complex constant. The LCMV beamformer linearly constrains the weights, such that any signal coming from electrical angle ϕ_i is passed to the output with response (gain) g. Comparing the constraint of Eq. (44) with that of Eq. (46), we see that the MVDR beamformer is indeed a special case of the LCMV beamformer for $g = 1$.

Adaptation in beam space. The MVDR beamformer performs adaptation directly in the *data space*. The adaptation process for interference cancelation may also be performed in *beam space*. To do so, the input data (received by the array of sensors) are transformed into the beam space by means of an *orthogonal multiple-beamforming network*, as illustrated in the block diagram of Fig. 33. The resulting output is processed by a *multiple sidelobe canceler* so as to cancel interference(s) from unknown directions.

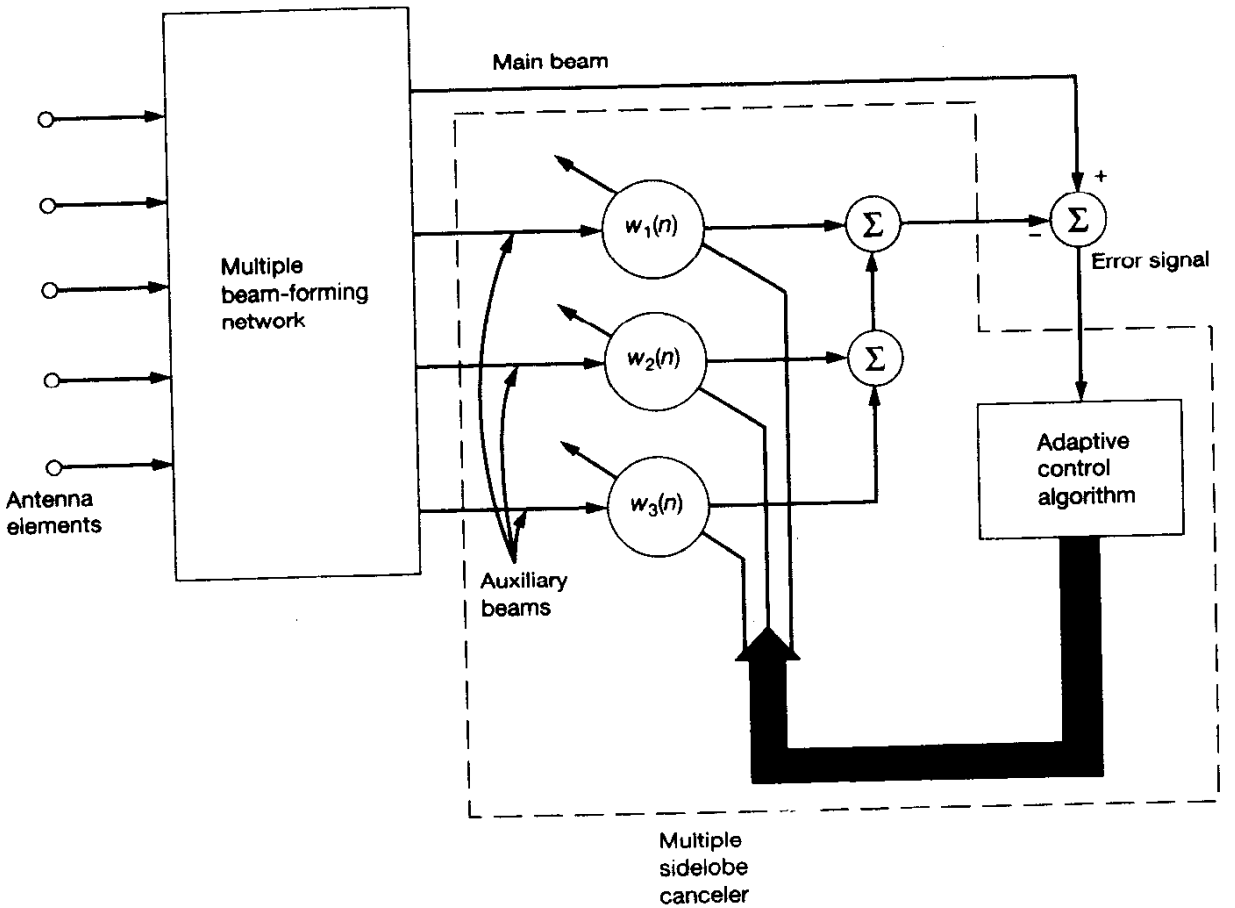


Figure 33 Block diagram of adaptive combiner with fixed beams; owing to the symmetric nature of the multiple beamforming network, final values of the weights are real valued.

