Adaptive Filter Theory

Second Edition

SIMON HAYKIN
McMaster University

PRENTICE HALL
Englewood Cliffs, NJ 07632

20 Blind Deconvolution

Deconvolution is a signal processing operation that ideally unravels the effects of convolution. Specifically, the objective of deconvolution is to recover the input signal applied to a linear time-invariant (possibly nonminimum phase) system, given the signal developed at the output of the system. As the name implies, "blind deconvolution" refers to the ability of an adaptive algorithm to perform deconvolution in a *blindfolded* or *self-recovered fashion*. An adaptive filtering algorithm designed in this way does *not* need an external source for supplying the desired response. Rather, the algorithm makes up for it by using some form of *nonlinearity* to extract useful information from the unknown system output, which is unprocessed by a conventional linear adaptive filter.

In this chapter, we study two important families of blind deconvolution algorithms:

1. *Bussgang algorithms*, which perform blind deconvolution in an *iterative* fashion; they are so-called because the deconvolved sequence assumes Bussgang statistics when the algorithm reaches convergence in the mean.
2. *Polyspectra-based algorithms*, which use *higher-order cumulants* or their discrete Fourier transforms known as *polyspectra;* the property of polyspectra to preserve phase information makes them well suited for blind deconvolution.

We begin our study of blind deconvolution by discussing its theoretical requirements and practical importance, which we do in the next section.

20.1 THEORETICAL AND PRACTICAL CONSIDERATIONS

Consider an *unknown* linear time-invariant system $\mathscr{S}$ with input $\{x(n)\}$ as depicted in Fig. 20.1. The input consists of an *unobserved* white data (information-bearing) sequence with known probability distribution. *The problem is to restore* $\{x(n)\}$ *or*

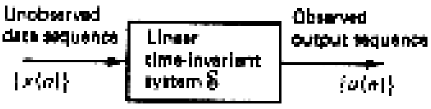

Figure 20.1 Setting the stage for blind deconvolution.

equivalently, to identify the inverse $\mathcal{S}^{-1}$ of the system $\mathcal{S}$, given the observed sequence $\{u(n)\}$ at the system output.

If the system $\mathcal{S}$ is *minimum-phase* (i.e., the transfer function of the system has all of its poles and zeros confined to the interior of the unit circle in the z-plane), then not only is the system $\mathcal{S}$ stable but so is the inverse system $\mathcal{S}^{-1}$. In this case, we may view the input sequence $\{(x(n)\}$ as the "innovation" of the system output $\{u(n)\}$, and the inverse system $\mathcal{S}^{-1}$ is just a *whitening filter*. These observations follow from the study of linear prediction presented in Chapter 6.

In many practical situations, however, the system $\mathcal{S}$ may *not* be minimum phase. A system is said to be *nonminimum phase* if its transfer function has any of its zeros located outside the unit circle in the z-plane; exponential stability of the system dictates that the poles be located inside the unit circle. Practical examples of a nonminimum phase system include a telephone channel and a fading radio channel. In this situation, the restoration of the input sequence $\{(x(n)\}$, given the channel output, is a more difficult problem. In particular, we may make the following important observations (Benveniste et al., 1980):

1. For the estimation of a nonminimum-phase characteristic to be feasible, and therefore for the inverse-filtering (i.e., deconvolution) problem to have a solution, the input sequence $\{x(n)\}$ must be *non-Gaussian*.
2. Since the use of a minimum mean-squared error criterion results in a linear filter that is minimum phase, we must invoke the use of *higher-order statistics* (cumulants or moments) and therefore *nonlinear estimation*.
3. The inverse system $\mathcal{S}^{-1}$ is *unstable*, because its transfer function has poles outside the unit circle in the z-plane. Hence, the *on-line* restoration of the input sequence $\{x(n)\}$ is impossible. However, we may *truncate* the impulse response of the inverse system $\mathcal{S}^{-1}$, thereby permitting the restoration of $\{x(n)\}$ to take place in real time, but with a constant and finite delay; this delay is of no serious concern in digital communications.

It is thus apparent that for the blind deconvolution problem to have a solution, two requirements must be satisfied. First, the unobserved data sequence $\{x(n)\}$ must be non-Gaussian. Second, the processing of the observed output sequence $\{u(n)\}$ must include some form of nonlinear estimation. Practical applications of blind deconvolution include adaptive equalization and seismic deconvolution, as discussed next.

Typically, adaptive equalizers used in digital communications require an initial *training period*, during which a *known* data sequence is transmitted. A replica of this sequence is made available at the receiver in proper synchronism with the transmitter, thereby making it possible for adjustments to be made to the equalizer coefficients in accordance with the adaptive filtering algorithm employed in the equalizer design. When the training is completed, the equalizer is switched to its *decision-directed mode*, and normal data transmission may then commence. (These two modes of operation of an adaptive equalizer were discussed in Section 1.7.) However, there are practical situations where it would be highly desirable for a receiver to be able to achieve complete adaptation *without* the cooperation of the transmitter. For example, in a *multipoint data network* involving a *control unit* connected to several *data terminal equipments* (DTEs), we have a "master–slave" situation in that a DTE is permitted to transmit only when its modem is polled by the modem of the control unit. A problem peculiar to these networks is that of retraining the receiver of a DTE unable to recognize data and polling messages, due to severe variations in channel characteristics or simply because that particular receiver was not powered-on during initial synchronization of the network. Clearly, in a large or heavily loaded multipoint network, data throughput is increased and the burden of monitoring the network is eased if some form of *blind equalization* is built into the receiver design (Godard, 1980).

Another class of communication systems that may need blind equalization is high capacity line-of-sight *digital radio*. Most of the time, a line-of-sight microwave radio link behaves as a wide-band, low-noise channel capable of providing highly reliable, high-speed data transmission. The channel, however, suffers from anomalous propagation conditions that arise from natural phenomena, which can cause the error performance of a digital radio to be severely degraded. Such anomalies manifest themselves by causing the transmitted signal to propagate along several paths, each of different electrical length. This phenomenon is called *multipath fading*. Conventional linear adaptive equalization techniques (e.g., the LMS algorithm) perform satisfactorily in a digital radio system, except during severe multipath fading that is the main cause of outage in such systems. Blind equalization provides a possible mechanism for dealing with the severe multipath fading problem in digital radio links [Beneviste and Goursat (1984); Ross (1989)].

In reflection seismology, the traditional method of removing the source waveform from a seismogram is to use *linear-predictive deconvolution* (see Section 1.7). The method of predictive deconvolution is derived from four fundamental assumptions (Gray, 1979):

1. The reflectivity series is *white*. This assumption is, however, often violated by reflection seismograms as the reflectivities result from a differential process applied to acoustic impedances. In many sedimentary basins there are thin beds that cause the reflectivity series to be correlated in sign.
2. The source signal (wavelet) is *minimum phase*. This assumption is valid for several explosive sources (e.g., dynamite), but it is only approximate for more complicated sources such as those used in marine exploration.
3. The reflectivity series and noise are statistically independent and stationary in time. The stationarity assumption, however, is violated because of spherical divergence and attenuation of seismic waves. To cope with nonstationarity of the data, we may use adaptive deconvolution, but such a method often destroys primary events of interest.
4. The *minimum mean-square error criterion* is used to solve the linear prediction problem. This criterion is appropriate only when the prediction errors (the reflectivity series and noise) have a Gaussian distribution. Statistical tests per-

formed on reflectivity series, however, show that their kurtosis is much higher than that expected from a Gaussian distribution.

