USC-SIPI REPORT #202

Adaptive Blind Equalization

by

Yuanjie Chen and Chrysostomos L. Nikias

May 1991

Signal and Image Processing Institute
UNIVERSITY OF SOUTHERN CALIFORNIA
Department of Electrical Engineering-Systems
3740 McClintock Avenue, Room 404
Los Angeles, CA 90089-2564 U.S.A.

ABSTRACT

This tutorial paper is focused on two topics, namely: (i) to describe systematic methodologies for selecting nonlinear transformations for blind equalization algorithms (and thus new types of cumulants), and (ii) to give an overview of the existing blind equalization algorithms and point out their strengths as well as weaknesses. It is shown in this paper that all blind equalization algorithms belong in one of the following three categories, depending where the nonlinear transformation is being applied on the data: (i) the Bussgang algorithms, where the nonlinearity is in the output of the adaptive equalization filter; (ii) the polyspectra (or Higher-Order Spectra) algorithms, where the nonlinearity is in the input of the adaptive equalization filter; and (iii) the algorithms where the nonlinearity is inside the adaptive filter, *i.e.*, the nonlinear filter or neural network. We describe methodologies for selecting nonlinear transformations based on various optimality criteria such as MSE or MAP. We illustrate that such existing algorithms as Sato, Benveniste-Goursat, Godard or CMA, Stop-and-Go and Donoho are indeed special cases of the Bussgang family of techniques when the nonlinearity is memoryless. We present results that demonstrate the polyspectra-based algorithms exhibit faster convergence rate than Bussgang algorithms. However, this improved performance is at the expense of more computations per iteration. We also show that blind equalizers based on nonlinear filters or neural networks are more suited for channels that have nonlinear distortions.

The Godard or CMA algorithm is probably the most widely used blind equalizer in digital communications today due to its simplicity, low complexity and constant modulus property. Its main drawbacks, however, are slow convergence and no guarantee for global convergence starting from arbitrary initial guess. We present a new method for blind equalization, the CRIMNO algorithm (*i.e.*, criterion with memory nonlinearity), which is shown to have the same advantages as Godard (simplicity, low complexity, constant modulus property) and yet guaranteeing much faster convergence. The CRIMNO algorithm is flexible enough to address blind deconvolution problems when the input sequence is colored.

1 INTRODUCTION

Blind deconvolution or equalization is a signal processing procedure that recovers the input sequence applied to a linear time-invariant nonminimum phase system from its output only. Blind equalization algorithms are essentially adaptive filtering algorithms designed in such a way that they do not need the external supply of a desired response to generate the error signal in the output of the adaptive filter. In other words, the adaptive algorithm is "blind" to the desired response. However, the algorithm itself generates the desired response by applying a nonlinear transformation on sequences involved in the adaptation process. All blind equalization algorithms belong to one of the following three categories, depending where the nonlinear transformation is being applied on the data:

- The Bussgang algorithms, where the nonlinearity is in the **output** of the adaptive equalization filter;
- The Polyspectra (or Higher-Order Spectra) algorithms, where the nonlinearity is in the **input** of the adaptive equalization filter;
- The algorithms where the nonlinearity is **inside** the adaptive filter; *i.e.*, the filter is nonlinear (*e.g.* Volterra) or neural network.

The purpose of this paper is to provide an overview of the existing blind equalization algorithms and to discuss their advantages and limitations. Conventional equalization and carrier recovery techniques used in multilevel digital communication systems usually require an initial training period, during which a known data sequence (*i.e.*, training sequence) is transmitted [43], [45]. An alternative effective approach to this problem is to utilize blind equalizers which do not require any known training sequence during the startup period.

The paper describes systematic methodologies for selecting the nonlinearity based on various optimality criteria, such as maximum likelihood (ML), mean-square error (MSE) or maximum a posteriori (MAP). As an example, it is illustrated that such existing algorithms as Sato [46], [47] Benveniste-Goursat [5], [6] Godard or CMA [22], [50] and Stop-and-Go [41] are indeed special cases of the family of Bussgang techniques where the nonlinearity is memoryless [3], [4]. It is demonstrated that the polyspectra-based algorithms exhibit faster convergence rate than the Bussgang algorithms. However, this improved performance is at the expense of more computational complexity. On the other hand, blind equalizers based on nonlinear filters are well suited for channels that have nonlinear distortions [39], [40].

The Godard algorithm is probably the most widely used blind equalizer in digital communications today due to its simplicity, low computational complexity, and constant modulus property. Its main drawbacks, however, is slow convergence and no guarantee for global convergence (convergence starting from arbitrary initial guess). The paper describes the development of the CRIMNO algorithm (*i.e.*, criterion with memory nonlinearity) which is shown to have the same advantages as Godard algorithm (simplicity, low complexity, constant modulus property) and yet guaranteeing much faster convergence [12], [13]. Extension of the CRIMNO algorithm to the case of colored input signals is also presented.

The polyspectra-based adaptive blind equalization algorithms are also described in the paper. In particular, the Tricepstrum Equalization Algorithm (TEA) [24], the Power Cepstrum and Tricoherence Equalization Algorithm (POTEA) [7], and the Cross-Tricepstrum Equalization Algorithm (CTEA) [8] are presented, as well as their advantages and limitations. It is shown that these algorithms perform simultaneous identification and equalization of a nonminimum phase communication channel from its output only. Simulations with PAM and QAM signals

demonstrate the effectiveness of the polyspectra-based algorithms.

Finally, the paper provides an overview of the neural network based adaptive equalization algorithms either with or without a training sequence [11], [20], [26], [27], [39], [40], [49].

2 DEFINITION OF BLIND EQUALIZATION PROBLEM

Let us consider the discrete-time linear transmission channel whose impulse response $\{f(i)\}$ is unknown and possibly time-varying. The input data $\{x(i)\}$ are assumed to be independent and identically distributed (i.i.d.) random variables, with non-Gaussian probability density function. Let us also assume, without loss of generality, that the sequence $\{x(i)\}$ has mean $E\{x(i)\} = 0$ and variance $E\{|x(i)|^2\} = Q_x$. If $x(i)$ is real, we may drop the magnitude function and simply write $E\{x^2(i)\}$. Initially, noise is not taken into account in the output of the channel. From Figure 2.1, it follows that the model we consider is

$$\begin{aligned} y(i) &= f(i) * x(i) \\ &= \sum_k f(k)\, x(i-k) \end{aligned} \tag{2.1}$$

where "$*$" denotes linear convolution and $\{y(i)\}$ is the received sequence. The problem is to reconstruct (or restore) the input sequence $\{x(i)\}$ from the received sequence $\{y(i)\}$ or, equivalently, to identify the inverse filter (equalizer) $\{u(i)\}$ for the channel.

From Figure 2.1, we see that the output sequence $\{\tilde{x}(i)\}$ of the equalizer is given by

$$\tilde{x}(i) = u(i) * y(i)$$

$$= u(i) * (f(i) * x(i))$$

$$= u(i) * f(i) * x(i). \tag{2.2}$$

So, to achieve

$$\tilde{x}(i) = x(i-D)e^{j\theta} \tag{2.3}$$

where D is a constant delay and θ is a constant phase shift, it is required that

$$u(i) * f(i) = \delta(i-D) \tag{2.4}$$

where

$$\delta(i) = \begin{cases} (1)(e^{j\theta}), & i = 0 \\ 0, & \text{otherwise.} \end{cases}$$

Performing the Fourier transform on (2.4), we obtain

$$U(w) \cdot F(w) = e^{j(\theta - wD)}. \tag{2.5}$$

In other words, the objective of the equalizer is to achieve a transfer function

$$U(w) = \frac{1}{F(w)} e^{j(\theta - wD)}. \tag{2.6}$$

In general, D and θ are unknown. However, the constant delay D does not affect the reconstruction of the original input sequence $\{x(i)\}$. The constant phase shift θ can be removed by a carry recovery technique. As such, in the sequel, it will be assumed that $D = 0$ and $\theta = 0$.

Blind equalization schemes may be classified into three categories; *i.e.*, those which utilize

nonlinearities in the **output** of the adaptive equalization filter, those which place the nonlinearity in the **input** of the adaptive equalization filter, and those which utilize adaptive nonlinear equalization filters. The Bussgang equalization algorithms with memoryless or memory nonlinearity belong to the first category whereas the higher-order cumulant-based equalizers (TEA, POTEA, etc.) belong to the second category, as they perform **memory** nonlinear transformation on the input data of the equalization filter. Blind equalizers based on nonlinear filters, such as the Volterra filter or neural networks, belong to the third category. Figures 2.2 (a)-(c) illustrate the block diagrams of the aforementioned three families of blind equalizers.

3 PERFORMANCE MEASURES FOR ALGORITHM EVALUATION

Four different performance measures are usually considered in simulation experiments for the testing of the blind equalization algorithms: the time-average squared error ($\mathrm{E_{ASE}}$), the transitional symbol error rate (SER), the residual intersymbol interference (ISI) and the discrete eye patterns [43], [44]. They are defined as follows.

Time-Average Squared Error($\mathrm{E_{ASE}}$ or MSE)

At iteration (i), the mean square error in the output of the equalizer is defined as :

$$\mathrm{E_{ASE}} = \frac{1}{N}\sum_{i=1}^{N}|x(i-D)-\tilde{x}(i)|^2 \tag{3.1}$$

where $\tilde{x}(i)$ is the output of the equalizer at iteration (i) and $x(i-D)$ is the corresponding true value. Note that the delay D, which is introduced by the channel and the equalizer, does not affect the recovery of the original information $\{x(i)\}$. However, it must be taken into account in

the calculation of MSE (i). The MSE (i) gives a measure of both the noise and residual ISI at the output of the equalizer.

Transitional Symbol Error Rate (SER)

The SER indicates the percentage of wrongly detected symbols in consecutive intervals of 500 symbols, *i.e.*,

$$\text{SER} = \frac{\#\text{of wrong detections in 500 symbols}}{500} \tag{3.2}$$

Residual ISI

The residual ISI in the output of equalizer is defined as follows. Let $\{f(i)\}$ be the channel impulse response and $\{u(i)\}$ the equalizer tap coefficients at iteration (i). Let $s(i) = f(i) * u(i)$, then

$$\text{ISI}(i) = \frac{\sum_i |s(i)|^2 - \max\{|s(i)|^2\}}{\max\{|s(i)|^2\}} \tag{3.3}$$

Physically, this indicates the amount of ISI present at the output of the equalizer due to imperfect equalization.

Discrete eye patterns

Discrete eye patterns (or equalized signal constellation) consist of all possible values of the output of the equalizer, $\tilde{x}(i)$, at iteration (i), **drawn in two-dimensional space.** We say that the eye pattern is open whenever the ideal decoding thresholds are easily distinguishable between neighboring equalized states.