The beamforming network is designed to generate a set of *orthogonal* beams. The multiple outputs of the beamforming network are referred to as *beam ports*. Assume that the sensor outputs are equally weighted and have a *uniform* phase. Under this condition, the response of the array produced by an incident plane wave arriving at the array along direction θ, measured with respect to the normal to the array, is given by

$$A(\phi, \alpha) = \sum_{n=-N}^{N} e^{jn\phi} e^{-jn\alpha} \tag{47}$$

where $M = (2N + 1)$ is the total number of sensors in the array, with the sensor at the midpoint of the array treated as the point of reference. The electrical angle ϕ is related to θ by Eq. (45), and α is a constant called the *uniform phase factor*. The quantity $A(\phi, \alpha)$ is called the *array pattern*. For $d = \lambda/2$, we find from Eq. (45) that

$$\phi = \pi \sin \theta$$

Summing the geometric series in Eq. (47), we may express the array pattern as

$$A(\phi, \alpha) = \frac{\sin[\frac{1}{2}(2N+1)(\phi - \alpha)]}{\sin[\frac{1}{2}(\phi - \alpha)]} \tag{48}$$

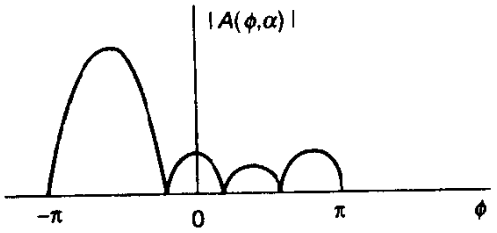
By assigning different values to α, the main beam of the antenna is thus scanned across the range $-\pi < \phi \leq \pi$. To generate an orthogonal set of beams, equal to $2N$ in number, we assign the following discrete values to the uniform phase factor

$$\alpha = \frac{\pi}{2N+1} k, \qquad k = \pm 1, \pm 3, \ldots, \pm 2N - 1 \tag{49}$$

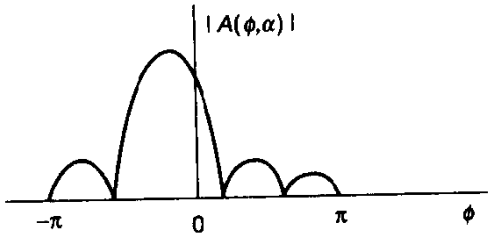
Figure 34 illustrates the variations of the magnitude of the array pattern $A(\phi, \alpha)$ with ϕ for the case of $2N + 1 = 5$ elements and $\alpha = \pm\pi/5, \pm 3\pi/5$. Note that owing to the symmetric nature of the beamformer, the final values of the weights are real valued.

The orthogonal beams generated by the beamforming network represent $2N$ independent *look directions*, one per beam. Depending on the target direction of interest, a particular beam in the set is identified as the *main beam* and the remainder are viewed as *auxiliary beams*. We note from Fig. 34 that each of the auxiliary beams has a *null in the look direction of the main beam*. The auxiliary beams are adaptively weighted by the multiple sidelobe canceler so as to form a cancelation beam that is subtracted from the main beam. The resulting estimation error is fed back to the multiple sidelobe canceler so as to control the corrections applied to its adjustable weights.

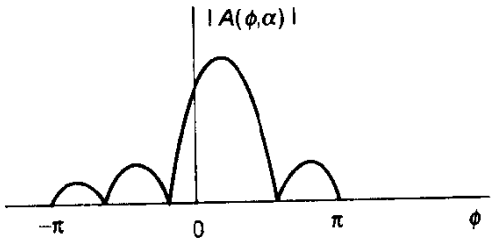
Since all the auxiliary beams have nulls in the look direction of the main beam, and the main beam is excluded from the multiple sidelobe canceler, the overall output of the adaptive beamformer is constrained to have a constant response in the look direction of the main beam (i.e., along the direction of the target). Moreover, with $(2N - 1)$ degrees of freedom (i.e., the number of available auxiliary beams) the system is capable of placing up to $(2N - 1)$ nulls along the (unknown) directions of independent interferences.



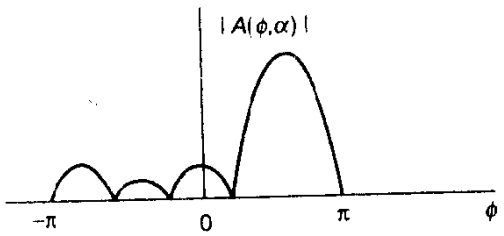
$\alpha = -\frac{3\pi}{5}$



$\alpha = -\frac{\pi}{5}$



$\alpha = +\frac{\pi}{5}$



$\alpha = +\frac{3\pi}{5}$

Figure 34 Variations of the magnitude of the array pattern $A(\phi, \alpha)$ with ϕ and α.