Assumptions 1 and 2 were explicitly mentioned in the presentation of the method of predictive deconvolution in Chapter 1. Assumptions 3 and 4 are implicit in the application of Wiener filtering that is basic to the solution of the linear prediction problem, as presented in Chapter 6. The main point of this discussion is that valuable phase information contained in a reflection seismogram is ignored by the method of predictive deconvolution. This limitation is overcome by using *blind deconvolution* (Godfrey and Rocca, 1981).

Blind equalization in digital communications and blind deconvolution in reflection seismology are examples of a special kind of adaptive inverse filtering that operate *without* access to the source of the desired response (i.e., the transmitted signal). Only the received signal and some additional information in the form of a *probabilistic model* are needed. In the case of equalization for digital communications, the model describes the statistics of the transmitted data sequence. In the case of seismic deconvolution, the model describes the statistics of the earth's reflection coefficients.

Having clarified the framework within which the use of blind deconvolution is feasible, we are ready to undertake a detailed study of its operation. Specifically, we consider the Bussgang family and the polyspectra-based family of blind deconvolution algorithms. The discussion is presented in that order, and in the context of blind equalization for digital communications.

0.2 BUSSGANG ALGORITHM FOR BLIND EQUALIZATION OF REAL BASEBAND CHANNELS

Consider the *baseband model* of a digital communication system, depicted in Fig. 20.2. The model consists of the cascade connection of a *linear communication channel* and a *blind equalizer*.

The channel includes the combined effects of a transmit filter, a transmission medium, and a receive filter. It is characterized by an impulse response $\{h_n\}$ that is *unknown*; it may be time varying, albeit slowly. The nature of the impulse response $\{h_n\}$ (i.e., whether it is real or complex valued) is determined by the type of modulation employed. To simplify the discussion, we assume that the impulse response is real, which corresponds to the use of *multilevel pulse-amplitude modulation* (*M-ary PAM*); the case of a complex impulse response is considered in the next section. We may thus describe the input-output relation of the channel by the *convolution sum*

$$u(n) = \sum_k h_k x(n-k), \qquad n = 0, \pm 1, \pm 2, \dots \tag{20.1}$$

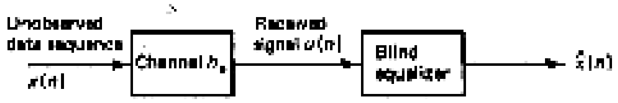

Figure 20.2 Cascade connection of an unknown channel and blind equalizer.

where $\{x(n)\}$ is the *data (message) sequence* applied to the channel input, and $\{u(n)\}$ is the resulting *channel output*. We assume that

$$\sum_k h_k^2 = 1 \tag{20.2}$$

Equation (20.2) implies the use of *automatic gain control* (AGC) that keeps the variance of the channel output $u(n)$ constant. We further assume that the channel is nonminimum phase (i.e., noncausal), which means that

$$h_n \neq 0, \qquad \text{for } n < 0 \tag{20.3}$$

In the mathematical model of Eq. (20.1), we have ignored the effect of receiver noise. We are justified to do so, because the degradation in the performance of data transmission (over a voice-grade telephone channel, say) is usually dominated by *intersymbol interference* due to channel dispersion.

The problem we wish to solve is the following:

Given the received signal $\{u(n)\}$, reconstruct the original data sequence $\{x(n)\}$ applied to the channel input.

Equivalently, we may restate the problem as follows:

Design a blind equalizer that is the inverse of the unknown channel, with the channel input being unobservable.

To solve this blind equalization problem, we need to prescribe a *probabilistic model* for the data sequence $\{x(n)\}$. For the problem at hand, we assume the following (Bellini, 1986, 1988):

1. The data sequence $\{x(n)]$ is *white*; that is, the data symbols $x(n)$ are *independent, identically distributed* (iid) *random variables*, with zero mean and unit variance, as shown by

$$E[x(n)] = 0 \tag{20.4}$$

and

$$E[x(n)x(k)] = \begin{cases} 1, & k = n \\ 0, & k \neq n \end{cases} \tag{20.5}$$

where E is the expectation operator.

2. The *probability density function* of the data symbol $x(n)$ is *symmetric* and *uniform;* that is (see Fig. 20.3),

$$f_X(x) = \begin{cases} 1/2\sqrt{3}, & -\sqrt{3} \leq x < \sqrt{3} \\ 0 & \text{otherwise} \end{cases} \tag{20.6}$$

This distribution has the merit of being independent of the number M of amplitude levels employed in the modulation process.

Note that Eq. (20.4) and the first line of Eq. (20.5) follow from (20.6).

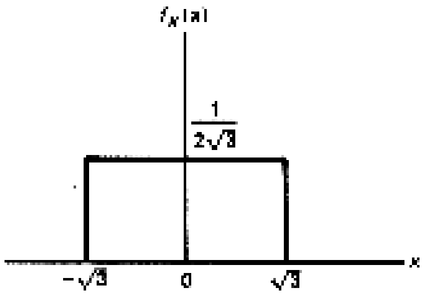

Figure 20.3 Uniform distribution.

With the distribution of the datum $x(n)$ assumed to be symmetric, as in Fig. 20.3, we find that the whole data sequence $\{-x(n)\}$ has the same law as $\{x(n)\}$. Hence, we cannot distinguish the desired inverse filter $\mathcal{H}^{-1}$ (corresponding to $\{x(n)\}$ from the opposite one $-\mathcal{H}^{-1}$ (corresponding to $\{-x(n)\}$). We may overcome this *sign ambiguity problem* by *initializing* the deconvolution algorithm such that there is a single nonzero tap weight with the desired algebraic sign (Benveniste et al., 1980).

Iterative Deconvolution: The Objective

Let $\{w_i\}$ denote the impulse response of the *ideal inverse filter*, which is related to the impulse response $\{h_i\}$ of the channel as follows:

$$\sum_i w_i h_{l-i} = \delta_l \tag{20.7}$$

where δ_l is the *Kronecker delta:*

$$\delta_l = \begin{cases} 1, & l = 0 \\ 0, & l \neq 0 \end{cases} \tag{20.8}$$

An inverse filter defined in this way is "ideal" in the sense that it reconstructs the transmitted data sequence $\{x(n)\}$ *correctly*. To demonstrate this, we first write

$$\sum_i w_i u(n-i) = \sum_i \sum_k w_i h_k x(n-i-k) \tag{20.9}$$

Let

$$k = l - i$$

Making this change of indices in Eq. (20.9), and interchanging the order of summation, we get

$$\sum_i w_i u(n-i) = \sum_l x(n-l) \sum_i w_i h_{l-i} \tag{20.10}$$

Hence, using Eq. (20.7) in (20.10) and then applying the definition of Eq. (20.8), we get

$$\begin{aligned} \sum_i w_i u(n-i) &= \sum_l \delta_l x(n-l) \\ &= x(n) \end{aligned} \tag{20.11}$$

which is the desired result.