In our simulations, all performance measures were calculated for many independent signal and noise realizations. For the E_{ASE}, time averaging over 100 samples were performed for each realization. The eye pattern at iteration (i) was obtained by drawing the output of equalizer for all

independent realizations and for a specific number of samples (for each realization) symmetrically located around (i).

4 ALGORITHMS WITH NONLINEARITY IN THE OUTPUT OF THE EQUALIZATION FILTER

Let us assume that a guess for the impulse response of the inverse filter (equalizer), $u_g(i)$ has been selected. Then,

$$u_g(i) * f(i) = \delta(i) + \epsilon(i) \tag{4.1}$$

where $\epsilon(i)$ accounts for the difference (error) between our guess $u_g(i)$ and the actual values of $u(i)$. If we convolve the initial guess of the inverse filter, $\{u_g(i)\}$, with the received sequence, $\{y(i)\}$, we obtain

$$\begin{aligned} \tilde{x}(i) &= y(i) * u_g(i) \\ &= x(i) * f(i) * u_g(i). \end{aligned} \tag{4.2}$$

Combining (4.2) with(4.1), we obtain

$$\begin{aligned} \tilde{x} &= x(i) * (\delta(i) + \epsilon(i)) \\ &= [x(i) * \delta(i)] + [x(i) * \epsilon(i)] \\ &= x(i) + n(i) \end{aligned} \tag{4.3}$$

where

$$n(i) = x(i) * \epsilon(i) \tag{4.4}$$

is the "convolutional noise", namely, the residual ISI arising from the difference between our guess $u_g(i)$ and the actual inverse filter $u(i)$.

Our problem now is to utilize the deconvolved sequence $\{\tilde{x}(i)\}$ to find the "best" estimate of $\{\tilde{x}(i)\}$; namely, $\{d(i)\}$. Note that in adaptive-filter literature $d(i)$ is used to represent the desired response [25]. Two criteria are employed to determine the "best" estimate of $x(i)$ from the given $\tilde{x}(i)$. These are the mean-square error (MSE) and maximum a posteriori (MAP).

Since the transmitted sequence $x(i)$ has a non-Gaussian probability density function, the MSE and MAP estimates are nonlinear transformations of $\tilde{x}(i)$. In general, the "best" estimate $d(i)$ is given by [3], [4], [23], [54].

$$d(i) = g[\tilde{x}(i)] \qquad \text{(memoryless)}$$

or

$$d(i) = g[\tilde{x}(i), \tilde{x}(i-1), \ldots, \tilde{x}(i-m)] \qquad (m\text{th} - \text{order memory}) \tag{4.5}$$

where $g[\cdot]$ is a nonlinear function with or without memory. The $d(i)$ is fed back into the adaptive equalization filter as shown in Figure 4.1. From this figure, it is also apparent that the nonlinear function $g[\cdot]$ is in the **output** of the equalization filter.

4.1 Optimum Selection of Nonlinearities

4.1.1 Nonlinearities with MSE Estimates

In summary, a well treated classical estimation problem is as follows:

$$\tilde{x}(i) = x(i) + n(i) \tag{4.6}$$

where

(i) $n(i)$ is Gaussian. Note that if $\epsilon(i)$ in (4.4) is long enough, the central limit theorem makes the Gaussianity assumption for $n(i)$ reasonable.

(ii) $\{x(i)\}$ are independent, identically distributed (i.i.d.) and in general non-Gaussian. The *pdf* of $x(i)$ is known; in digital communications the $\{x(i)\}$ are usually equi-probable discrete signal points.

(iii) $x(i)$ and $n(i)$ are assumed independent.

Given the $\tilde{x}(i)$, we seek the MSE estimate of $x(i)$, namely, $d_{\text{mse}}(i)$.

From Van Trees [52, p. 58], it follows that the best MSE estimate of $\{x(i)\}$ given $\{\tilde{x}(i)\}$ is the **mean of the a posteriori density**, *i.e.*,

$$\begin{aligned} d_{\text{mse}}(i) &= \int_{-\infty}^{+\infty} dx\ x P_{x/\tilde{x}}(x/\tilde{x}) \\ &= E\{x(i)/\tilde{x}(i)\}. \end{aligned} \tag{4.7}$$

where $P_{x/\tilde{x}}(x/\tilde{x}) = \frac{P_{x/\tilde{x}}(x/\tilde{x}) \cdot P_x(x)}{P_{\tilde{x}}(\tilde{x})}$ is the a posteriori density; $P_{x/\tilde{x}}(x/\tilde{x})$ is Gaussian, $N(x(i), Q_n)$, with Q_n being the variance of $\{n(i)\}$; the a priori density $P_x(x)$ is the *pdf* of $x(i)$, and $P_{x/\tilde{x}}(\tilde{x})$ behaves as a normalization constant in the integral of (4.7).

If $x(i)$ is zero-mean Gaussian with variance Q_x; *i.e.*, $P_x(x)$ is $N(0, Q_x)$, (4.7) reduces to

$$d_{\text{mse}}(i) = \frac{Q_x}{Q_x + Q_n}\tilde{x}(i) \tag{4.8}$$

which, in turn, implies that $g[\tilde{x}(i)]$ is a **linear** function. However, when $P_x(x)$ is non-Gaussian, the integral (4.7) can not be reduced to a simple expression and $g[\cdot]$ will be a nonlinear function. In the sequel, we show $d_{\text{mse}}(i)$ versus $\tilde{x}(i)$ when *pdf* $P_x(x)$ is uniform and Laplace.

Uniform Distribution

The a priori *pdf* is given by

$$P_x(x) = \begin{cases} \frac{1}{2\lambda} & -\lambda \leq x \leq \lambda \\ 0, & \text{otherwise.} \end{cases} \tag{4.9}$$

Consequently, the a posteriori *pdf* takes the form

$$P_{x/\tilde{x}}(x/\tilde{x}) = \begin{cases} \frac{A_1(x,\tilde{x})}{B_1(\tilde{x})} & -\lambda \leq x \leq \lambda \\ 0, & \text{otherwise.} \end{cases} \tag{4.10}$$

where

$$A_1(x,\tilde{x}) = \frac{1}{2\lambda}\frac{1}{\sqrt{2\pi Q_n}}\exp\left[-\frac{(x-\tilde{x})^2}{2Q_n}\right]$$
$$B_1(\tilde{x}) = \int_{-l}^{\lambda} A_1(x)dx.$$

Substituting (4.10) into (4.7), we obtain $d_{\text{mse}}(i)$ as a function of $\tilde{x}$. However, this relationship is not easy to express analytically and is obtained by numerical integration as shown in Figure 4.2.

Laplace Distribution

The a priori density is given by

$$P_x(x) = \frac{\lambda}{2}\exp[-\lambda|x|] \tag{4.11}$$

and thus the a posteriori density takes the form

$$P_{x/\tilde{x}}(x/\tilde{x}) = \frac{A_2(x,\tilde{x})}{B_2(\tilde{x})} \tag{4.12}$$

where

$$\begin{aligned} A_2(x,\tilde{x}) &= \frac{\lambda}{2}\cdot\exp\left[-\lambda|x|\cdot\frac{1}{\sqrt{2\pi Q_n}}\exp[-\frac{(x-\tilde{x})^2}{2Q_n}\right] \\ B_2(\tilde{x}) &= \int_{-\infty}^{+\infty} A_2(x)dx. \end{aligned}$$

Combining (4.12) with (4.7) and using numerical integration we obtain d_{mse} vs $\tilde{x}$ as shown in Figure 4.3.

4.1.2 Nonlinearities with MAP Estimates

In this section we treat the estimation problem

$$\tilde{x}(i) = x(i) + n(i)$$

where $n(i)$ is Gaussian and $x(i)$ is *i.i.d.* non-Gaussian. However, we seek MAP estimate of $x(i)$, namely $d_{\text{map}}(i)$ when $n(i)$ is white or colored, or correlated with $x(i)$. The colored noise case,

as well as the case of correlated noise with $x(i)$, will result into a memory nonlinear relationship between d_{map} and $\tilde{x}(i)$; *i.e.*, $d_{\text{map}}(i) = g[\tilde{x}(i), \tilde{x}(i-1), \ldots, \tilde{x}(i-m)]$. If $x(i)$ is Gaussian *i.i.d.* and $n(i)$ is white Gaussian, independent from $x(i)$, then the $d_{\text{map}}(i)$ is identical to $d_{\text{mse}}(i)$ and is given by (4.8).

If we denote $\underline{x} = [x(i), x(i-1), \ldots, x(1)]$ and $\underline{\tilde{x}} = [\tilde{x}(i), \tilde{x}(i-1), \ldots, \tilde{x}(1)]$, then a posteriori *pdf* is given by Van Trees [p. 58]

$$P_{\underline{x}/\underline{\tilde{x}}}(\underline{x}/\underline{\tilde{x}}) = \frac{P_{\underline{x}}(\underline{x}) \cdot P_{\underline{\tilde{x}}/\underline{x}}(\underline{\tilde{x}}/\underline{x})}{P(\underline{\tilde{x}})} \tag{4.13}$$

and the MAP estimate, $\underline{d}_{\text{map}}$, of $\underline{x}$ given $\underline{\tilde{x}}$ is the value of $\underline{x}$ which maximizes $\ell(\underline{x})$, where

$$\ell(\underline{x}) = \ell n P_{\underline{\tilde{x}}/\underline{x}}(\underline{\tilde{x}}/\underline{x}) + \ell n P_{\underline{x}}(\underline{x}). \tag{4.14}$$

where the denominator of (4.13) does not contribute to the maximization of $\ell(\underline{x})$.

CASE I: White Gaussian Noise

In this case the $n(i)$ is white, Gaussian $N(0, Q_n)$, and independent of $x(i)$. It is also assumed that $\{x(i)\}$ are *i.i.d.* and non-Gaussian. Consequently, joint *pdfs* are expressed as products of marginal *pdfs* and the MAP estimate at each iteration $\{i\}$, $d_{\text{map}}(i)$, is obtained by maximizing

$$\ell(x(i)) = \ell n P_{\underline{\tilde{x}}/\underline{x}}(\underline{\tilde{x}}/\underline{x}) + \ell n P_{\underline{x}}(\underline{x}).$$

That is to say that the **estimation problem is decoupled** and the resulting relationship $d_{\text{map}}(i)$ vs $\tilde{x}(i)$, is **memoryless.**

The following memoryless nonlinearities can be derived.

(i) Uniform Distribution (4.9)

$$d_{\text{map}}(i) = \begin{cases} -\lambda, & \tilde{x}(i) < -\lambda \\ \tilde{x}(i), & -\lambda \leq \tilde{x}(i) \leq \lambda \\ \lambda, & \tilde{x}(i) > \lambda \end{cases} \tag{4.15}$$

Note that d_{map} does not depend on Q_n.