Note that with an array of $(2N + 1)$ sensors, we may produce a beamforming network with $(2N + 1)$ orthogonal beam ports by assigning the uniform phase factor the following set of values:

$$\alpha = \frac{k\pi}{2N - 1}, \qquad k = 0, \pm 2, \ldots, \pm 2N \tag{50}$$

In this case, a small fraction of the main lobe of the beam port at either end lies in the nonvisible region. Nevertheless, with one of the beam ports providing the main beam and the remaining $2N$ ports providing the auxiliary beams, the adaptive beamformer is now capable of producing up to $2N$ independent nulls.

8. SOME HISTORICAL NOTES

To understand a science it is necessary to know its history.
Auguste Comte (1798—1857)

We complete this introductory chapter by presenting a brief historical review of developments in four areas that are closely related insofar as the subject matter of this book is concerned. The areas are: linear estimation theory, linear adaptive filters, neural networks, and adaptive signal-processing applications.

Linear Estimation Theory

The earliest stimulus for the development of linear estimation theory[15] was apparently provided by astonomical studies in which the motion of planets and comets was studied using telescopic measurement data. The beginnings of a "theory" of estimation in which attempts are made to minimize various functions of errors can be attributed to Galileo Galilei in 1632. However, the origin of linear estimation theory is credited to Gauss who, at the age of 18 in 1795, invented the *method of least squares* to study the motion of heavenly bodies (Gauss, 1809). Nevertheless, in the early nineteenth century, there was considerable controversy regarding the actual inventor of the method of least squares. The controversy arose because Gauss did not publish his discovery in 1795. Rather, it was first published by Legendre in 1805, who independently invented the method (Legendre, 1810).

The first studies of minimum mean-square estimation in stochastic processes were made by Kolmogorov, Krein, and Wiener during the late 1930s and early 1940s (Kolmogorov, 1939; Krein, 1945; Wiener, 1949). The works of Kolmogorov and Krein were independent of Wiener's, and while there was some overlap in the results, their aims were rather different. There were many conceptual differences (as one would expect after 140 years) between Gauss's problem and the problem treated by Kolmogorov, Krein, and Wiener.

[15]The notes presented on linear estimation are influenced by the following review papers: Sorenson (1970), Kailath (1974), and Makhoul (1975).

Kolmogorov, inspired by some early work of Wold on discrete-time stationary processes (Wold, 1938), developed a comprehensive treatment of the linear prediction problem for discrete-time stochastic processes. Krein noted the relationship of Kolmogorov's results to some early work by Szegö on orthogonal polynomials (Szegö, 1939; Grenander and Szegö, 1958) and extended the results to continuous time by clever use of a bilinear transformation.

Wiener, independently, formulated the continuous-time linear prediction problem and derived an explicit formula for the optimum predictor. Wiener also considered the "filtering" problem of estimating a process corrupted by an additive "noise" process. The explicit formula for the optimum estimate required the solution of an integral equation known as the *Wiener–Hopf equation* (Wiener and Hopf, 1931).

In 1947, Levinson formulated the Wiener filtering problem in discrete time. In the case of discrete-time signals, the Wiener–Hopf equation takes on a matrix form described by[16]

$$\mathbf{R}\mathbf{w}_0 = \mathbf{p} \tag{51}$$

where $\mathbf{w}_0$ is the tap-weight vector of the optimum Wiener filter structured in the form of a transversal filter, $\mathbf{R}$ is the correlation matrix of the tap inputs, and $\mathbf{p}$ is the cross-correlation vector between the tap inputs and the desired response. For stationary inputs, the correlation matrix $\mathbf{R}$ assumes a special structure known as *Toeplitz*, so named after the mathematician O. Toeplitz. By exploiting the properties of a Toeplitz matrix, Levinson derived an elegant recursive procedure for solving the matrix form of the Wiener–Hopf equation (Levinson, 1947). In 1960, Durbin rediscovered Levinson's recursive procedure as a scheme for recursive fitting of autoregressive models to scalar time-series data (Durbin, 1960). The problem considered by Durbin is a special case of Eq. (51) in that the column vector $\mathbf{p}$ comprises the same elements found in the correlation matrix $\mathbf{R}$. In 1963, Whittle showed there is a close relationship between the Levinson–Durbin recursion and that for Szegö's orthogonal polynomials, and also derived a multivariate generalization of the Levinson–Durbin recursion (Whittle, 1963).