For the situation described herein, the impulse response $\{h_n\}$ is unknown. We therefore cannot use Eq. (20.7) to determine the inverse filter. Instead, we use an *iterative deconvolution procedure* to compute an *approximate inverse filter* characterized by the impulse response $\{\hat{w}_i(n)\}$. The index i refers to the *tap-weight number* in the *transversal filter* realization of the approximate inverse filter, as indicated in Fig. 20.4. The index n refers to the *iteration number*; each iteration corresponds to the transmission of a data symbol. The computation is performed iteratively in such a way that the convolution of the impulse response $\{\hat{w}_i(n)\}$ with the received signal $\{u(i)\}$ results in the complete or partial removal of the intersymbol interference (Godfrey and Rocca, 1981). Thus, at the nth iteration we have an approximately deconvolved sequence

$$y(n) = \sum_{i=-L}^{L} \hat{w}_i(n)u(n-i) \tag{20.12}$$

where $2L + 1$ is the truncated *length* of the impulse response $\{\hat{w}_i(n)\}$ (see Fig. 20.4).

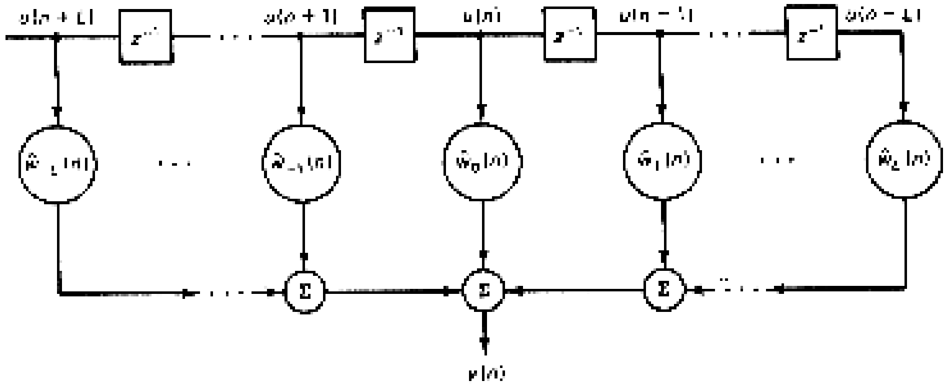

Figure 20.4 Transversal filter realization of approximate inverse filter; use of real data is assumed.

The convolution sum on the left side of Eq. (20.11), pertaining to the ideal inverse filter, is *infinite* in extent in that the index i ranges from $-\infty$ to ∞. On the other hand, the convolution sum on the right side of Eq. (20.12) pertaining to the approximate inverse filter, is *finite* in extent in that i extends from $-L$ to L. Clearly, we may rewrite Eq. (20.12) as follows:

$$y(n) = \sum_i \hat{w}_i(n)u(n-i), \qquad \hat{w}_i(n) = 0 \text{ for } |i| > L$$

or, equivalently,

$$y(n) = \sum_i w_i u(n-i) + \sum_i [\hat{w}_i(n) - w_i]u(n-i) \tag{20.13}$$

Let

$$\nu(n) = \sum_i [\hat{w}_i(n) - w_i]u(n-i), \qquad \hat{w}_i = 0 \text{ for } |i| > L \tag{20.14}$$

Then, using the ideal result of Eq. (20.11) and the definition of Eq. (20.14), we may simplify Eq. (20.13) as follows:

$$y(n) = x(n) + \nu(n) \tag{20.15}$$

The term $\nu(n)$ is called the *convolutional noise*, representing the *residual* intersymbol interference that results from the use of an approximate inverse filter.

The inverse filter output $y(n)$ is next applied to a *zero-memory nonlinear estimator*, producing the estimate $\hat{x}(n)$ for the datum $x(n)$. This operation is depicted in the block diagram of Fig. 20.5. We may thus write

$$\hat{x}(n) = g(y(n)) \tag{20.16}$$

where $g(\cdot)$ is some nonlinear function. The issue of nonlinear estimation is discussed in the next subsection.

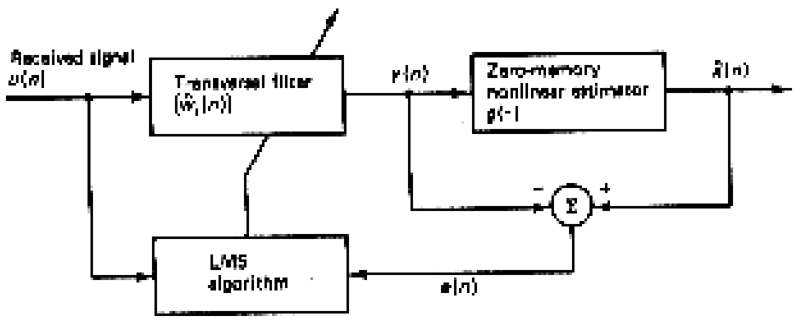
Received signal $u(n)$
Transversal filter $\{\hat{w}_i(n)\}$
$y(n)$
Zero-memory nonlinear estimator $g(\cdot)$
$\hat{x}(n)$
$-$
Σ
$+$
LMS algorithm
$e(n)$

Figure 20.5 Block diagram of blind equalizer.

Ordinarily, we find that the estimate $\hat{x}(n)$ is not reliable enough. Nevertheless, we may use it in an *adaptive* scheme to obtain a "better" estimate at iteration $n + 1$. Indeed, we have a variety of *linear* adaptive filtering algorithms (discussed in previous chapters) at our disposal that we can use to perform this adaptive parameter estimation. In particular, a simple and yet effective scheme is provided by the LMS algorithm. To apply it to the problem at hand, we note the following:

1. The ith tap input of the transversal filter at iteration (time) n is $u(n - i)$.
2. Viewing the nonlinear estimate $\hat{x}(n)$ as the "desired" response [since the transmitted data symbol $x(n)$ is unavailable to us], and recognizing that the corresponding transversal filter output is $y(n)$, we may express the *estimation error* for the iterative deconvolution procedure as

$$e(n) = \hat{x}(n) - y(n) \tag{20.17}$$

3. The ith tap weight $\hat{w}_i(n)$ at iteration n represents the "old" parameter estimate.

Accordingly, the "updated" value of the ith tap weight at iteration $n + 1$ is computed as follows:

$$\hat{w}_i(n + 1) = \hat{w}_i(n) + \mu u(n - i)e(n), \qquad i = 0, \pm 1, \ldots, \pm L \tag{20.18}$$

Let

$$n - i - k = l$$

Hence, we may also write

$$v(n) = \sum_l x(l)\nabla(n - l) \tag{20.22}$$

where

$$\nabla(n) = \sum_k h_k[\hat{w}_{n-k}(n) - w_{n-k}] \tag{20.23}$$

The sequence $\{\nabla(n)\}$ is a sequence of small numbers, corresponding to the *residual impulse response* of the channel. We imagine the sequence $\{\nabla(n)\}$ as a *long and oscillatory wave* that is convolved with the transmitted data sequence $\{x(n)\}$ to produce the convolutional noise sequence $\{v(n)\}$, as indicated in Eq. (20.22).