(ii) Laplace Distribution (4.11)

$$d_{\text{map}}(i) = \begin{cases} \tilde{x}(i) + \lambda Q_n, & \tilde{x}(i) < -\lambda Q_n \\ 0, & -\lambda Q_n \leq \tilde{x}(i) \leq \lambda Q_n \\ \tilde{x}(i) - \lambda Q_n, & \tilde{x}(i) > \lambda Q_n. \end{cases} \tag{4.16}$$

Here the MAP estimate depends on Q_n. For the symmetric uniform and Laplace a priori distributions the resulting a posteriori *pdf*, $P_{\underline{\tilde{x}}/\underline{x}}(\underline{\tilde{x}}/\underline{x})$, is asymmetric.

Figures 4.4 and 4.5 illustrate the MAP memoryless nonlinearities.

CASE II: Colored Gaussian Noise

In this case we assume that $n(i)$ is colored Gaussian $N(0, \underline{R})$ where $\underline{R}$ is $m \times m$ correlation matrix. On the other hand, $\{n(i)\}$. Based on these assumptions, the numerator of (4.13) is

$$P_{\underline{x}}(\underline{x}) \cdot P_{\underline{\tilde{x}}/\underline{x}}(\underline{\tilde{x}}/\underline{x}) = \left[\prod_{i=1}^{m} P_x(x(i))\right] \cdot P_{\underline{\tilde{x}}/\underline{x}}(\underline{\tilde{x}}/\underline{x}) \tag{4.17}$$

where

$$P_{\underline{\tilde{x}}/\underline{x}}(\underline{\tilde{x}}/\underline{x}) = \left(\frac{1}{2\pi|R|}\right)^{\frac{m}{2}} \exp\left[-\frac{1}{2}(\underline{\tilde{x}} - \underline{x})^T \underline{R}^{-1}(\underline{\tilde{x}} - \underline{x})\right]$$

and

$$\prod_{i=1}^{m} P_x(x(i)) \;=\; [P_X(x)]^m.$$

For mathematical tractability, we consider the case $m = 2$ and derive the memory nonlinear relationships $\underline{d}_{\text{map}}(i)$ vs $\underline{\tilde{x}}$.

For $m = 2$ the correlation matrix takes the form

$$\underline{R} = Q_n \cdot \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}, \quad |\rho| < 1. \tag{4.18}$$

For simplicity, we also define the following vectors

$$\begin{pmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{pmatrix} \;\triangleq\; \begin{pmatrix} \tilde{x}(i) \\ \tilde{x}(i-1) \end{pmatrix} \;=\; \underline{\tilde{x}}$$

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \;\triangleq\; \begin{pmatrix} x(i) \\ x(i-1) \end{pmatrix} \;=\; \underline{x}\,. \tag{4.19}$$

(i) Uniform Distribution (4.9)

Maximizing (4.17) is equivalent here to minimizing

$$J \;=\; (\underline{\tilde{x}} - \underline{x})^T \underline{R}^{-1} (\underline{\tilde{x}} - \underline{x}) \tag{4.20}$$

with the restrictions $-\lambda \le x_1 \le \lambda$, $-\lambda \le x_2 \le \lambda$. Hence, we seek a point in the area $X_2 = \{(x_1, x_2) : -\lambda \le x_1 \le \lambda,\ -\lambda \le x_2 \le \lambda\}$ such that J is minimized. Differentiating J

with respect to x_1 and x_2 and setting the derivative to zero we obtain

$$\begin{aligned}(\tilde{x}_1 - x_1) - \rho(\tilde{x}_2 - x_2) &= 0 \\ (\tilde{x}_2 - x_2) - \rho(\tilde{x}_1 - x_1) &= 0.\end{aligned} \tag{4.21}$$

From (4.21), it is apparent that if $\underline{\tilde{x}} \epsilon X_2$, that is $-\lambda \leq \tilde{x}_1 \leq \lambda$ and $-\lambda \leq \tilde{x}_2 \leq \lambda$, then

$$\underline{d}_{1\text{map}} = \begin{pmatrix} d_{1\text{map}} \\ d_{2\text{map}} \end{pmatrix} = \begin{pmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{pmatrix} \quad \text{for} \quad \tilde{x} \epsilon X_2 \tag{4.22}$$

when $\underline{\tilde{x}}$ is outside X_2, the minimum is achieved on the boundary of X_2. That is

$$\begin{aligned} d_{1\text{map}} &= k \cdot \lambda \cdot \text{sgn}[\tilde{x}_1] + (1-k) f_c[\tilde{x}_1 - \rho(\tilde{x}_2 - \lambda \text{sgn}[\tilde{x}_2])] \\ d_{2\text{map}} &= (1-k) \cdot \lambda \cdot \text{sgn}[\tilde{x}_2] + k \cdot f_c[\tilde{x}_2 - \rho(\tilde{x}_1 - \lambda \text{sgn}[\tilde{x}_1])] \\ &\text{for} \quad \underline{\tilde{x}} \notin X_2 \end{aligned} \tag{4.23}$$

where

$$f_c(x) = \begin{cases} \lambda, & x > \lambda \\ x, & |x| \leq \lambda \\ -\lambda, & x < -\lambda. \end{cases} \tag{4.24}$$

(ii) Laplace Distribution (4.11)

To obtain the MAP estimate is equivalent to minimize

$$J = \lambda|x_1| + \lambda|x_2| + \frac{1}{2}[(\underline{\tilde{x}} - \underline{x})^T \underline{R}^{-1} (\underline{\tilde{x}} - \underline{x})]. \tag{4.25}$$

The necessary conditions are

$$\begin{aligned} \lambda \text{sgn}[x_1] + c(x_1 - \tilde{x}_1) - c\rho(x_2 - \tilde{x}_2) &= 0 \\ \lambda \text{sgn}[x_2] + c(x_2 - \tilde{x}_2) - c\rho(x_1 - \tilde{x}_1) &= 0. \end{aligned} \tag{4.26}$$

where $c = \frac{1}{Q_n(1-\rho^2)}$. Clearly, (4.26) is a nonlinear system of equations. Two special cases are the following: 1) when $-\lambda/c \leq \tilde{x}_1 - \rho\tilde{x}_2,\ \tilde{x}_2 - \rho\tilde{x}_1 \leq \lambda/c$, then $d_{1\text{map}} = 0$, and 2) when $\rho = 0$, the problem reduces to the case of white Gaussian noise.

4.2 The Bussgang Algorithms

Fig. 4.1 illustrates the Bussgang adaptive blind equalization algorithms when an LMS type or stochastic gradient algorithm [53] is used for the adaptation of the equalizer coefficients, and the nonlinearity $g^{(i)}[\cdot]$ is memoryless [3], [4], [23]. The following equations, consistent with the block diagram of Fig. 4.1, describe the Bussgang family of algorithms:

$$\begin{aligned} \underline{u}(i) &= [u_1(i), \ldots, u_N(i)]^T && \text{equalizer taps} \\ \underline{u}(0) &= [0, \ldots, 1, \ldots, 0]^T && \text{initial tap values} \\ \underline{y}(i) &= [y(i), \ldots, y(i-N+1)]^T && \text{input to the equalizer block of data} \\ i &= 0, 1, 2, \ldots && \text{iteration index} \\ \tilde{x}(i) &= \underline{u}^H(i)\underline{y}(i) && \text{equalizer output or reconstructed sequence} \\ d(i) &= g^{(i)}[\tilde{x}(i)] = g^{(i)}[\underline{u}^H(i)\underline{y}(i)] && \text{output of nonlinearity} \\ e(i) &= d(i) - \tilde{x}(i) && \text{error sequence} \\ \underline{u}(i+1) &= \underline{u}(i) + \mu\underline{y}(i) \cdot e^*(i) && \text{LMS-type adaptation} \end{aligned} \tag{4.27}$$

4.2.1 Convergence Rate and Properties

From (4.27) and Figure 4.1, it is apparent that the **output sequence** of the **nonlinear function**, $d(i)$, **"plays the role" of the desired response or the training sequence.** It is also apparent that the Bussgang technique is simple to implement and understand, and it may be viewed as a minor modification of the original LMS algorithm (the desired response of the original LMS adaptation is a memoryless transformation of the transversal filter output). As such, it is expected that the technique will have convergence that will depend on the eigenvalue spread of the autocorrelation matrix of the received data $\{y(i)\}$.

From (4.27), the LMS adaptation equation for the equalizer coefficients is given by

$$\underline{u}(i+1) = \underline{u}(i) + \mu \underline{y}(i)\, e^*(i) \tag{4.28}$$

If we obtain the expected value (ensemble averaging) of (4.28), we have

$$\begin{aligned} E\{\underline{u}(i+1)\} &= E\{\underline{u}(i)\} + \mu E\left\{\underline{y}(i)\left(g^{(i)^*}[\tilde{x}(i)] - \tilde{x}^*(i)\right)\right\} \\ &= E\{\underline{u}(i)\} + \mu E\left\{\underline{y}(i) g^{(i)^*}[\tilde{x}(i)]\right\} - \mu E\{\underline{y}(i)\tilde{x}^*(i)\}. \end{aligned} \tag{4.29}$$

The adaptive algorithm converges in the mean when

$$E\left\{\underline{y}(i) g^{(i)^*}[\tilde{x}(i)]\right\} = E\{\underline{y}(i)\tilde{x}^*(i)\} \qquad \text{(equilibrium)}$$

and it converges in the mean-square when

$$E\left\{\underline{u}^H(i)\underline{y}(i) g^{(i)^*}[\tilde{x}(i)]\right\} = E\{\underline{u}^H(i)\underline{y}(i)\tilde{x}^*(i)\}$$

$$E\left\{\tilde{x}(i)g^{(i)^*}[\tilde{x}(i)]\right\} = E\{\tilde{x}(i)\tilde{x}^*(i)\}. \tag{4.30}$$

Thus, **it is required that the equalizer output $\tilde{x}(i)$ be Bussgang at equilibrium.** Note that identity (4.30) states that the autocorrelation of $\tilde{x}(i)$ (right-hand side) equals the cross correlation between $\tilde{x}(i)$ and a nonlinear transformation of $\tilde{x}(i)$ (left-hand side). Processes which satisfy property (4.30) are said to be Bussgang [10]. In summary, the adaptive Bussgang techniques converge when the equalizer output sequence, $\{\tilde{x}(i)\}$, becomes Bussgang (necessary condition).