Wiener and Kolmogorov assumed an infinite amount of data and assumed the stochastic processes to be stationary. During the 1950s, some generalizations of the Wiener–Kolmogorov filter theory were made by various authors to cover the estimation of stationary processes given only for a finite observation interval and to cover the estimation of nonstationary processes. However, there were dissatisfactions with the most significant of the results of this period because they were rather complicated, difficult to update with increases in the observations interval, and difficult to modify for the vector case. These last two difficulties became particularly evident in the late 1950s in the problem of determining satellite orbits. In this application, there were generally vector observations of

[16]The Wiener–Hopf equation, originally formulated as an integral equation, specifies the optimum solution of a continuous-time linear filter subject to the constraint of causality. This is a difficult-to-solve equation that has resulted in the development of a considerable amount of theory, including spectral factorization. For a tutorial treatment of this subject, see Gardner (1990).

some combinations of position and velocity, and there were also large amounts of data sequentially accumulated with each pass of the satellite over a tracking station. Swerling was one of the first to tackle this problem by presenting some useful recursive algorithms (Swerling, 1958). For different reasons, Kalman independently developed a somewhat more restricted algorithm than Swerling's, but it was an algorithm that seemed particularly matched to the dynamical estimation problems that were brought by the advent of the space age (Kalman, 1960). After Kalman had published his paper and it had attained considerable fame, Swerling wrote a letter claiming priority for the Kalman filter equations (Swerling, 1963). However, history shows that Swerling's plea has fallen on deaf ears. It is ironic that orbit determination problems provided the stimulus for both Gauss's method of least squares and the Kalman filter, and that there were squabbles concerning their inventors. Kalman's original formulation of the linear filtering problem was derived for discrete-time processes. The continuous-time filter was derived by Kalman in his subsequent collaboration with Bucy; this latter solution is sometimes referred to as the *Kalman–Bucy filter* (Kalman and Bucy, 1961).

In a series of stimulating papers, Kailath reformulated the solution to the linear filtering problem by using the *innovations* approach (Kailath, 1968, 1970; Kailath and Frost, 1968; Kailath and Geesey, 1973). In this approach, a stochastic process $u(n)$ is represented as the output of a causal and causally invertible filter driven by a white-noise process $v(n)$. The white noise process $v(n)$ is called the *innovations process*, with the term "innovation" denoting "newness." The reason for this terminology is that each sample of the process $v(n)$ provides entirely new information, in the sense that it is statistically independent of all past samples of the original process $u(n)$, assuming Gaussianity; otherwise, it is only uncorrelated with all past samples of $u(n)$. The idea of innovations approach was introduced by Kolmogorov (1941).

Linear Adaptive Filters

Stochastic gradient algorithms. The earliest work on adaptive filters may be traced back to the late 1950s, during which time a number of researchers were working independently on different applications of adaptive filters. From this early work, the *least-mean-square (LMS) algorithm* emerged as a simple and yet effective algorithm for the operation of adaptive transversal filters. The LMS algorithm was devised by Widrow and Hoff in 1959 in their study of a pattern recognition scheme known as the *adaptive linear* (threshold logic) *element*, commonly referred to in the literature as the *Adaline* (Widrow and Hoff, 1960; Widrow, 1970). The LMS algorithm is a stochastic gradient algorithm in that it iterates each tap weight of a transversal filter in the direction of the gradient of the squared magnitude of an error signal with respect to the tap weight. As such, the LMS algorithm is closely related to the concept of *stochastic approximation* developed by Robbins and Monro (1951) in statistics for solving certain sequential parameter estimation problems. The primary difference between them is that the LMS algorithm uses a fixed step-size parameter to control the correction applied to each tap weight from one iteration

to the next, whereas in stochastic approximation methods the step-size parameter is made inversely proportional to time n or to a power of n. Another stochastic gradient algorithm, closely related to the LMS algorithm, is the *gradient adaptive lattice (GAL) algorithm* (Griffiths, 1977, 1978); the difference between them is structural in that the GAL algorithm is lattice-based, whereas the LMS algorithm uses a transversal filter.

In 1981, Zames introduced the H^∞ *norm* (or *minimax criterion*) as a robust index of performance for solving problems in estimation and control, and with it the field of robust control took on a new research direction. In this context, it is particularly noteworthy that Sayed and Rupp (1994) have shown that the LMS algorithm is indeed optimal under the H^∞ criterion. Thus, for the first time, theoretical evidence was presented for the robust performance of the LMS algorithm. It is also of interest to note that the zero-forcing algorithm, which represents an alternative to the LMS algorithm for the adaptive equalization of communication channels, also uses a minimax type of performance criterion (Lucky, 1965).

Recursive least-squares algorithms. Turning next to the recursive least-squares (RLS) family of adaptive filtering algorithms, the original paper on the *standard RLS algorithm* appears to be that of Plackett (1950), though it must be said that many other investigators have derived and rederived the RLS algorithm. In 1974, Godard used Kalman filter theory to derive a variant of the RLS algorithm, which is sometimes referred to in the literature as the *Godard algorithm*. Although prior to this date, several investigators had applied Kalman filter theory to solve the adaptive filtering problem, Godard's approach was widely accepted as the most successful application of Kalman filter theory for a span of two decades. Then, Sayed and Kailath (1994) published an expository paper, in which the *exact* relationship between the RLS algorithm and Kalman filter theory was delineated for the first time, thereby laying the groundwork for how to exploit the vast literature on Kalman filters for solving linear adaptive filtering problems.