The definition of Eq. (20.22) is basic to the statistical characterization of the convolution noise $v(n)$. The mean of $v(n)$ is zero, as shown by

$$\begin{aligned} E[v(n)] &= E\left[\sum_l x(l)\nabla(n - l)\right] \\ &= \sum_l \nabla(n - l)E[x(l)] \\ &= 0 \end{aligned} \tag{20.24}$$

where in the last line we have made use of Eq. (20.4). Next, the autocorrelation function of $v(n)$ for a lag j is given by

$$\begin{aligned} E[v(n)v(n - j)] &= E\left[\sum_l x(l)\nabla(n - l)\sum_m x(m)\nabla(n - m - j)\right] \\ &= \sum_l \sum_m \nabla(n - l)\nabla(n - m - j)E[x(l)x(m)] \\ &= \sum_l \nabla(n - l)\nabla(n - l - j) \end{aligned} \tag{20.25}$$

where in the last line we have made use of Eq. (20.5). Since $\{\nabla(n)\}$ is a long and oscillatory waveform, the sum on the right side of Eq. (20.25) is nonzero only for $j = 0$, obtaining

$$E[v(n)v(n - j)] = \begin{cases} \sigma^2, & j = 0 \\ 0, & j \neq 0 \end{cases} \tag{20.26}$$

where

$$\sigma^2(n) = \sum_l \nabla^2(n - l) \tag{20.27}$$

Based on Eqs. (20.24) and (20.26), we may thus describe the convolutional noise process $\{v(n)\}$ as a *zero-mean white-noise process of time-varying variance equal* to $\sigma^2(n)$, defined by Eq. (20.27).

According to the model of Eq. (20.22), the convolutional noise $\nu(n)$ is a weighted sum of statistically independent and identically distributed random variables representing different transmissions of data symbols. If, therefore, the residual impulse response $\{\nabla(n)\}$ is long enough, the *central limit theorem* makes the *Gaussian* model for $\nu(n)$ plausible.

Having characterized the convolutional noise $\nu(n)$ by itself, all that remains for us to do is to evaluate the *cross-correlation* between it and the data sample $x(n)$. These two samples are certainly *correlated* with each other, since $\{\nu(n)\}$ is the result of convolving the residual impulse response $\{\nabla(n)\}$ with $\{x(n)\}$, as shown in Eq. (20.22). However, the cross-correlation between $\nu(n)$ and $x(n)$ is negligible compared to the variance of $\nu(n)$. To demonstrate this, we write

$$\begin{aligned} E[x(n)\nu(n - j)] &= E[x(n) \sum_l x(l)\nabla(n - l - j)] \\ &= \sum_l \nabla(n - l - j)E[x(n)x(l)] \qquad (20.28) \\ &= \nabla(-j) \end{aligned}$$

where, in the last line, we have made use of Eq. (20.5). Here again, using the assumption that $\{\nabla(n)\}$ is a long and oscillatory waveform, we deduce that the standard deviation of $\nu(n)$ is large compared to the magnitude of the cross-correlation $E[x(n)\nu(n - j)]$.

Since the data sequence $\{x(n)\}$ is white by assumption and the convolutional noise sequence $\{\nu(n)\}$ is approximately white by deduction, and since these two sequences are essentially uncorrelated, it follows that their sum $\{y(n)\}$ is approximately white too. This suggests that $\{x(n)\}$ and $\{\nu(n)\}$ may be taken to be essentially independent. We may thus model the convolutional noise $\{\nu(n)\}$ as an *additive, zero-mean, white Gaussian noise process that is statistically independent of the data sequence* $\{x(n)\}$.

Because of the approximations made in deriving the model described herein for the convolutional noise, its use in an iterative deconvolution process yields a *suboptimal* estimator for the data sequence. In particular, given that the iterative deconvolution process is convergent, the intersymbol interference (ISI) during the latter stages of the process may be small enough for the model to be applicable. In the early stages of the iterative deconvolution process, however, the ISI is typically large with the result that the data sequence and the convolutional noise are strongly correlated, and the convolutional noise sequence is more uniform than Gaussian [Godfrey and Rocca (1981)].

Zero-Memory Nonlinear Estimation of the Data Sequence

We are now ready to consider the next important issue, namely, that of estimating the data sequence $\{x(n)\}$, given the deconvolved sequence $\{y(n)\}$ at the transversal filter output. Specifically, we may formulate the estimation problem as follows: We are given a (filtered) observation $y(n)$ that consists of the sum of two components [see Eq. (20.15)]:

1. A uniformly distributed data symbol $x(n)$ with zero mean and unit variance
2. A white Gaussian noise $\nu(n)$ with zero mean and variance $\sigma^2(n)$, which is statistically independent of $x(n)$

The requirement is to derive a *Bayes estimate* of $x(n)$, optimized in some sense.

Before proceeding with this classical estimation problem, two noteworthy observations are in order. First, the estimate is naturally a *conditional estimate* that depends on the optimization criterion. Second, although the estimate (in theory) is optimum in a mean-square error sense, in the context of our present situation, it is *suboptimum* by virtue of the approximations made in the development of the model for the convolutional noise $\nu(n)$.

An optimization criterion of particular interest is that of minimizing the mean-square value of the error between the actual transmission $x(n)$ and the estimation $\hat{x}(n)$. The choice of this optimization criterion yields a *conditional mean estimator*[1] that is both sensible and robust.

For convenience of presentation, we will suppress the dependence of random variables on time n. Thus, given the observation y, the conditional mean estimate $\hat{x}$ of the random variable x is written as $E[\hat{x}|y]$, where E is the expectation operator (see Fig. 20.6). Let $f_X(x|y)$ denote the *conditional probability density function* of x, given y. We thus have

$$\begin{aligned}\hat{x} &= E[x|y] \\ &= \int_{-\infty}^{\infty} x f_X(x|y)\,dx\end{aligned} \tag{20.29}$$

From *Bayes' rule*, we have

$$f_X(x|y) = \frac{f_Y(y|x)f_X(x)}{f_Y(y)} \tag{20.30}$$

where $f_Y(y|x)$ is the conditional probability density function of y, given x; and $f_X(x)$ and $f_Y(y)$ are the probability density functions of x and y, respectively. We may therefore rewrite the formula of Eq. (20.29) as

$$\hat{x} = \frac{1}{f_Y(y)} \int_{-\infty}^{\infty} x f_Y(y|x) f_X(x)\,dx \tag{20.31}$$

Let

$$y = c_0 x + \nu \tag{20.32}$$

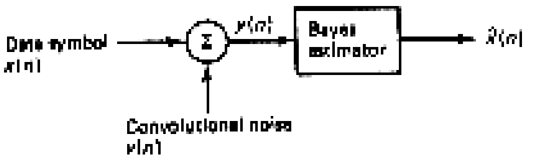

Figure 20.6 Estimation of the data symbol $x(n)$, given the observation $y(n)$.

[1] For a derivation of the conditional mean and its relation to mean-squared-error estimation, see Appendix E.