A stochastic gradient algorithm (steepest descent) essentially minimizes iteratively a performance index $J(i) = E\{G[\tilde{x}(i)]\}$ with respect to the equalizer coefficients $\underline{u}(i)$. A more general form of the equalizer taps adaptation equation (4.28) is [25]

$$\underline{u}(i+1) = \underline{u}(i) - \mu\nabla_u J(i) \tag{4.31}$$

where $\nabla_u J(i)$ is the gradient of $J(i)$. Differentiating $J(i)$ by using the composite function rule, we obtain

$$\begin{aligned}\nabla_u J(i) &= -E\{\nabla_u[\tilde{x}(i)] \cdot \nabla_{\tilde{x}}[G(\tilde{x}(i))]\} \\ &= -E\{\underline{y}(i) \cdot \nabla_{\tilde{x}}[G(\tilde{x}(i))]\}\end{aligned} \tag{4.32}$$

By dropping the expectation operation, *i.e.*, by using a single-point unbiased estimate, we obtain

$$\hat{\nabla}_u J(i) \;=\; -\underline{y}(i)e^*(i) \tag{4.33}$$

where

$$\begin{aligned} e^*(i) &= \nabla_{\tilde{x}}[G(\tilde{x}(i))] \\ &= g^{(i)^*}[\tilde{x}(i)] - \tilde{x}^*(i) \end{aligned} \tag{4.34}$$

Equation (4.3.4) shows the relationship between the nonlinear function $g^{(i)}[\cdot]$ used in the Bussgang Techniques with the nonlinear cost function $G[\cdot]$ which defines the performance index, $J[\cdot]$.

Example for one-dimensional modulation (PAM)

The first blind equalization algorithm was introduced by Sato in 1975 [47] for PAM signals. He chose the simple nonlinear function

$$g(\tilde{x}) \;=\; \gamma \mathrm{sgn}[\tilde{x}] \tag{4.35}$$

where γ is a gain parameter which must be chosen to satisfy the Bussgang property (4.30) *i.e.*,

$$E\{\tilde{x}(i) \cdot \gamma \mathrm{sgn}[\tilde{x}(i)]\} \;=\; E\{|\tilde{x}(i)|^2\}$$

or

$$\gamma \;=\; E\{\tilde{x}(i)|^2\} \,/ E\{|\tilde{x}(i)|\}. \tag{4.36}$$

We could also write Sato's algorithm in terms of

$$G(\tilde{x}) = \frac{1}{2}\tilde{x}^2 - \gamma|\tilde{x}| \tag{4.37}$$

4.2.2 Extension to QAM modulation

The extension of Bussgang algorithms to two-dimensional constellations (QAM) is somewhat straightforward [3], [4]. In the case of two independent quadrature carriers, the conditional mean estimate of an equivalent complex transmitted symbol x given the complex observation $\tilde{x} = \tilde{x}_R + j\tilde{x}_I$ can be written as

$$d = E\{x\,/\tilde{x}\} = g[\tilde{x}_R] + jg[\tilde{x}_I]. \tag{4.38}$$

We keep the notation simple by omitting (i). For example, the Sato nonlinearity for QAM signals takes the form [47].

$$g(\tilde{x}) = \gamma \text{csgn}(\tilde{x}) = \gamma\{\text{sgn}[\tilde{x}_R] + j\ \text{sgn}[\tilde{x}_I]\}. \tag{4.39}$$

It is clear that real and imaginary parts of the data can be estimated separately. The complex data equivalent of the adaptive Bussgang Techniques is described in (4.27), but with

$$g^{(i)}[\tilde{x}(i)] \triangleq g^{(i)}[\tilde{x}_R(i)] + j\ g^{(i)}[\tilde{x}_I(i)]. \tag{4.40}$$

Consequently, the error sequence is

$$e(i) = \{g^{(i)}[\tilde{x}_R(i)] - \tilde{x}_R(i)\} + j\left\{g^{(i)}[\tilde{x}_I(i)] - \tilde{x}_I(i)\right\}. \tag{4.41}$$

For example, the "Stop-and-Go" algorithm introduced by Picchi and Prati [41] is an adaptive Bussgang technique with the following nonlinearity

$$\begin{aligned} g[\tilde{x}(i)] &= \tilde{x}(i) + \frac{1}{2}A\hat{x}(i) - \frac{1}{2}A\tilde{x}(i) \\ &\quad + \frac{1}{2}B\hat{x}^*(i) - \frac{1}{2}B\tilde{x}^*(i) \end{aligned} \tag{4.42}$$

where $\tilde{x}(i)$ is defined as the quantizer (slicer) output in Figure 4.1 and (A, B) is a pair of integers taking values $(2,0)$ or $(1,1)$ or $(1,-1)$ or $(0,0)$. The values of (A, B) are generally different at each iteration, and how they are chosen is described later in this section.

Another example of a Bussgang technique is the **heuristic modification** of the Sato algorithm suggested by Benveniste and Goursat [5], [6]. In this case, the nonlinear function takes the form

$$\begin{aligned} g[\tilde{x}(i)] &= \tilde{x}(i) + k_1\hat{x}(i) - k_1\tilde{x}(i) + \\ &\quad k_2|\hat{x}(i) - \tilde{x}(i)| \cdot [\gamma \mathrm{csgn}[\tilde{x}(i)] - \tilde{x}(i)] \end{aligned}$$

or

$$\begin{aligned} g[\tilde{x}(i)] &= \tilde{x}(i) + |\hat{x}(i) - \tilde{x}(i)| \quad \{k_1 e^{\mathrm{j}\arg[\hat{x}(i) - \tilde{x}(i)]} + \\ &\quad k_2[\gamma \mathrm{csgn}[\tilde{x}(i)] - \tilde{x}(i)]\} \end{aligned} \tag{4.43}$$

where k_2, k_2 are constants. From (4.38) we observe that the Benveniste-Goursat error function

may be seen as a weighted sum of the Decision Directed (DD) [43] and Sato errors. On the other hand the "Stop-and-Go" error function (4.37) is the weighted sum of the DD error and its conjugate. The weights of the two algorithms, however, are chosen in a completely different manner.

4.2.3 Unknown Carrier Phase: The Constant Modulus Property

Equation (4.33) can be written in polar coordinates as

$$d \;=\; E\left\{x\,/\tilde{x}\right\} \;= r\, e^{j\theta}\;. \tag{4.44}$$

If we assume that all rotated constellations are equally likely, since the carrier phase is unknown, then the conditional mean d in (4.39) has the same argument as $\tilde{x}$, and is given by

$$d = \hat{g}[|\tilde{x}|]\cdot e^{j\ \arg(\tilde{x})} \tag{4.45}$$

where $\hat{g}[\cdot]$ is a nonlinear function and $|\tilde{x}| = \sqrt{\tilde{x}_R^2 + \tilde{x}_I^2}$, $\arg(\tilde{x}) = \arctan[\tilde{x}_I/\tilde{x}_R]$. Combining (4.39) with (4.40) we obtain [3], [4], [23]

$$\begin{aligned} e(i) &= d(i) - \tilde{x}(i) \\ &= \hat{g}[|\tilde{x}(i)|]e^{j\arg[\tilde{x}(i)]} - \tilde{x}(i) \\ &= \tilde{x}(i)[\frac{g[|\tilde{x}(i)|]}{|\tilde{x}(i)|} - 1]. \end{aligned} \tag{4.46}$$

Hence, the error term is independent of any fixed phase rotation of the signal constellation. Equation (4.27) also represents the Bussgang technique for the case of unknown carrier phase,

provided we substitute $e(i)$ in (4.27) by $e(i)$ of (4.41).

Example: The Godard (or CMA) Algorithm [22], [50]

Under the assumption that all rotated constellations are equally likely, Godard [22] suggested that $\hat{g}[|\tilde{x}|]$ in (4.41) be chosen as

$$\hat{g}[|\tilde{x}|] \;=\; |\tilde{x}| + R_p|\tilde{x}|^{p-1} - |\tilde{x}|^{2p-1} \tag{4.47}$$

where R_p is a real constant. As we shall see this form has some very nice properties. Special cases of (4.42) include

$$\hat{g}[|\tilde{x}|] \;=\; (1+R_2)|\tilde{x}| - |\tilde{x}|^3 \qquad (p=2)$$

and

$$\hat{g}([|\tilde{x}|]) \;=\; R_1 \qquad (p=1).$$

The parameter R_p is a gain constant which has to be chosen according to (4.30). Since

$$g[\tilde{x}(i)] \;=\; \frac{\tilde{x}(i)\hat{g}[|\tilde{x}(i)|]}{|\tilde{x}(i)|} \tag{4.48}$$

combining (4.43) with (4.30), we obtain

$$E\{|\tilde{x}(i)|^2 + R_p|\tilde{x}(i)|^p - |\tilde{x}(i)|^{2p}\} \;=\; E\{|\tilde{x}(i)|^2\}$$

or

$$R_p \;=\; \frac{E\{|\tilde{x}(i)|^{2p}\}}{E\{|\tilde{x}(i)|^p\}} \tag{4.49}$$

At perfect equalization, $\tilde{x}(i) = x(i)e^{j\theta}$ (assuming time delay $D = 0$), and thus

$$R_p = \frac{m_{2p}}{m_p}, \quad \text{where} \quad m_p = E\{|\tilde{x}(i)|^p\}.$$

Combining (4.34) and 4.43), we obtain the Godard performance index nonlinearity, namely,

$$G(\tilde{x}(i)) = \frac{1}{2p}(|\tilde{x}(i)|^p - R_p) \tag{4.50}$$

Fig. 4.6 summarizes the nonlinear functions of the Bussgang iterative techniques.

4.2.4 The Sato and Benveniste-Goursat Algorithms

Sato [46] introduced the first blind equalization scheme in 1975 by introducing the sign nonlinearity to generate the desired response of the adaptive scheme shown in Figure 4.1, *i.e.*, $d(i) = \gamma$ sgn $[\tilde{x}(i)]$. In 1986, Sato [47] extended his 1-D PAM algorithm to the multidimensional blind equalization problem where all transmitted signals become vector processes and all impulse responses (channel and equalizer) are square matrices. The extension, however, is straightforward. For example, in the two-dimensional case of QAM signals the "sign" nonlinearity becomes the "complex sign" defined by (4.34). The error signal of the Sato algorithm

$$e_s(i) = \gamma \text{ cgn } [\tilde{x}(i)] - \tilde{x}(i) \tag{4.51}$$

is very noisy around the solution unless the transmitted sequence $x(i)$ takes only the values ± 1. In other words, although $e_s(i)$ is zero-mean at the solution, it has a large variance. On the other hand, the Decision Directed (DD) error signal $e_D(i) = \tilde{x}(i) - \hat{x}(i)$ (see Figure 4.6) [33], though not robost for blind equalizers, enjoys the property of being identically zero at the solution. Hence,

Benveniste-Goursat [5] suggested the idea of combining (heuristically) both error signals in the form of a weighted averaging as follows

$$e_{BG}(i) = k_1\, e_D(i) + k_2\, e_S(i)\, |e_D(i)| \tag{4.52}$$

where k_1, k_2 are constants. The rationale behind the error expression (4.47) is the following. Before the eye of the equalizer opens, $|e_D(i)|$ is large and thus the Sato error $e_S(i)$ contributes to the proper direction. At the opening of the eye and thereafter $|e_D(i)|$ becomes small and the DD mode of the error $e_{BG}(i)$ takes over to speed up convergence and to achieve faster rate than the original Sato algorithm with $e_S(i)$. It is no wonder, therefore, that in our simulation experience we have seen the Benveniste-Goursat (BG) algorithm exhibiting initially very slow convergence.