In 1981, Gentleman and Kung introduced a numerically robust method, based on the *QR-decomposition* of matrix algebra, for solving the recursive least-squares problem. The resulting adaptive filter structure, sometimes referred to as the *Gentleman–Kung (systolic) array*, was subsequently refined and extended in various ways by many other investigators.

In the 1970s and during subsequent years, a great deal of research effort was expended on the development of numerically stable *fast RLS algorithms*, with the aim of reducing computational complexity to a level comparable to that of the LMS algorithm. In one form or another, the development of these algorithms can be traced back to results derived by Morf in 1974 for solving the deterministic counterpart of the stochastic filtering problem solved efficiently by the Levinson–Durbin algorithm for stationary inputs.

Returning to the paper by Sayed and Kailath (1994), the one-to-one correspondences between RLS and Kalman variables was exploited in that paper to show that QR-decomposition-based RLS algorithms and fast RLS algorithms are all in fact special cases of the Kalman filter, thereby providing a unified treatment of the RLS family of linear adaptive filters in a rather elegant and compact fashion.

Neural Networks[17]

Research interest in neural networks began with the pioneering work of McCulloch and Pitts (1943), who described a logical calculus for neural networks. Then, in 1958, Rosenblatt introduced a new approach to the pattern-classification problem using a neural network known as the *perceptron*. Out of this early work on neural networks, the LMS algorithm was pioneered by Widrow and Hoff in 1959, which, as mentioned previously, was used to formulate the Adaline. In the 1960s, it seemed as if neural networks could solve any problem. But then came the book by Minsky and Papert (1969), who used elegant mathematics to demonstrate that there are fundamental limits on what single-layer perceptrons can compute, and with it interest in neural networks took a sharp downturn.

In 1986, successful development of the *back-propagation algorithm* was reported by Rumelhart, Hinton, and Williams as a device for the training of multilayer perceptrons; the back-propagation algorithm is a generalization of the LMS algorithm. In that same year, the two-volume seminal book, *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*, with Rumelhart and McClelland as editors, was published. This book has been a major influence in reviving interest in the use of neural networks. After the publication of this book, however, it became known that the back-propagation algorithm had actually been described earlier by Werbos in his Ph.D. thesis at Harvard University in 1974.

The multilayer perceptron represents one important type of feedforward layered network that is well suited for adaptive signal processing. Another equally important feedforward layered network is the *radial-basis function (RBF) network*, which was described by Broomhead and Lowe in 1988. However, the basic idea of RBF networks may be traced back to earlier work by Bashkirov, Braverman, and Muchnick in 1964 on the method of potential functions.

The field of neural networks encompasses many other types of network structures and learning algorithms. Indeed, they have been established as an interdisciplinary subject with deep roots in the neurosciences, psychology, mathematics, the physical sciences, and engineering. Needless to say, they have a major impact on adaptive signal processing, particularly in those applications that require the use of nonlinearity.

Adaptive Signal-Processing Applications

Adaptive Equalization. Until the early 1960s, the equalization of telephone channels to combat the degrading effects of intersymbol interference on data transmission was performed by using either fixed equalizers (resulting in a performance loss) or equalizers whose parameters were adjusted manually (a rather cumbersome procedure). In 1965, Lucky made a major breakthrough in the equalization problem by proposing a *zero-forcing algorithm* for automatically adjusting the tap weights of a transversal equalizer. A distinguishing feature of the work by Lucky was the use of a *minimax* type of performance

[17]For a more complete historical account of neural networks, see Cowan (1990) and Haykin (1994).

criterion. In particular, he used a performance index called *peak distortion*, which is directly related to the maximum value of intersymbol interference that can occur. The tap weights in the equalizer are adjusted to minimize the peak distortion. This has the effect of *forcing* the intersymbol interference due to those adjacent pulses that are contained in the transversal equalizer to become *zero*; hence the name of the algorithm. A sufficient, but not necessary, condition for optimality of the zero-forcing algorithm is that the *initial distortion* (the distortion that exists at the equalizer input) be less than unity. In a subsequent paper published in 1966, Lucky extended the use of the zero-forcing algorithm to the tracking mode of operation. In 1965, DiToro independently used adaptive equalization for combatting the effect of intersymbol interference on data transmitted over high-frequency links.

The pioneering work by Lucky inspired many other significant contributions to different aspects of the adaptive equalization problem in one way or another. Gersho (1969) and Proakis and Miller (1969) independently reformulated the adaptive equalization problem using a mean-square-error criterion. In 1972, Ungerboeck presented a detailed mathematical analysis of the convergence properties of an adaptive transversal equalizer using the LMS algorithm. In 1974, as mentioned previously, Godard used Kalman filter theory to derive a powerful algorithm for adjusting the tap weights of a transversal equalizer. In 1978, Falconer and Ljung presented a modification of this algorithm that simplified its computational complexity to a level comparable to that of the simple LMS algorithm. Satorius and Alexander (1979) and Satorius and Pack (1981) demonstrated the usefulness of lattice-based algorithms for adaptive equalization of dispersive channels.