The scaling factor c_0 is slightly smaller than unity. This factor has been included in Eq. (20.32) so as to keep $E[y^2]$ equal to 1. In accordance with the statistical model for the convolutional noise υ developed previously, x and υ are statistically independent. With υ modeled to have zero mean and variance σ^2, it follows from Eq. (20.32) that the scaling factor c_0 is defined by

$$c_0 = \sqrt{1 - \sigma^2} \tag{20.33}$$

Furthermore, from Eq. (20.32) it follows that

$$f_Y(y|x) = f_V(y - c_0 x) \tag{20.34}$$

Accordingly, the use of Eq. (20.34) in (20.31) yields

$$\hat{x} = \frac{1}{f_Y(y)} \int_{-\infty}^{\infty} x f_V(y - c_0 x) f_X(x)\, dx \tag{20.35}$$

The evaluation of $\hat{x}$ is straightforward but tedious. To proceed with it, we may note the following:

1. The mathematical form of the estimate $\hat{x}(n)$ produced at the output of the Bayes (conditional mean) estimator depends on the probability density function of the original data symbol $x(n)$. For the analysis presented herein, we assume that the data symbol x is *uniformly distributed* with zero mean and unit variance; its probability density function is given in Eq. (20.6), which is reproduced here for convenience:

$$f_X(x) = \begin{cases} 1/2\sqrt{3} & -\sqrt{3} \le x < \sqrt{3} \\ 0, & \text{otherwise} \end{cases} \tag{20.36}$$

2. The convolutional noise υ is *Gaussian distributed* with zero mean and variance σ^2; its probability density function is

$$f_V(\upsilon) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\upsilon^2}{2\sigma^2}\right) \tag{20.37}$$

3. The filtered observation y is the sum of $c_0 x$ and υ; its probability density function is therefore equal to the convolution of the probability density function of x with that of υ, as shown by

$$f_Y(y) = \int_{-\infty}^{\infty} f_X(x) f_V(y - c_0 x)\, dx \tag{20.38}$$

Using Eqs. (20.36) to (20.38) in (20.35), we get (Bellini, 1988):

$$\hat{x} = \frac{1}{c_0} y + \frac{\sigma}{c_0} \frac{Z(y + c_0/\sqrt{3}) - Z(y - c_0/\sqrt{3})}{Q(y - c_0/\sqrt{3}) - Q(y + c_0/\sqrt{3})} \tag{20.39}$$

where $Z(y)$ is the *standardized* Gassian probability density function .

$$Z(y) = \frac{1}{\sqrt{2\pi}} e^{-y^2/2} \tag{20.40}$$

and $Q(y)$ is the corresponding probability distribution function

$$Q(y) = \frac{1}{\sqrt{2\pi}} \int_{y}^{\infty} e^{-u^2/2}\, du \tag{20.41}$$

For a detailed derivation of Eq. (20.39), the reader is referred to Problem 2.

A small *gain correction* to the nonlinear estimator of Eq. (20.39) is needed in order to achieve perfect equalization when the iterative deconvolution algorithm [described by Eqs. (20.16) to (20.18)] converges eventually. Perfect equalization² requires that $y = x$. Convergence of the algorithm in the mean is satisfied when the estimation error in the LMS algorithm is orthogonal to each of the tap inputs in the transversal filter realization of the approximate inverse filter. Putting all of this together, we find that the following condition must hold (Bellini, 1986, 1988):

$$E[\hat{x}g(\hat{x})] = 1 \tag{20.42}$$

where $g(\hat{x})$ is the nonlinear estimator $\hat{x} = g(y)$ with $y = \hat{x}$ (for perfect equalization (see Problem 3).

Figure 20.7 shows the nonlinear estimator $\hat{x} = g(y)$ for an eight-level PAM system (Bellini, 1986, 1988). The estimator is normalized in accordance with Eq.

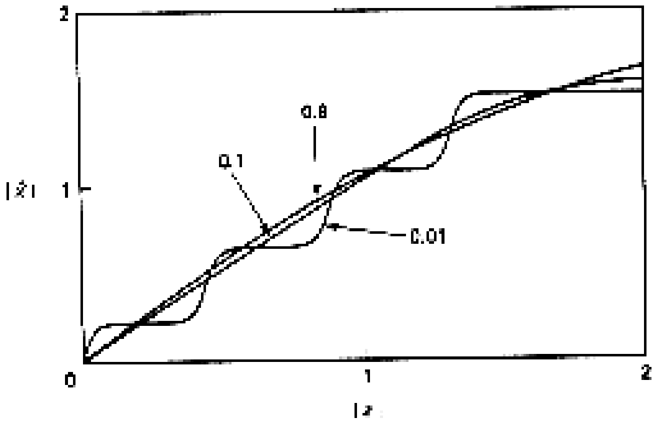

Figure 20.7 Nonlinear estimators of eight-level data in Gaussian noise. The noise-to-(signal + noise) ratios are 0.01, 0.1, and 0.8. [From Bellini (1986) with permission of IEEE.]

²In general, for perfect equalization we require that

$$y = (x - D)e^{j\phi}$$

where D is a constant delay and ϕ is a constant phase shift. This condition corresponds to an equalizer whose transfer function has magnitude one and a linear phase response. We note that the input data sequence $\{x_n\}$ is stationary and the channel is linear time-invariant. Hence, the observed sequence $\{y(n)\}$ at the channel output is also stationary; its probability density function is therefore invariant to the constant delay D. The constant phase shift ϕ is also of no immediate consequence when the probability density function of the input sequence remains symmetric under rotation, which is indeed the case for the assumed density function given in Eq. (20.36). We may therefore simplify the condition for perfect equalization by requiring that $y = x$.

(20.42). In this figure, three levels of convolutional noise are considered. Here we note from Eq. (20.32) that the *distortion-to (signal plus distortion) ratio* is given by

$$\begin{aligned}\frac{E[(y-x)^2]}{E[y^2]} &= (1-c_0)^2+\sigma^2 \\ &= 2(1-c_0)\end{aligned} \tag{20.43}$$

where in the last line we have made use of Eq. (20.33). The curves presented in Fig. 20.7 correspond to three values of this ratio, namely, 0.01, 0.1, and 0.8. We observe the following from these curves:

1. When the convolutional noise is low, the blind equalization algorithm tends to a minimum mean-squared-error criterion.
2. When the convolutional noise is high, the nonlinear estimator appears to be independent of the fine structure of the amplitude-modulated data. Indeed, different values of amplitude modulation levels result in only very small gain differences due to the normalization defined by Eq. (20.42). This suggests that the use of a uniform amplitude distribution for multilevel modulation systems is an adequate approximation.
3. The nonlinear estimator is robust with respect to variations in the variance of the convolutional noise.

It is of interest to note that for high levels of convolutional noise, the input-output characteristic of the nonlinear estimator is well approximated by the *sigmoid nonlinearity:*[3]

$$\begin{aligned}\hat{x} &= a\frac{1-e^{-by}}{1+e^{-by}} \\ &= a\tanh\left(\frac{by}{2}\right)\end{aligned} \tag{20.44}$$

For the situation described in Fig. 20.7, the following values for the constants a and b provide a good fit:

$$a = 1.945$$

$$b = 1.25$$

This is demonstrated in Fig. 20.8.