A faster convergence rate has been observed only after the eye opens. The Benveniste-Goursat algorithm may be seen as the Sato algorithm that switches automatically to a DD one when the eye of the equalizer opens. The extension of the Benveniste-Goursat algorithm to a Decision Feedback Equalization (DFE) implementation [2] was given by Macchi et al. [32].

4.2.5 The Godard and Donoho (or Shalvi-Weinstein) Algorithms

The basic motivation behind the development of Godard's algorithm introduced in 1980 [22] was to find a cost function that characterizes the amount of ISI at the equalizer output independently of the carrier phase. Since the input sequence $x(i)$ is *i.i.d.*, the cost function that satisfies the aforementioned conditions is

$$J^{(p)} \;=\; E\{(|\tilde{x}(i)|^p - |x(i)|^p)^q\}, \tag{4.53}$$

which depends on the input sequence, For $p = 2$, and $q = 2$, $J^{(2)}$ takes the form

$$J^{(2)} = E\{|\tilde{x}(i)|^4 + |x(i)|^4 - 2|\tilde{x}(i)|^2|x(i)|^2\} \tag{4.54}$$

where we assume that $E\{x^2(i)\} = 0$. However, (4.48) or (4.49) can not be used in practice because $\{x(i)\}$ is inaccessible. To avoid this difficulty, Godard [22] suggested the use of a dispersion function

$$D^{(p)} = E\{(|\tilde{x}(i)|^p - R_p)^q\} \tag{4.55}$$

which was shown to behave like the cost function $J^{(p)}$ and yet it is independent of the input sequence. Note that R_p is defined by (4.44). Assuming $p = 2$, $q = 2$, (4.49) and (4.50) can be written as [22]

$$\begin{aligned} J^{(2)} &= J_1 + J_2 + \\ &\left\{4(E\{|x(i)|^2\})^2 \cdot |f(0)|^2 - 2(E\{|x(i)|^2\})^2\right\} \cdot \sum_k{}' |f(k)|^2 \end{aligned} \tag{4.56}$$

and

$$\begin{aligned} D^{(2)} &= J_1 + J_2 + \\ &\left\{4(E\{|x(i)|^2\})^2 \cdot |f(0)|^2 - 2E\{|x(i)|^4\}\right\} \cdot \left\{\sum_k{}' |f(k)|^2 + R_2^2 - E\{|x(i)|^4\}\right\} \end{aligned} \tag{4.57}$$

where $\sum_k'$ is taken for $k \neq 0$ and

$$J_1 = E\{|x(i)|^4\}\,(1 - |f(0)^2) + E\{|x(i)|^4\} \cdot \sum_k{}' |f(k)|^4,$$

$$J_2 = 2(E\{|x(i)|^2\})^2 \cdot \left\{ \left(\sum_k' |f(k)|^2 \right)^2 - \sum_k' |f(k)|^4 \right\}. \tag{4.58}$$

Comparing (4.51) with (4.52), we see that for $D^{(2)}$ to be similar to $J^{(2)}$, the following inequality must be satisfied:

$$4(E\{|x(i)|^2\})^2 \, |f(0)|^2 - 2E\{|x(i)|^4\} > 0$$

or

$$|f(0)|^2 > \frac{E\{|x(i)|^4\}}{2(E\{|x(i)|^2\})^2}. \tag{4.59}$$

Godard suggests (4.53) and $f(i) = 0$ for $i \neq 0$ as a way of initializing his algorithm.

Based on what has been reported in literature [50] and on our simulation experience, the Godard algorithm has always converged to a minimum that opens the eye when Godard's initialization procedure is being followed. The Godard algorithm is summarized in (4.27) and Fig. 4.6. Its convergence for $p = 2$ is better than $p = 1$. In addition, Godard noted that convergence improves when the step size μ is divided by 2 at each 10,000 iterations [22]. The Constant Modulus Algorithm (CMA), suggested independently by Treichler and Agee in 1983 [50], is the Godard algorithm for $p = 2$ and $R_2 = 1$. Ding et al. [15] reported that the Godard-type algorithms exhibit local (not global) undesirable minima.

Shalvi and Weinstein recently introduced [48] a blind equalization scheme based on the idea of matching the kurtosis measures between the transmitted sequence $\{x(i)\}$ and the reconstructed sequence $\{\tilde{x}(i)\}$ at the output of the equalizer. The kurtosis of the input complex sequence $x(i)$, is defined by

$$K(x(i)) = E\{|x(i)|^4\} - 2E^2\{|x(i)|^2\} - |E\{x^2(i)|\}|^2 \tag{4.60}$$

which is zero for complex Gaussian random variables. The important point made in [48] is that if $E\{|\tilde{x}(i)|^2\} = E\{|x(i)|^2\}$, then (1) $|K(\tilde{x}(i))| \leq |K(x(i))|$ and (2) $|K(\tilde{x}(i))| = |K(x(i))|$ if perfect equalization is achieved. Thus, the problem is to maximize the magnitude of the kurtosis measure $|K(\tilde{x}(i))|$ in the output of the equalizer at each iteration subject to the constraint $E\{|\tilde{x}(i)|^2\} = E\{|x(i)|^2\}$. One of the special cases of the Shalvi-Weinstein algorithm is the original Godard algorithm. It has recently been recently reported that the Shalvi-Weinstein algorithm was originally introduced by Donoho [16] for real-valued signals and that the algorithm's convergence is only guaranteed for infinite-length equalization filters.

4.2.6 The Stop-and-Go and Decision-Directed Algorithms

The basic idea behind the Stop-and-Go algorithm, which was proposed by Picchi and Prati [41] in 1987, is to retain the advantages of simplicity and fast convergence (in open eye-pattern conditions) of the Decision directed (DD) algorithm [33] while attempting to improve its blind convergence capabilities.

The adaptation error $e_D(i)$ used in the DD algorithm is [33]

$$e_D(i) = \hat{x}(i) - \tilde{x}(i) \tag{4.61}$$

where $\tilde{x}(i)$ is the output of the equalizer and $\hat{x}(i)$ the output of the threshold detector. Assuming that the equalizer initial tap setting corresponds to a closed eye-pattern, $e_D(i)$ will be large most of the time due to the large number of incorrect decisions $\hat{x}(i)$. Consequently, the DD algorithm cannot converge in closed eye-pattern conditions.

In the Stop-and-Go algorithm, Picchi and Prati proposed the use of the error sequence

$$e(i) = \frac{1}{2}\{A(i)e_D(i) + B(i)e_D^*(i)\} \tag{4.62}$$

where

$$\begin{aligned} A(i) &= I_R(i) + I_I(i) \\ B(i) &= I_R(i) - I_I(i) \end{aligned}$$

and

$$I_R(i) = \begin{cases} 1, & \text{if} \quad \text{sgn}[e_D(i)]_R = \text{sgn}[e_S(i)]_R \\ 0, & \text{otherwise} \end{cases}$$

$$I_I(i) = \begin{cases} 1, & \text{if} \quad \text{sgn}[e_D(i)]_I = \text{sgn}[e_S(i)]_I \\ 0, & \text{otherwise.} \end{cases}$$

Note that $e_S(i)$ is the Sato error given by (4.46).

From the foregoing, it is clear that the Stop-and-Go algorithm is essentially the DD algorithm when the eye is open. It is mostly during closed eye-pattern conditions that the Stop-and-Go adaptation rule takes place. Also, it is clear that the Benveniste-Goursat and Stop-and-Go algorithms have different convergence properties when the eye-pattern is closed and similar convergence properties when the eye is open. The modifications of this algorithms have been proposed to incorporate joint equalization and carrier recovery, decision feedback equalization [1] as well as fractionally spaced equalization [21], [45].

4.3 The CRIMNO Algorithm

Although the Bussgang algorithms are different from each other, as we have seen, they perform only memoryless nonlinear transformations on the equalizer outputs to generate the desired response. This, in turn, implies that the cost functions they attempt to minimize with respect to the equalizer coefficients are also memoryless. These algorithms do not explicitly employ the fact that *the transmitted data are statistically independent*, which is the essence of the new criterion we introduce in this section. Since statistical independence of the transmitted data involves more than one data symbols, this results in a memory nonlinear transformation on the equalizer outputs and thus a memory nonlinear cost function.

4.3.1 Criterion with Memory Nonlinearity

As we have seen, Godard solves the blind equalization problem by proposing a cost function which is independent of the transmitted data, and yet reaches its global minimum at perfect equalization. The Godard cost function (also known as the constant modulus algorithm (CMA) [22] is given by (4.50) and (4.44).

Note that only the expected value of some function of the *current* equalizer output appears in Godard's cost function. Therefore, the Godard criterion only makes use of the probability distribution of the transmitted data. It does not explicitly use the fact that the *transmitted data are statistically independent.*

Assume that perfect equalization is achievable and consider the situation where perfect equalization has indeed been achieved. That is

$$\tilde{x}(i) \;=\; x(i-D)$$

where d is some positive number, which accounts for the delay. Since the transmitted data $x(i)$ are statistically independent from each other, so are the equalizer outputs $\tilde{x}(i)$ at perfect equalization. In addition, for most transmitted data constellations, the mean of transmitted data $x(i)$ is zero. Therefore, at perfect equalization , we have

$$E\{\tilde{x}(i)\tilde{x}^*(i-l)\} = E\{x(i-D)x^*(i-l-D)\} = E\{x(i-D)\} \cdot E\{x^*(i-l-D)\} = 0$$

By making use of this property and combining it with Godard's criterion, we obtain a new criterion, called criterion with memory nonlinearity (CRIMNO), which is the minimization of the following cost function:

$$M^{(p)} = w_0 E\left(|\tilde{x}(i)|^p - R_p\right)^2 + w_1 \left|E\{\tilde{x}(i)\tilde{x}^*(i-1)\}\right|^2 + \cdots + w_M |E\{\tilde{x}(i)\tilde{x}^*(i-M)\}|^2. \quad (4.63)$$

The rationale behind the CRIMNO is that since each term reaches its global minimum at perfect equalization, by appropriately combining them, we can increase the convergence speed of the corresponding CRIMNO algorithm [12], [13]. This is clearly demonstrated in the simulations section.

Remarks:

1. Memory nonlinearity: the CRIMNO cost function depends not only on the *current* equalizer output, but also on the *previous* equalizer outputs. As such, it results to a criterion with memory nonlinearity. The parameter M determines the size of memory.

2. Generalization of the Godard criterion: when $w_0 = 1$, $w_i = 0$ for $i \neq 0$, the CRIMNO cost function reduces to the Godard cost function. Therefore, the CRIMNO criterion may be seen as a generalization of the Godard criterion.