This brief historical review pertains to the use of adaptive equalizers for *linear synchronous receivers*; by "synchronous" we mean that the equalizer in the receiver has its taps spaced at the reciprocal of the symbol rate. Even though our interest in adaptive equalizers is largely restricted to this class of receivers, nevertheless, such a historical review would be incomplete without some mention of fractionally spaced equalizers and decision-feedback equalizers.

In a *fractionally spaced equalizer (FSE)*, the equalizer taps are spaced closer than the reciprocal of the symbol rate. An FSE has the capability of compensating for delay distortion much more effectively than a conventional synchronous equalizer. Another advantage of the FSE is the fact that data transmission may begin with an arbitrary sampling phase. However, mathematical analysis of the FSE is much more complicated than for a conventional synchronous equalizer. It appears that early work on the FSE was initiated by Brady (1970). Other contributions to the subject include subsequent work by Ungerboeck (1976) and Gitlin and Weinstein (1981).

A *decision-feedback equalizer* consists of a feedforward section and a feedback section connected as shown in Fig. 35. The feedforward section itself consists of a transversal filter whose taps are spaced at the reciprocal of the symbol rate. The data sequence to be equalized is applied to the input of this section. The feedback section consists of another transversal filter whose taps are also spaced at the reciprocal of the symbol rate. The input applied to the feedback section is made up of decisions on previously detected symbols. The function of the feedback section is to subtract out that portion of intersymbol interference produced by previously detected symbols from the estimates of future symbols. This

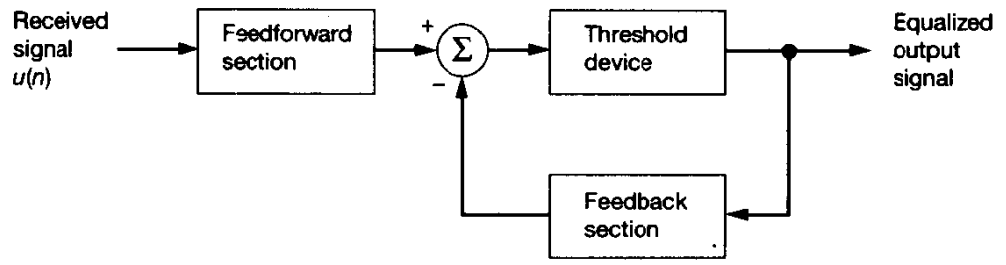


Figure 35 Block diagram of decision-feedback equalizer.

cancelation is an old idea known as the *bootstrap technique*. A decision-feedback equalizer yields good performance in the presence of severe intersymbol interference as experienced in fading radio channels, for example. The first report on decision-feedback equalization was published by Austin (1967), and the optimization of the decision-feedback receiver for minimum mean-squared error was first accomplished by Monsen (1971).

Coding of speech. In 1966, Saito and Itakura used a *maximum likelihood* approach for the application of prediction to speech. A standard assumption in the application of the maximum likelihood principle is that the input process is Gaussian. Under this condition, the exact application of the maximum likelihood principle yields a set of nonlinear equations for the parameters of the predictor. To overcome this difficulty, Itakura and Saito utilized approximations based on the assumption that the number of available data points greatly exceeds the prediction order. The use of this assumption makes the result obtained from the maximum likelihood principle assume an approximate form that is the same as the *autocorrelation method* of linear prediction. The application of the maximum likelihood principle is justified on the assumption that speech is a stationary Gaussian process, which seems reasonable in the case of unvoiced sounds.

In 1970, Atal presented the first use of the term "linear prediction" for speech analysis. Details of this new approach, linear predictive coding (LPC), to speech analysis and synthesis were published by Atal and Hanauer in 1971, in which the speech waveform is represented directly in terms of time-varying parameters related to the transfer function of the vocal tract and the characteristics of the excitation. The predictor coefficients are determined by minimizing the mean-squared error, with the error defined as the difference between the actual and predicted values of the speech samples. In the work by Atal and Hanauer, the speech wave was sampled at 10 kHz and then analyzed by predicting the present speech sample as a linear combination of the 12 previous samples. Thus 15 parameters [the 12 parameters of the predictor, the pitch period, a binary parameter indicating whether the speech is voiced or unvoiced, and the root-mean-square (rms) value of the speech samples] were used to describe the speech analyzer. For the speech synthesizer, an all-pole filter was used, with a sequence of quasi-periodic pulses or a white-noise source providing the excitation.