In any event, the input–output characteristic of the nonlinear estimator is computed in advance, and the result is stored in memory. Then, it is merely a matter of reading off the value of the estimate $\hat{x}$ corresponding to the value y obtained at the inverse filter output.

[3] A sigmoid nonlinearity is a common element in the design of neural networks. In particular, the combination of a linear combiner and a sigmoid nonlinearity constitutes a *neuron*. Accordingly, we may view the blind equalizer depicted in Fig. 20.5 as being essentially a single neuron with its linear combiner and sigmoid nonlinearity represented by the transversal filter and zero-memory nonlinear estimator, respectively. The error signal for controlling the weights at the input end of the neuron is obtained by comparing the input and output signals of the sigmoid nonlinearity.

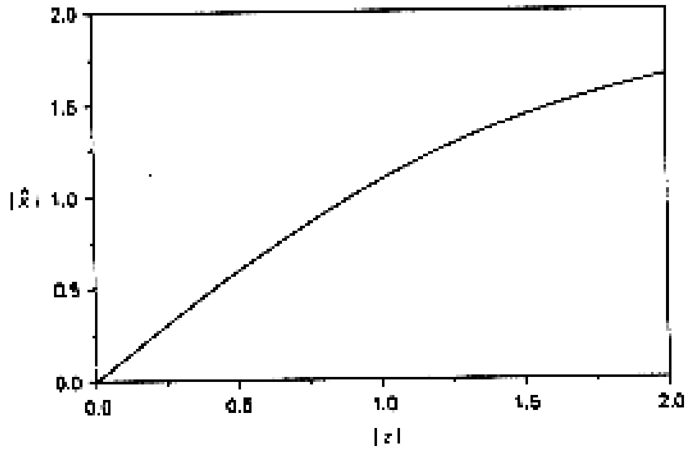

Figure 20.8 Input-output characteristic of sigmoid nonlinearity.

Convergence Considerations

For the iterative convolutional algorithm described by Eqs. (20.16) to (20.18) to converge in the mean, we require that the expected value of the tap weight $\hat{w}_i(n)$ approach some constant value as the number of iterations n approaches infinity. Correspondingly, from Eq. (20.18) we find that the *condition for convergence in the mean* is described by

$$E[u(n-i)y(n)] = E[u(n-i)g(y(n))], \qquad \text{large } n, \quad i = 0, \pm 1, \ldots, \pm L$$

Multiplying both sides of this equation by $\hat{w}_{i-k}$ and summing over i, we get

$$E\left[y(n) \sum_{i=-L}^{L} \hat{w}_{i-k}(n) u(n-i)\right] = E\left[g(y(n)) \sum_{i=-L}^{L} \hat{w}_{i-k}(n) u(n-i)\right], \qquad \text{large } n \tag{20.45}$$

We next note from Eq. (20.12) that

$$\begin{aligned} y(n-k) &= \sum_{l=-L}^{L} \hat{w}_l(n) u(n-k-l) \\ &= \sum_{i=-L-k}^{L-k} \hat{w}_{i-k}(n) u(n-i) \qquad \text{large } n \end{aligned}$$

Provided that L is large enough for the transversal equalizer to achieve perfect equalization, we may approximate the expression for $y(n-k)$ as

$$y(n-k) \approx \sum_{i=-L}^{L} \hat{w}_{i-k}(n) u(n-i) \qquad \text{large } n \text{ and large } L \tag{20.46}$$

Accordingly, we may use Eq. (20.46) to simplify (20.45) as follows

$$E[y(n)y(n-k)] \approx E[g(y(n))y(n-k)] \qquad \text{large } n \text{ and large } L$$

Equivalently, we may write

$$E[y(n)y(n+k)] \simeq E[y(n)g(y(n+k))] \qquad \text{large } n \text{ and large } L \qquad (20.47)$$

We now recognize the following property. A stochastic process $\{y(n)\}$ is said to be a *Bussgang process* if it satisfies the condition

$$E[y(n)y(n+k)] = E[y(n)g(y(n+k))] \qquad (20.48)$$

where the function $g(\cdot)$ is a zero-memory nonlinearity.[4] In other words, a Bussgang process has the property that its autocorrelation function is equal to the cross-correlation between that process and the output of a zero-memory nonlinearity produced by that process, with both correlations being measured for the same lag.

Returning to the issue at hand, we may state that the process $\{y(n)\}$ acting as the input to the zero-memory nonlinearity in Fig. 20.5 is *approximately* a Bussgang process, provided that L is large; the approximation becomes better as L is made larger. It is for this reason that the blind equalization algorithm described herein is referred to as a *Bussgang algorithm* [Bellini (1986, 1988)].

In general, convergence of the Bussgang algorithm is not guaranteed. Indeed, the cost function of the Bussgang algorithm operating with a finite L is *nonconvex*; it may therefore have false minima.

For the idealized case of an *infinite length* equalizer, however, a rough proof of convergence of the Bussgang algorithm may be sketched as follows [Bellini (1988)]. The proof relies on a theorem derived in Benveniste et al. (1980), which provides sufficient conditions for convergence.[5] Let the function $\psi(y)$ denote the dependence of the estimation error in the LMS algorithm on the transversal filter output $y(n)$. According to our terminology, we have[6] [see Eqs. (20.16) and (20.17)]

$$\psi(y) = g(y) - y \qquad (20.49)$$

The *Benveniste–Goursat–Ruget theorem* states that convergence of the Bussgang algorithm is guaranteed if the probability distribution of the data sequence $\{x(n)\}$ is *sub-Gaussian* and the second derivative of $\psi(y)$ is negative on the interval $(0, \infty)$. In particular, we may state the following:

1. A random variable x is said to be sub-Gaussian if its probability density function is

$$f_X(x) = Ke^{-|x/\beta|^\nu} \qquad (20.50)$$

where $\nu > 2$. For the limiting case of $\nu = \infty$, the probability density function of Eq. (20.50) reduces to that of a uniformly distributed random variable.

[4] A number of stochastic processes belong to the class of Bussgang processes. Bussgang (1952) was the first to recognize that any correlated Gaussian process has the property described in Eq. (20.48). Subsequently, Barrett and Lampard (1955) extended Bussgang's result to all stochastic processes with exponentially decaying autocorrelation functions. This includes an independent process, since its autocorrelation function consists of a delta function that may be viewed as an infinitely fast decaying exponential [Gray (1979)].

[5] The theorem of interest in Benveniste et al. (1980) is Theorem 3.6 and the accompanying Lemma 3.4.

[6] Note that the function $\psi(y)$ defined in Eq. (20.48) is the negative of that defined in Benveniste et al. (1980).

Also, by choosing $\beta = \sqrt{3}$, we have $E[x^2] = 1$. Thus, the probabilistic model assumed in Eq. (20.6) satisfies the first part of the Benveniste–Goursat–Ruget theorem.

2. The second part of the theorem is also satisfied by the Bussgang algorithm, since we have

$$\frac{\partial^2 \psi}{\partial y^2} < 0, \qquad \text{for } 0 < y < \infty \tag{20.51}$$

This is readily verified by examining the curves plotted in Fig. 20.7 or the sigmoid nonlinearity defined in Eq. (20.44).