3. Constant Modulus Property: the CRIMNO criterion preserves the constant modulus property inherent in Godard.

4.3.2 CRIMNO Blind Equalization Algorithm

Define the equalizer coefficient vector $\underline{u}(i) \triangleq [u_1(i), \cdots, u_N(i)]^T$, and the received signal vector $\underline{y}(i) \triangleq [y(i), \cdots, y(i-N+1)]^T$, where N is the length of the equalizer. Then the equalizer outputs are

$$\tilde{x}(i-l) = \underline{y}^T(i-l) \cdot \underline{u}(i),\ l = 0, 1, \cdots, M, \tag{4.64}$$

where superscript T denotes transposition of a vector.

Differentiating the cost function $M^{(2)}$ with respect to the equalizer coefficient vector $\underline{u}(i)$, we obtain [12]

$$\begin{aligned}\frac{\partial M^{(2)}}{\partial \underline{u}(i)} &= 4w_0 E[\underline{y}^*(i)\tilde{x}(i)(|\tilde{x}(i)|^2 - R_2)] \\ &+2w_1[E(\underline{y}^*(i-1)\tilde{x}(i))E(\tilde{x}^*(i)\tilde{x}(i-1)) + E(\underline{y}^*(i)\tilde{x}(i-1))E(\tilde{x}(i)\tilde{x}^*(i-1))] \\ &+\cdots \\ &+2w_M[E(\underline{y}^*(i-M)\tilde{x}(i))E(\tilde{x}^*(i)\tilde{x}(i-M)) + E(\underline{y}^*(i)\tilde{x}(i-M))E(\tilde{x}(i)\tilde{x}^*(i-M))].\end{aligned} \tag{4.65}$$

By using the steepest descent method to search for the minimum point, we obtain

$$\underline{u}(i+1) = \underline{u}(i) - \alpha \cdot \{4w_0 E[\underline{y}^*(i)\tilde{x}(i)(|\tilde{x}(i)|^2 - R_2]$$

$$+2w_1[E(\underline{y}^*(i-1)\tilde{x}(i))E(\tilde{x}^*(i)\tilde{x}(i-1)) + E(\underline{y}^*(i)\tilde{x}(i-1))E(\tilde{x}(i)\tilde{x}^*(i-1))]$$

$$+\cdots$$

$$+2w_M[E(\underline{y}^*(i-M)\tilde{x}(i))E(\tilde{x}^*(i)\tilde{x}(i-M)) + E(\underline{y}^*(i)\tilde{x}(i-M))E(\tilde{x}(i)\tilde{x}^*(i-M))] \quad (4.66)$$

where

$$\underline{u}(i) \triangleq [u_1(i), \ldots, u_N(i)]^T$$

In (4.6), the expectation are the ensemble averages taken with respect to transmitted data $x(i)$ while the channel impulse response $f(i)$ and the equalizer coefficients $u(i)$ are treated as fixed.

If we use single point estimates for the ensemble averages, we obtain the stochastic gradient CRIMNO algorithm:

$$\begin{aligned}
\underline{u}(i+1) &= \underline{u}(i) - \alpha[4w_0\underline{y}^*(i)\tilde{x}(i)(|\tilde{x}(i)|^2 - R_2) + 2w_1(\underline{y}^*(i-1)\tilde{x}(i)|\tilde{x}(i-1)|^2 + \underline{y}^*(i-1)\tilde{x}(i-1)|\tilde{x}(i)|^2) \\
&\quad + \cdots + 2w_M(\underline{y}^*(i)\tilde{x}(i)|\tilde{x}(i-M)|^2 + \underline{y}^*(i-M)\tilde{x}(i-M)|\tilde{x}(i)|^2)] \\
&= \underline{u}(i) - \alpha[\underline{y}^*(i)\tilde{x}(i) * (4w_0|\tilde{x}(i)|^2 + 2w_1|\tilde{x}(i-1)|^2 + \cdots + 2w_M|\tilde{x}(i-M)|^2 - 4w_0R_2) \\
&\quad + 2w_1\underline{y}^*(i-1)\tilde{x}(i-1)|\tilde{x}(i)|^2 + \cdots + 2w_M\underline{y}^*(i-M)\tilde{x}(i-M)|\tilde{x}(i)|^2]. \qquad (4.67)
\end{aligned}$$

Note that at each iteration, all equalizer outputs $\tilde{x}(i-l), l = 0, 1, \cdots, M$ are recalculated using current (most recent) equalizer coefficient vector $\underline{u}(i)$ via $\tilde{x}(i-l) = \underline{y}^T(i-l)\underline{u}(i)$. This requires a lot of computations. If, instead of using the current equalizer coefficient vector $\underline{u}(i)$, we use the delayed equalizer coefficient vector $\underline{u}(i-l)$ to calculate $\tilde{x}(i-l)$. Note that (for small step-size, which is required for the stability of stochastic gradient-type algorithm, the difference between $\underline{u}(i)$ and $\underline{u}(i-l)$ is negligible. Then at each iteration we will need to calculate only one equalizer

output $\tilde{x}(i)$ using the current equalizer coefficient vector $\underline{u}(i)$.

4.3.3 Adaptive Weight CRIMNO Algorithm

The shape of the cost function depends on the choice of weight w_l. So does the performance of the CRIMNO algorithm. Here, we describe an ad hoc way of adjusting the weights on-line in the blind equalization process.

The basic idea is to estimate the values of all terms in the CRIMNO cost function over a block of data and then set the weights used in the next block proportional to the deviations of the corresponding terms from their ideal values at perfect equalization. The rationale behind this scheme is that if one term in the criterion has a large deviation from its ideal value, then in the next block the weight associated with it will be set equal to a large value, and consequently, the gradient-descent method will bring it down quickly.

To elaborate on this idea, we rewrite the CRIMNO cost function as

$$M^{(p)} = w_0 J_0 + w_1 J_1 + \cdots + w_M J_M, \tag{4.68}$$

where

$$\begin{aligned} J_0 &= E(|\tilde{x}(i)|^p - R_p)^2 \\ J_l &= |E(\tilde{x}(i)\tilde{x}^*(i-l)|^2 \; 1 \le l \le M. \end{aligned} \tag{4.69}$$

Define the deviation of the lth term $D(J_l)$ by

$$D(J_l) \triangleq |J_l - J_l^{(o)}|, \tag{4.70}$$

where $J_l^{(o)}$ is the value of J_l at perfect equalization ($J_l^{(o)} = 0,\ l = 1, \cdots, M$). Then the weights are adjusted using the following formulae:

$$\begin{aligned} w_0 &= \begin{cases} \gamma_0 D(J_0) & \gamma_0 D(J_0) < \lambda \\ \lambda & \gamma_0 D(J_0) \geq \lambda \end{cases} \\ w_i &= \begin{cases} \gamma D(J_l) & \gamma D(J_l) < \lambda \\ \lambda & \gamma D(J_l) \geq \lambda \end{cases} \end{aligned} \tag{4.71}$$

where $\lambda_0 > 0$ is the scaling constant for the first term, $\gamma > 0$ is the scaling constant for the other terms in the CRIMNO cost function, and λ is a constraint on the maximum value of the weights to guarantee the stability of the algorithm.

The CRIMNO algorithm with weights adjusted in this way is called adaptive weight CRIMNO algorithm. Some in-depth comments are provided below:

1. When the deviations of all terms vary proportionally, the adaptive weight scheme becomes an adaptive step-size algorithm. Moreover, the adaptation is done automatically. So when the algorithm converges, then weights decrease to zero. Hence, the adaptive weight CRIMNO algorithm acquires as a byproduct the decreasing step-size, which has been proven to be an optimal strategy for equalization [51].

2. For the adaptive weight CRIMNO algorithm, the shape of the cost function is changing.

The local minima of the cost function are also changing. Thus, what is local minimum of the cost function at one iteration may not be at the next iteration. However, **whatever the change of the weights, the global minimum does not change, and it always corresponds to perfect equalization.**

3. The adaptive weight CRIMNO algorithm tends to move out of a local minimum of the cost function quickly, if the cost function has local minima and the algorithm gets trapped in one of them. This is based on the following arguments. In the adaptive weight CRIMNO algorithm, the equalizer coefficient increment, $\Delta\underline{u}(i+1) = \underline{u}(i+1)-\underline{u}(i)$ is a random vector, the variance of which determines how fast the algorithm will move out of a local minimum. The variance of the equalizer coefficient increment depends on the step-size α, gradient $\frac{\partial J_l}{\partial \underline{u}(i)}$ and the weights w_l (proportional to $D(J_l)$). The step-size and gradient are the same with the fixed weight CRIMNO algorithm; we thus concentrate on the third one: w_l, or equivalently $D(J_l)$. At a global minimum of the cost function, $D(J_l)$ are all small, thus, the variance of the equalizer coefficient increment is small. Therefore, the algorithm will remain near the global minimum. However, that is not the case with a local minimum. In that case, $D(J_l)$ will be large, therefore, the variance of the equalizer coefficient increment will be large (relative ot the case at the global minimum), and the algorithm will move out of that minimum quickly. Moreover, the larger the deviation $D(J_l)$, the more quickly the algorithm will move out of the local minimum.

4. Blocks of data are used to estimate $\{J_l\}$. The block length should be sufficiently long to make the variances of the estimates small, but not long enough to make the weight update fall behind.

4.3.4 CRIMNO Extensions

In this section, the CRIMNO ideas, *i.e.*, memory nonlinearity, are extended to the following cases: (1) the case of correlated inputs; (2) the case when higher-order correlation terms [38] are utilized.

Colored CRIMNO

One of the key assumptions in the CRIMNO criterion is that the transmitted data are independent and identically distributed (*i.i.d*). However, in practice, this may not be true for QAM signals. Usually, in order to overcome the phase ambiguity caused by the squaring loop for carrier recovery, differential encoding techniques are used, which correlate the input data when the source symbols are not equiprobable. Since the operations of differential encoding are known, the autocorrelations of the input data can be derived. In the case where the autocorrelations of the input data are known a priori, the CRIMNO criterion can be modified as follows:

$$M_c^{(p)} = w_0 E(|\tilde{x}(i)|^p - R_p)^2 + w_1 |E(\tilde{x}(i)\tilde{x}^*(i-1) - \beta_1|^2 + \cdots + w_M |E(\tilde{x}(i)\tilde{x}^*(i-M)) - \beta_M|^2 \quad (4.72)$$

where $\beta_l \triangleq E(x(i)x^*(i-l))$ are **the known autocorrelations of the transmitted data.**

Higher-Order Correlation CRIMNO

Here, a criterion which exploits the higher-order correlations, such as the fourth-order statistics of the equalizer output, is given below:

$$\begin{aligned} M_h^{(p)} = {} & w_0 E(|\tilde{x}(i)|^p - R_p)^2 + \sum_l w_l |E(\tilde{x}(i)\tilde{x}^*(i-l))|^2 \\ & + \sum_{j,k,l \text{ all different}} v_{jkl} |E(\tilde{x}(i)\tilde{x}^*(i-j)\tilde{x}(i-k)\tilde{x}^*(i-l)|^2 \end{aligned} \quad (4.73)$$

The performance of both (4.73) and (4.74) criteria needs to be investigated.