Another significant contribution to the linear prediction of speech was made in 1972 by Itakura and Saito; they used partial correlation techniques to develop a new structure,

the lattice, for formulating the linear prediction problem.[18] The parameters that characterize the lattice predictor are called *reflection coefficients* or *partial correlation (PARCOR) coefficients*, depending on the algebraic sign used in the definition. Although by that time the essence of the lattice structure had been considered by several other investigators, the invention of the lattice predictor is credited to Saito and Itakura. In 1973, Wakita showed that the filtering actions of the lattice predictor model and an acoustic tube model of speech are identical, with the reflection coefficients in the acoustic tube model as common factors. This discovery made possible the extraction of the reflection coefficients by the use of a lattice predictor.

Early designs of a lattice predictor were based on a *block processing* approach (Burg, 1967). In 1981, Makhoul and Cossell used an *adaptive* approach for designing the lattice predictor for applications in speech analysis and synthesis. They showed that the convergence of the adaptive lattice predictor is fast enough for its performance to equal that of the optimal (but more expensive) adaptive autocorrelation method.

This historical review on speech coding relates to LPC vocoders. We next present a historical review of the adaptive predictive coding of speech, starting with ordinary pulse-code modulation (PCM).

PCM was invented in 1937 by Reeves (1975). This was followed by the invention of differential pulse-code modulation (DPCM) by Cutler (1952). The early use of DPCM for the predictive coding of speech signals was limited to linear predictors with *fixed* parameters (McDonald, 1966). However, due to the nonstationary nature of speech signals, a fixed predictor cannot predict the signal values efficiently at all times. In order to respond to the nonstationary characteristics of speech signals, the predictor has to be adaptive (Atal and Schroeder, 1967). In 1970, Atal and Schroeder described a sophisticated scheme for adaptive predictive coding of speech. The scheme recognizes that there are two main causes of redundancy in speech (Schroeder, 1966): (1) quasi-periodicity during voiced segments, and (2) lack of flatness of the short-time spectral envelope. Thus, the predictor is designed to remove signal redundancy in two stages. The first stage of the predictor removes the quasi-periodic nature of the signal. The second stage removes formant information from the spectral envelope. The scheme achieves dramatic reductions in bit rate at the expense of a significant increase in circuit complexity. Atal and Schroeder (1970) report that the scheme can transmit speech at 10 kb/s, which is several times less than the bit rate required for logarithmic-PCM encoding with comparable speech quality.

Spectrum analysis. At the turn of the twentieth century, Schuster introduced the *periodogram* for analyzing the power spectrum[19] of a time series (Schuster, 1898). The periodogram is defined as the squared amplitude of the discrete Fourier transform of the time series. The periodogram was originally used by Schuster to detect and estimate the amplitude of a sine wave of known frequency that is buried in noise. Until the work of

[18]According to Markel and Gray (1976), the work of Itakura and Saito in Japan on the PARCOR formulation of linear prediction had been presented in 1969.

[19]For a fascinating historical account of the concept of power spectrum, its origin and its estimation, see Robinson (1982).

Yule in 1927, the periodogram was the only numerical method available for spectrum analysis. However, the periodogram suffers from the limitation that when it is applied to empirical time series observed in nature the results obtained are very erratic. This led Yule to introduce a new approach based on the concept of a *finite parameter model* for a stationary stochastic process in his investigation of the periodicities in time series with special reference to Wolfer's sunspot number (Yule, 1927). Yule, in effect, created a stochastic feedback model in which the present sample value of the time series is assumed to consist of a linear combination of past sample values plus an error term. This model is called an autoregressive model in that a sample of the time series regresses on its own past values, and the method of spectrum analysis based on such a model is accordingly called autoregressive spectrum analysis. The name "autoregressive" was coined by Wold in his doctoral thesis (Wold, 1938).

Interest in the autoregressive method was reinitiated by Burg (1967, 1975). Burg introduced the term *maximum-entropy method* to describe an algorithmic approach for estimating the power spectrum directly from the available time series. The idea behind the maximum-entropy method is to extrapolate the autocorrelation function of the time series in such a way that the *entropy* of the corresponding probability density function is maximized at each step of the extrapolation. In 1971, Van den Bos showed that the maximum-entropy method is equivalent to least-squares fitting of an autoregressive model to the known autocorrelation sequence.

Another important contribution made to the literature on spectrum analysis is that by Thomson (1982). His *method of multiple windows*, based on the prolate spheroidal wave functions, represents a nonparametric method for spectrum estimation that overcomes many of the limitations of the above-mentioned techniques.

Adaptive Noise Cancelation. The initial work on adaptive echo cancelers started around 1965. It appears that Kelly of Bell Telephone Laboratories was the first to propose the use of an adaptive filter for echo cancelation, with the speech signal itself utilized in performing the adaptation; Kelly's contribution is recognized in the paper by Sondhi (1967). This invention and its refinement are described in the patents by Kelly and Logan (1970) and Sondhi (1970).