The Benveniste–Goursat–Ruget theorem exploited in this proof is based on the assumption of an infinitely parameterized equalizer (i.e., L is infinitely large). Unfortunately, this assumption breaks down in practice as we have to work with a finite L. To date, no zero-memory nonlinear function $g(\cdot)$ is known, which would result in global convergence of the blind equalizer in Fig. 20.5 to the inverse of the unknown channel [Verdu (1984); Johnson (1991)]. The global convergence of the Bussgang algorithm for an arbitrarily large but finite L remains an open problem.

Decision-Directed Algorithm

When the Bussgang algorithm has converged and the eye pattern appears "open," the equalizer should be switched smoothly to the *decision-directed mode* of operation, and minimum mean-squared-error control of the tap weights of the transversal filter component in the equalizer is exercised, as in a conventional adaptive equalizer.

Figure 20.9 presents a block diagram of the equalizer operating in its decision-directed mode. The only difference between this mode of operation and that of blind equalization lies in the type of zero-memory nonlinearity employed. Specifically, the conditional mean estimation of the blind equalizer in Fig. 20.9 is replaced by a *threshold decision device*. Given the observation $y(n)$, that is, the equalized signal at the transversal filter output, the threshold device makes a *decision in favor of a particular value in the known alphabet of the transmitted data sequence that is closest to* $y(n)$. We may thus write

$$\hat{x}(n) = \text{dec}(y(n)) \tag{20.52}$$

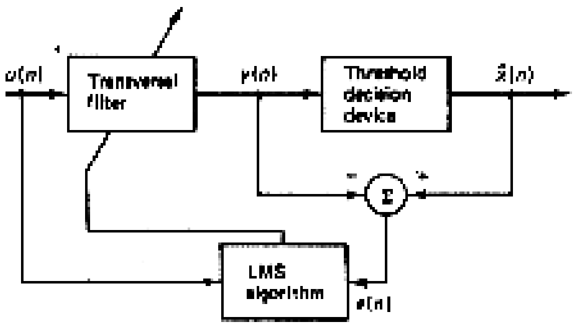

Figure 20.9 Block diagram of decision-directed mode of operation.

For example, in the simple case of a *binary equiprobable data sequence*, the data levels and decision levels are as follows, respectively,

$$x(n) = \begin{cases} +1, & \text{for symbol 1} \\ -1 & \text{for symbol 0} \end{cases} \tag{20.53}$$

and

$$\text{dec}((y(n)) = \text{sgn}(y(n)) \tag{20.54}$$

where sgn(·) is the *signum function* equal to $+1$ if the argument is positive, and -1 if it is negative.

The equations that govern the operation of the decision-directed algorithm are the same as those of the Bussgang algorithm, except for the use of Eq. (20.52) in place of (20.16). Herein lies an important practical advantage of a blind equalizer that is based on the Bussgang algorithm and incorporates the decision-directed algorithm: Its implementation is only slightly more complex than that of a conventional adaptive equalizer, yet it does not require the use of a training sequence.

Suppose that the following conditions are satisfied:

1. The eye pattern is open (which it should be as a result of the completion of blind equalization).
2. The step-size parameter μ used in the LMS implementation of the decision-directed algorithm is fixed (which is a common practice).
3. The sequence of observations at the channel output, denoted by the vector $\mathbf{u}(n)$, is *ergodic* in the sense that

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} \mathbf{u}(n)\mathbf{u}^T(n) \to E[\mathbf{u}(n)\mathbf{u}^T(n)] \quad \text{almost surely} \tag{20.55}$$

Then, under these conditions, *the tap-weight vector in the decision-directed algorithm converges to the optimum (Wiener) solution in the mean-square sense* (Macchi and Eweda, 1984). This is a powerful result, making the decision-directed algorithm an important adjunct of the Bussgang algorithm for blind equalization in digital communications.

0.3 EXTENSION OF BUSSGANG ALGORITHMS TO COMPLEX BASEBAND CHANNELS

Thus far we have only discussed the use of Bussgang algorithms for the blind equalization of M-ary PAM systems, characterized by a real baseband channel. In this section we extend the use of this family of blind equalization algorithms to *quadrature-amplitude modulation* (QAM) systems that involve a hybrid combination of amplitude and phase modulations.

In the case of a complex baseband channel, the transmitted data sequence $\{x(n)\}$, the channel impulse response $\{h_n\}$, and the received signal $\{u(n)\}$ are all complex valued. We may thus write

$$x(n) = x_I(n) + jx_Q(n) \tag{20.56}$$

$$h_n = h_{I,n} + jh_{Q,n} \tag{20.57}$$

and

$$u(n) = u_I(n) + ju_Q(n) \tag{20.58}$$

where the subscripts I and Q refer to the *in-phase* (*real*) and *quadrature* (*imaginary*) *components*, respectively. Correspondingly, the conditional mean estimate of the complex datum $x(n)$, given the observation $y(n)$ at the transversal filter output, is written as

$$\begin{aligned}\hat{x}(n) &= E[x(n)|y(n)] \\ &= \hat{x}_I(n) + j\hat{x}_Q(n) \\ &= g(y_I(n)) + jg(y_Q(n))\end{aligned} \tag{20.59}$$

where $g(\cdot)$ describes a zero-memory nonlinearity. Equation (20.59) states that the in-phase and quadrature components of the transmitted data sequence $\{x(n)\}$ may be estimated separately from the in-phase and quadrature components of the transversal filter output $y(n)$, respectively. Note, however, that the conditional mean $E[x(n)|y(n)]$ can only be expressed as in Eq. (20.59) if the data transmitted in the in-phase and quadrature channels are statistically independent of each other, which is usually the case.

Clearly, Bussgang algorithms for complex baseband channels include the corresponding algorithms for real baseband channels as a special case. Table 20.1 presents a summary of Bussgang algorithms for a complex baseband channel.

TABLE 20.1 SUMMARY OF BUSSGANG ALGORITHMS FOR BLIND EQUALIZATION OF COMPLEX BASEBAND CHANNELS

Initialization: Set

$$\hat{w}_i(0) = \begin{cases}1, & i = 0 \\ 0, & i = \pm 1, \ldots, \pm L\end{cases}$$

Computation: $n = 1, 2, \ldots$

$$\begin{aligned}y(n) &= y_I(n) + jy_Q(n) \\ &= \sum_{i=-L}^{L} \hat{w}_i^*(n)u(n-i) \\ \hat{x}(n) &= \hat{x}_I(n) + j\hat{x}_Q(n) \\ &= g(y_I(n)) + jg(y_Q(n)) \\ e(n) &= \hat{x}(n) - y(n) \\ \hat{w}_i(n+1) &= \hat{w}_i(n) + \mu u(n-i)e^*(n), \qquad i = 0, \pm 1, \ldots, \pm L\end{aligned}$$