4.3.5 Computer Simulation

Computer simulations have been conducted to compare the performance of the adaptive weight CRIMNO algorithm with that of the Godard (or CMA) algorithm. Fig. 4.6 shows the performance of the adaptive weight CRIMNO algorithm, compared with that of the Godard algorithm under the different step-sizes, including the optimum one: We see that the performance of the adaptive weight CRIMNO algorithm is better than or approaches that of the Godard algorithm with optimum step-size. Fig. 4.7 shows the performance of the adaptive weight CRIMNO algorithm for different memory sizes ($M = 2.4.6$). Fig. 4.8 shows that the corresponding eye-patterns at iteration 20000. We see that the larger the memory size M, the better the performance of the adaptive weight CRIMNO algorithm. Table 4.2 lists the computational complexity of the CRIMNO algorithm, the adaptive weight CRIMNO algorithm, and the Godard algorithm. We see that there is only a little increase in computational complexity. Therefore, the performance improvement is achieved at the expense of little increase in computational complexity.

5 ALGORITHMS WITH NONLINEARITY IN THE INPUT OF THE EQUALIZATION FILTER

The Polyspectra Based Techniques

Another class of blind equalization algorithms are those algorithms which are based on higher-order cumulants or polyspectra [36], such as the tricepstrum equalization algorithm (TEA) [24], the power cepstrum and tricoherence equalization algorithm (POTEA) [7], and the cross-tricepstrum equalization algorithm (CTEA) [8]. All these algorithms perform nonlinear transfor-

mation on the input of the equalization filter. This nonlinear transformation, *e.g.* the generation of the higher-order cumulants or polyspectra of the received data, is a memory nonlinear transformation, because it employs both the present and the past values of the received data. The use of the higher-order statistics of the received data is necessary for blind equalization, since the correct phase information about the channel can not be extracted from only the second-order statistics of the received data [14], [29], [34], [35], [37], [42].

5.1 Definitions and Properties: Cumulants and Higher Order Spectra

The readers are assumed to be somewhat familiar with the basic material of higher-order spectra. However, some important properties which will be used in the subsequent sections are given.

5.1.1 Definitions

1. Definition of Cumulants:

   Given a set of n real random variables $\{x_1, x_2, \cdots, x_n\}$, their nth joint cumulants of order is defined as

$$L(x_1, x_2, \cdots, x_n) \triangleq (-j)^n \frac{\partial^n ln\Phi(v_1, v_2, \cdots, v_n)}{\partial v_1 \partial v_2 \cdots \partial v_n}\Bigg| v_1 = v_2 = \cdots = v_n = 0 \qquad (5.1)$$

   where

$$\Phi(v_1, v_2, \cdots, v_n) = E\{\exp\, j(v_1 x_1 + \cdots + v_n x_n)\}. \qquad (5.2)$$

   Given a real stationary random sequence $\{x(i)\}$ with zero mean, $E\{x(i)\} = 0$, then the nth-order cumulant of the random sequence depends only on the time difference and is

defined as

$$L_x(\tau_1, \tau_2, \cdots, \tau_{n-1}) \triangleq (-j)^n \frac{\partial^n ln\Phi_x(v_1, v_2, \cdots, v_n)}{\partial v_1 \partial v_2 \cdots \partial v_n}\Bigg| v_1 = v_2 = \cdots = v_n = 0 \qquad (5.3)$$

where $\tau_1, \tau_2, \cdots, \tau_{n-1}$ are integers and

$$\Phi_x(v_1, v_2, \cdots, v_n) = E\{\exp j(v_1 x(i) + v_2 x(i+\tau_1) + \cdots + v_n x(i+\tau_{n-1}))\} \qquad (5.4)$$

Given a set of real jointly stationary random sequences $\{x_k(i)\}$, $k = 1, 2, \cdots, n$ with zero mean, $E\{x_k(i)\} = 0$, then the nth-order cross-cumulant of the sequences depends only on the time difference and is defined as

$$L_{x,1,2,\cdots,n}(\tau_1, \tau_2, \cdots, \tau_{n-1}) \triangleq (-j)^n \frac{\partial^n ln\Phi_{x,1,2,\cdots,n}(v_1, v_2, \cdots, v_n)}{\partial v_1 \partial v_2 \cdots \partial v_n}\Bigg| v_1 = v_2 = \cdots = v_n = 0 \qquad (5.5)$$

where $\tau_1, \tau_2, \cdots, \tau_{n-1}$ are integers and

$$\Phi_{x,1,2,\cdots,n}(v_1, v_2, \cdots, v_n) = E\{\exp j(v_1 x_1(i) + v_2 x_2(i+\tau_1) + \cdots + v_n x_n(i+\tau_{n-1}))\}. \qquad (5.6)$$

2. Definitions of Higher-Order Spectra.

Higher-order spectra are defined to be the Z-transforms of the corresponding cumulants [34], [38]. Specifically, a nth-order spectrum of a real stationary zero mean random sequence $\{x(i)\}$ is just the $(n-1)$-dimensional Fourier transform of the nth-order cumulant $L_x(\tau_1, \tau_2, \cdots, \tau_{n-1})$ of the random sequence. That is

$$S_x(z_1, z_2 \cdots, z_{n-1}) \triangleq \sum_{\tau_1,\tau_2,\cdots,\tau_{n-1}} L_x(\tau_1, \tau_2, \cdots, \tau_{n-1}) \prod_{l=1}^{n-1} Z_l^{-\tau_l}. \tag{5.7}$$

When $n = 2, 3, 4$ the corresponding spectrum is called power spectrum, bispectrum, and trispectrum, respectively.

A nth-order cross-spectrum of a set of real stationary zero mean random sequences $\{x_k(i)\}$, $k = 1, 2, \cdots, n$, is defined as the $(n-1)$ dimensional Z-transform of the nth-order cumulant $L_{x,1,2,\cdots,n}(\tau_1, \tau_2, \cdots \tau_{n-1})$ of the random sequence, that is

$$S_{x,1,2,\cdots,n}(z_1, z_2, \cdots, z_{n-1}) \triangleq \sum_{\tau_1,\tau_2,\cdots\tau_{n-1}} L_{x,1,2,\cdots,n}(\tau_1, \tau_2, \cdots, \tau_{n-1}) \prod_{l=1}^{n-1} z_l^{-\tau_l}. \tag{5.8}$$

3. Definitions of coherence.

Coherence is defined as the higher-order spectrum normalized by the power spectrum. Specifically, a nth-order coherence of a real stationary zero mean random sequence $x(i)$ is defined as

$$R_x(z_1, z_2, \cdots, z_{n-1}) \triangleq \frac{S_x(z_1, z_2, \cdots, z_{n-1})}{[S_x(z_1)S_x(z_2)\cdots S_x(z_{n-1})S_x(\prod_{l=1}^{n-1} z_l^{-1})]^{\frac{1}{2}}} \tag{5.9}$$

An alternative definition for the nth-order coherence, which is equivalent to the above definitions, is

$$R_x(z_1, z_2, \cdots, z_{n-1}) \triangleq \left[\frac{S_x(z_1, z_2, \cdots, z_{n-1})}{S_x(z_1^{-1}, z_2^{-1}, \cdots, z_{n-1}^{-1})}\right]^{\frac{1}{2}} \tag{5.10}$$

4. Definitions of Cepstrum of Higher-Order Spectrum

The cepstrum is defined as the inverse Z-transform of the log function of the spectrum. Specifically, a cepstrum for the nth-order spectrum of a real stationary zero mean random

sequence $\{x(i)\}$ is defined as

$$c_x(\tau_1, \tau_2, \cdots, \tau_{n-1}) \triangleq Z^{-1}\left[ln\ S_x(z_1, z_2, \cdots, z_{n-1})\right] \tag{5.11}$$

A cepstrum for the nth-order cross spectrum of a set of real stationary zero mean random sequence $\{x(i)\}, i = 1, 2, \cdots, n$, is defined as

$$c_{x,1,2,\cdots,n}(\tau_1, \tau_2, \cdots \tau_{n-1}) \triangleq Z^{-1}\left[ln\ S_{x,1,2,\cdots,n}(z_1, z_2, \cdots, z_{n-1})\right] \tag{5.12}$$

When $n = 2, 3, 4$, the corresponding cepstrum is called power cepstrum, bicepstrum and tricepstrum, respectively.

5.1.2 Properties

Some important properties of cumulants are shown below.

1. If $x_1, x_2, \cdots, x_n$ can be divided into two or more groups which are statistically independent, then the cumulant $L(x_1, x_2, \cdots, x_n)$ is zero.

   Specifically, if $\{x(i)\}$ are an independent, identically distributed random variables, the nth-order cumulant of the sequence $\{x(i)\}$ is

$$L_x(\tau_1, \tau_2, \cdots, \tau_{n-1}) = \gamma\delta(\tau_1)\delta(\tau_2)\cdots\delta(\tau_{n-1}) \tag{5.13}$$

2. Cumulants of higher order ($n \geq 3$) are zero for Gaussian processes.

3. If $\{x(i)\}$ and $\{y(i)\}$ are statistically independent random sequences and, $z(i) = x(i) + y(i)$, then

$$L_z(\tau_1, \tau_2, \cdots, \tau_{n-1}) = L_x(\tau_1, \tau_2, \cdots, \tau_{n-1}) + L_y(\tau_1, \tau_2, \cdots, \tau_{n-1}). \tag{5.14}$$

5.2 Tricepstrum Equalization Algorithm (TEA)

5.2.1 Problem Formulations

We assume that the received sequence after being demodulated, low-pass filtered and synchronously sampled (at rate $\frac{1}{T}$) can be written as:

$$y(i) = z(i) + w(i) = \sum_{k=-L_1}^{L_2} f(k)x(i-k) + w(i) \tag{5.15}$$

where the nonminimum phase equivalent channel impulse response $\{f(i)\}$ accounts for the transmitter filter, non-ideal channel (or multipath propagation), and receiver filter impulse response; the input data sequence $\{x(i)\}$ is generally complex, non-Gaussian, white, *i.i.d.*, with $E\{x(i)\} = 0$, $E\{x(i)^3\} = 0$ and $E\{x(i)^4\} - 3[E\{x(i)^2\}]^2 = \gamma_x \neq 0$; for example $\{x(i)\}$ could be a multi-level symmetric PAM sequence or the complex baseband equivalent sequence of a symmetric QAM signal; the additive noise $\{w(i)\}$ is zero-mean, Gaussian, generally complex and statistically independent from $\{x(i)\}$; we also assume that the channel transfer function $F(z)$ (Z-transform of $\{f(i)\}$) admits the factorization [24]

$$F(z) = A \cdot I(z^{-1}) \cdot O(z) \tag{5.16}$$

the factor $I(z^{-1}) = \frac{\prod_{k=1}^{L_3}(1-a_k z^{-1})}{\prod_{k=1}^{L_4}(1-c_k z^{-1})}, |a_k| < 1, |c_k| < 1$, is a minimum phase polynomial, *i.e.*, with zeros and poles inside the unit circle. The factor $O(z) = \prod_{k=1}^{L_2}(1 - b_k z), |b_k| < 1$ is a maximum

phase polynomial, *i.e.*, with zeros outside the unit circle. The parameter A is a constant gain factor. Finally, the sequence $\{y(i)\}$ is the input to the blind equalizer.