The adaptive line enhancer was originated by Widrow and his co-workers at Stanford University. An early version of this device was built in 1965 to cancel 60-Hz interference at the output of an electrocardiographic amplifier and recorder. This work is described in the paper by Widrow et al. (1975b). The adaptive line enhancer and its application as an adaptive detector are patented by McCool et al. (1980).

The adaptive echo canceler and the adaptive linear enhancer, although intended for different applications, may be viewed as examples of the *adaptive noise canceler* discussed by Widrow et al. (1975). This scheme operates on the outputs of two sensors: a *primary sensor* that supplies a desired signal of interest buried in noise, and a *reference sensor* that supplies noise alone, as illustrated in Fig. 24. It is assumed that (1) the signal and noise at the output of the primary sensor are uncorrelated, and (2) the noise at the output of the reference sensor is correlated with the noise component of the primary sensor output.

The adaptive noise canceler consists of an adaptive filter that operates on the reference sensor output to produce an *estimate* of the noise, which is subtracted from the primary sensor output. The overall output of the canceler is used to control the adjustments applied to the tap weights in the adaptive filter. The adaptive canceler tends to minimize the mean-square value of the overall output, thereby causing the output to be the best estimate of the desired signal in the minimum-mean-square error sense.

Adaptive beamforming. The development of adaptive beamforming technology may be traced back to the invention of the *intermediate frequency (IF) sidelobe canceler* by Howells in the late 1950s. In a paper published in the 1976 Special Issue of the IEEE Transactions on Antennas and Propagation, Howells describes his personal observations on early work on adaptive antennas at the General Electric and Syracuse University Research Corporation (Howells, 1976). According to this historic report, Howells had developed by mid-1957 a sidelobe canceler capable of automatically nulling out the effect of one jammer. The sidelobe canceler uses a *primary* (high-gain) antenna and a *reference omni-directional* (low-gain) antenna to form a two-element array with one degree of freedom that makes it possible to steer a deep null anywhere in the sidelobe region of the combined antenna pattern. In particular, a null is placed in the direction of the jammer, with only a minor perturbation of the main lobe. Subsequently, Howells (1965) patented the sidelobe canceler.

The second major contribution to adaptive array antennas was made by Applebaum in 1966. In a classic report, he derived the *control law* governing the operation of an adaptive array antenna, with a control loop for each element of the array (Applebaum, 1966). The algorithm derived by Applebaum was based on maximizing the signal-to-noise ratio (SNR) at the array antenna output for any type of noise environment. Applebaum's theory included the sidelobe canceler as a special case. His 1966 classic report was reprinted in the 1976 Special Issue of IEEE Transactions on Antennas and Propagation.

Another algorithm for the weight adjustment in adaptive array antennas was advanced independently in 1967 by Widrow and his co-workers at Stanford University. They based their theory on the simple and yet effective LMS algorithm. The 1967 paper by Widrow et al. was not only the first publication in the open literature on adaptive array antenna systems, but also it is considered to be another classic of that era.

It is noteworthy that the maximum SNR algorithm (used by Applebaum) and the LMS algorithm (used by Widrow and his co-workers) for adaptive array antennas are rather similar. Both algorithms derive the control law for adaptive adjustment of the weights in the array antenna by sensing the correlation between element signals. Indeed, they both converge toward the optimum Wiener solution for stationary inputs (Gabriel, 1976).

A different method for solving the adaptive beamforming problem was proposed by Capon (1969). Capon realized that the poor performance of the delay-and-sum beamformer is due to the fact that its response along a direction of interest depends not only on the power of the incoming target signal but also undesirable contributions received from other sources of interference. To overcome this limitation of the delay-and-sum beam-

former, Capon proposed a new beamformer in which the weight vector $\mathbf{w}(n)$ is chosen so as to *minimize the variance* (i.e., average power) of the beamformer output, subject to the constraint $\mathbf{w}^H(n)\mathbf{s}(\phi) = 1$ for all n, where $\mathbf{s}(\phi)$ is a prescribed *steering vector*. This constrained minimization yields an adaptive beamformer with *minimum-variance distortionless response (MVDR)*.

In 1983, McWhirter proposed a simplification of the Gentleman–Kung (systolic) array for recursive least-squares estimation. The resulting filtering structure, often referred to as the *McWhirter (systolic) array*, is particularly well suited for adaptive beamforming applications.

The historical notes presented in this last section of the chapter on adaptive filter theory and applications are not claimed to be complete. Rather, they are intended to highlight many of the significant contributions made to this important part of the ever-expanding field of signal processing. Above all, it is hoped that they provide a source of inspiration to the reader.