SPECIAL CASES OF THE BUSSGANG ALGORITHM

Sato Algorithm

The idea of blind equalization in M-ary PAM systems dates back to the pioneering work of Sato (1975). The *Sato algorithm* consists of minimizing a *nonconvex* cost function

$$J(n) = E[(\hat{x}(n) - y(n))^2] \tag{20.60}$$

where $y(n)$ is the transversal filter output defined in Eq. (20.12), and $\hat{x}(n)$ is an estimate of the transmitted datum $x(n)$. This estimate is obtained by a zero-memory nonlinearity described as follows:

$$\hat{x}(n) = \gamma \operatorname{sgn}[y(n)] \tag{20.61}$$

The constant γ sets the *gain* of the equalizer; it is defined by

$$\gamma = \frac{E[x^2(n)]}{E[|x(n)|]} \tag{20.62}$$

The Sato algorithm for blind equalization was introduced originally to deal with one-dimensional multilevel (M-ary PAM) signals. Initially, the algorithm treats such a digital signal as a "binary" signal by estimating the most significant bit; the remaining bits of the signal are treated by the algorithm as additive noise insofar as the blind equalization process is concerned. The algorithm then uses the results of this preliminary step to modify the error signal obtained from a conventional decision-directed algorithm. The Sato algorithm is robust and superior to the decision-directed algorithm, but its rate of convergence is slower.

The Benveniste–Goursat–Ruget theorem for convergence holds for the Sato algorithm even though the nonlinear function $\psi(\cdot)$ defined in Eq. (20.49) is not differentiable. According to this theorem, global convergence of the Sato algorithm can be achieved provided that the probability density function of the transmitted data sequence can be approximated by a sub-Gaussian function such as the uniform distribution [Benveniste et al. (1980)]. However, this result has been disputed in Zing et al. (1989). Also, Mazo (1980) has reported indications of poor performance of the Sato algorithm. Nevertheless, it has been established that almost always the Sato algorithm converges to the correct equalizer coefficients once the eye pattern has been opened [Kumar (1983); Macchi and Eweda (1984)].

It is apparent that the Sato algorithm is a special (non-optimal) case of the Bussgang algorithm, with the nonlinear function $g(y)$ defined by

$$g(y) = \gamma \operatorname{sgn}(y) \tag{20.63}$$

where $\operatorname{sgn}(\cdot)$ is the signum function. The nonlinearity defined in Eq. (20.63) is similar to that in the decision-directed algorithm except for the data-dependent gain factor γ.

Godard Algorithm

Godard (1980) was the first to propose a family of *constant modulus* blind equalization algorithms for use in two-dimensional digital communication systems (e.g., M-ary phase-shift keying). Specifically, the *Godard algorithm* minimizes a nonconvex cost function of the form

$$J(n) = E[(|y(n)|^p - R_p)^2] \tag{20.64}$$

where p is a positive integer, and R_p is a positive real constant defined by

$$R_p = \frac{E[|x(n)|^{2p}]}{E[|x(n)|^p]} \tag{20.65}$$

The Godard algorithm is designed to penalize deviations of the blind equalizer output $x(n)$ from a constant modulus. The constant R_p is chosen in such a way that the gradient of the cost function $J(n)$ is zero when perfect equalization [i.e., $\hat{x}(n) = x(n)$] is attained

The tap-weight vector of the equalizer is adapted in accordance with the stochastic gradient algorithm [Godard (1980)]

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \mu \mathbf{u}(n) e^*(n) \tag{20.66}$$

where μ is the step-size parameter, $\mathbf{u}(n)$ is the tap-input vector, and $e(n)$ is the error signal defined by

$$e(n) = y(n)|y(n)|^{p-2}(R_p - |y(n)|^p) \tag{20.67}$$

From the definition of the cost function $J(n)$ in Eq. (20.64) and from (20.67), we see that the equalizer adaptation according to the Godard algorithm does not require carrier phase recovery. The algorithm therefore tends to converge slowly. However, it offers the advantage of decoupling the ISI equalization and carrier phase recovery problems from each other.

Two cases of the Godard algorithm are of specific interest:

Case 1: $p = 1$
The cost function of Eq. (20.64) for this case reduces to

$$J(n) = E[(|y(n)| - R_1)^2] \tag{20.68}$$

where

$$R_1 = \frac{E[|x(n)|^2]}{E[|x(n)|]} \tag{20.69}$$

This case may be viewed as a modification of the Sato algorithm.

Case 2: $p = 2$
In this case, the cost function of Eq. (20.64) reduces to

$$J(n) = E[(|y(n)|^2 - R_2)^2] \tag{20.70}$$

where

$$R_2 = \frac{E[|x(n)|^4]}{E[|x(n)|^2]} \tag{20.71}$$

This second special case is referred to in the literature as the *constant modulus algorithm* (CMA).[7]

There are conflicting reports in the literature on the convergence behavior of the Godard algorithm. In his original paper, Godard (1980) showed that the algorithm would converge to the global minimum and thereby achieve perfect equalization (i.e., zero ISI), provided that the algorithm is initialized in a special manner;

[7] The constant modulus algorithm (CMA) was so named by Treichler and Agee (1983), independently of Godard's 1980 paper. It is probably the most widely investigated blind equalization algorithm and the most widely used one in practice [Treichler and Larimore (1985a, b); Smith and Friedlander (1985); Johnson et al. (1988)].

otherwise, it is possible for the algorithm to converge to a local minimum. In a subsequent study, Foschini (1985) made the assertion that the cost function used in the Godard algorithm does not exhibit any locally stable undesirable equilibrium points. Yet, other investigators [Ding et al. (1989); Verdu (1984)] have demonstrated that under ideal modeling assumptions it is possible for the Godard algorithm to exhibit ill convergence due to the existence of local but (false) minima.

Summary of Special Forms of the Bussgang Algorithm

The decision-directed, Sato, and Godard algorithms may indeed be viewed as special cases of the Bussgang algorithm [Bellini (1986)]. In particular, we may use Eqs. (20.52), (20.61), and (20.67) to set up the entries shown in Table 20.2 for the special forms of the zero-memory nonlinear function $g(\cdot)$ pertaining to these three algorithms [Hatzinakos (1990)]. The entries for the decision-directed and Sato algorithms follow directly from the definition

$$\hat{x}(n) = g(y(n))$$

In the case of the Godard algorithm, we note that

$$e(n) = \hat{x}(n) - y(n)$$

or, equivalently,

$$g(y(n)) = y(n) + e(n)$$

Hence, we may use this relation and Eq. (20.67) to derive the entry for the Godard algorithm in Table 20.2.

TABLE 20.2 SPECIAL CASES OF THE BUSSGANG ALGORITHM

Algorithm	Zero-memory nonlinear function $g(\cdot)$	Definitions
Decision-directed	$\text{sgn}(\cdot)$	
Sato	$\gamma\,\text{sgn}(\cdot)$	$\gamma = \dfrac{E[x^2(n)]}{E[\lvert x(n)\rvert]}$
Godard	$\dfrac{y(n)}{\lvert y(n)\rvert}(\lvert y(n)\rvert + R_p\lvert y(n)\rvert^{p-1} - \lvert y(n)\rvert^{2p-1})$	$R_p = \dfrac{E[\lvert x(n)\rvert^{2p}]}{E[\lvert x(n)\rvert^p]}$