5.2.2 Relations of Tricepstrum of the Linear Filter Output

The input to the channel, $x(i)$, is a non-Gaussian i.i.d. random sequence, thus

$$S_x(z_1, z_2, z_3) = \gamma_x. \tag{5.17}$$

The trispectrum of the output, $y(i)$, of the channel (linear filter) is

$$\begin{aligned} S_y(z_1, z_2, z_3) &= \gamma_x F(z_1)F(z_2)F(z_3)F(z_1^{-1}z_2^{-1}z_3^{-1}) \\ &= \gamma_x \cdot A^4 \cdot I(z_1^{-1})I(z_2^{-1}) \cdot I(z_3^{-1}) \cdot I(z_1, z_2, z_3)\, O(z_1) \cdot O(z_2) \cdot O(z_3) \cdot O(z_1^{-1}z_2^{-1}z_3^{-1}) \end{aligned} \tag{5.18}$$

Taking the logarithm of $S_y(z_1, z_2, z_3)$ and then the inverse Z-transform, after some manipulation, we obtain [24]

$$c_y(m,n,l) = \frac{1}{2}\begin{cases} \log(\gamma_x A^4) & m=n=l=0 \\ -\frac{1}{m}A^{(m)} & m>0, n=l=0 \\ -\frac{1}{n}A^{(n)} & n>0, m=l=0 \\ -\frac{1}{l}A^{(l)} & l>0, m=n=0 \\ \frac{1}{m}B^{(-m)} & m<0, n=l=0 \\ \frac{1}{n}B^{(-n)} & n<0, m=l=0 \\ \frac{1}{l}B^{(-l)} & l<0, m=n=0 \\ -\frac{1}{n}B^{(n)} & m=n=l>0 \\ \frac{l}{n}A^{(n)} & m=n=l<0 \\ 0 & \text{otherwise} \end{cases} \tag{5.19}$$

where, $A^{(I)}, B^{(J)}$ are the minimum and maximum phase differential cepstrum parameters of the system, corresponding to $I(z^{-1})$ and $O(z)$, respectively. They are defined as follows:

$$A^{(I)} \stackrel{def}{=} \sum_{k=1}^{L_3} a_k^I - \sum_{k=1}^{L_4} c_k^I \qquad B^{(J)} \stackrel{def}{=} \sum_{k=1}^{L_2} b_k^J \tag{5.20}$$

In addition, the following identity holds between the fourth-order cumulants $L_y(m,n,l)$ and the tricepstrum $c_y(m,n,l)$:

$$\sum_{J=1}^{\infty}\Big\{A^{(J)}[L_y(m-J,n,l) - L_y(m+J,n+J,l+J)]\Big\} + \sum_{J=1}^{\infty}\Big\{B^{(J)}[L_y(m-J,n-J,l-J) - L_y(m+J,n,l)]\Big\} = -m \cdot L_y(m,n,l) \tag{5.21}$$

where we define,

$$J \cdot c_y(J,0,0) = \begin{cases} -A^{(J)}, & J = 1, \ldots \infty \\ B^{(-J)}, & J = -1, \ldots - \infty. \end{cases}$$

$A^{(J)}, B^{(J)}, J = 1, 2, \ldots$ are the minimum and maximum phase cepstral coefficients respectively, which are related to the zeros $F(z)$. However, in practice, the summation terms in (5.21) can be approximated by arbitrarily large but finite values because $A^{(J)}$ and $B^{(J)}$ decay exponentially as J increases.

In practice the fourth-order cumulants $L_y(\cdot)$ in (5.21) need to be substituted by their estimates $\hat{L}_y(\cdot)$ obtained from a finite length window of the received samples $\{y(i)\}$.

The TEA algorithm, uses (5.21) in order to form an overdetermined system of equations, *i.e.*, we have more equations than unknowns. Then, TEA solves this overdetermined system of equations, adaptively, using an LMS adaptation algorithm. At each iteration an estimate of the cepstral parameters $\{A^{(I)}\}$ and $\{B^{(J)}\}$ is computed. The coefficients of the equalizer are calculated for $\{A^{(I)}\}$ and $\{B^{(J)}\}$ by means of the iterative formulas.

5.2.3 TEA Algorithm

Let:

$\{y(i)\}$:	The received zero-mean synchronously sampled communication signal.
N_1, N_2:	Lengths of minimum and maximum phase components of the equalizer.
p, q:	Lengths of minimum and maximum phase cepstral parameters.
$\hat{M}_y^{(i)}(m,n,l)$:	Estimated fourth-order moments of $\{y(i)\}$ at iteration (i).
$\hat{R}_y^{(i)}(j)$:	Estimated second-order moments of $\{y(i)\}$ at iteration (i).
$\hat{L}_y^{(i)}(m,n,l)$:	Estimated fourth-order cumulants of $\{y(i)\}$ at iteration (i).

Symmetric PAM or QAM Signaling:

In general, for 1-D (*e.g.* PAM) or 2-D (*e.g.* QAM) signaling with symmetric constellations:

$$\hat{L}_y^{(i)}(m,n,l) = \hat{M}_y^{(i)}(m,n,l) - \hat{R}_y^{(i)}(m) \cdot \hat{R}_y^{(i)}(n-l) - \hat{R}_y^{(i)}(n)\hat{R}_y^{(i)}(l-m) - \hat{R}_y^{(i)}(l)\hat{R}_y^{(i)}(n-m) \quad (5.22)$$

For symmetric square ($L \times L$) QAM constellations:

$$\hat{L}_y^{(i)}(m,n,l) = \hat{M}_y^{(i)}(m,n,l) \quad (5.23)$$

and $A_{(i)}^{(I)}, B_{(i)}^{(J)}$ are the minimum and maximum phase differential cepstrum parameters at iteration (i) respectively. L_1 and L_2 are the orders of the minimum phase and maximum phase components of the FIR channel, respectively. Note that, $\{a_i\}$, $|a_i| < 1$ and $\{\frac{1}{b_i}\}$, $|b_i| < 1$ are the zeros of the minimum and maximum phase components of the FIR channel, respectively.

$\{u(i)\}$: The coefficients of the equalizer at iteration (i).

$\{\tilde{x}(i)\}$: The coefficients of the equalizer at iteration (i).

At iteration (i): $i = 1, 2, \ldots$

Step 1 Estimate adaptively the $L_y^{(i)}(m,n,l)$, $-M \leq m,n,l \leq M$, from finite length window of $\{y(k)\}$ as described below. M should be sufficiently large so that $L_y(m,n,l) \simeq 0$ for $|m|$, $|n|$, $|l| > M$. Assuming that at iteration (0) we have received the time samples $\{y(1), \ldots y(I_{\text{lag}})\}$ we proceed as follows:

Stationary Case with Growing Rectangular Window

$$\hat{M}_y^{(i)}(m,n,l) = (1-\eta(i)) \cdot \hat{M}_y^{(i-1)}(m,n,l) + \eta(i) \cdot y(S_4^i)y(S_4^i+m)y(S_4^i+n)y(S_4^i+l) \quad (5.24)$$

$$\hat{R}_y^{(i)}(j) = (1-\eta(i)) \cdot \hat{R}_y^{(i-1)}(j) + \eta(i) \cdot y(S_2^i)y(S_2^i+j) \quad (5.25)$$

where, $\eta(i) = \frac{1}{i+I_{\text{lag}}}$, $S_2^i = \min(i+I_{\text{lag}}, i+I_{\text{lag}}-m, i+I_{\text{lag}}-n, i+I_{\text{lag}}-l)$, $S_2^i = \min(i+$

$I_{\text{lag}}, i + I_{\text{lag}} - j)$. Finally substitute (5.24) and (5.25) into (5.22) or (5.23).

Nonstationary Case

First Way:

$$\begin{aligned} &\text{for} \quad i \leq K \quad \text{use (5.24), (5.25) with} \quad \eta(i) = \frac{1}{i + I_{\text{lag}}} \\ &\text{for} \quad i > K \quad \text{use (5.24), (5.25) with} \quad \eta(i) = \eta = \text{fixed} \end{aligned} \tag{5.26}$$

η should have a small value $(0 < \eta < 1)$, for example $\eta = 0.01$.

Second Way: (for symmetric L^2- QAM signaling)

Since in this case the second-order moment $R_y(j) = 0$, we can use $M_y(m, n, l)$ with a forgetting factor $w, 0 < w < 1$ as follows. (S_4^i is as before):

$$(i+I_{\text{lag}}) \cdot \hat{M}_y^{(i)}(m,n,l) = w \cdot (i-1+I_{\text{lag}}) \cdot \hat{M}_y^{(i-1)}(m,n,l) + y(S_4^i) y(S_4^i+m) y(S_4^i+n) y(S_4^i+l) \tag{5.27}$$

and substitute $(i + I_{\text{lag}}) \cdot \hat{M}_y^{(i)}(m, n, l)$ for $\hat{L}_y^{(i)}(m, n, l)$ everywhere.

Third Way:

Formulas (5.24) and (5.25) could be used in nonstationary environments by reinitializing the algorithm after certain number of iteration or when a channel change is detected,

Remarks:

- By using the symmetry properties of fourth-order cumulants only $\frac{(2M+1)^3}{24}$ cumulants need to be calculated.
- The assumption that I_{lag} data have been received at iteration (0) avoids ill conditioning of the matrices of the system given in Step 3. It causes a delay to I_{lag} at the input of the

equalizer.

Step 2

Select p, q arbitrarily large so that $A^{(I)} \simeq 0$ and $B^{(J)} \simeq 0$ for $I > p$ and $J > q$. For example, $C = 10^{-4}$ (very small constant)

$$\begin{aligned} A^{(I)} &\simeq 0 \quad \text{for} \quad I > p = int\left[\log \frac{C}{\alpha}\right] \\ B^{(J)} &\simeq 0 \quad \text{for} \quad J > p = int\left[\log \frac{C}{\beta}\right] \end{aligned} \tag{5.28}$$

where, $int[\cdot]$ denotes integer part and $max|a_i| < \alpha < 1$, $max|b_i| < \beta < 1$.

Define: $w = max(p, q), z \leq \frac{w}{2}, s \leq z.$

Step 3

Using the relation:

$$\begin{aligned} &\sum_{I=1}^{p} \left\{ A_{(i)}^{(I)} \left[\hat{L}_y^{(i)}(m-I, n, l) - \hat{L}_y^{(i)}(m+I, n+I, l+I) \right] \right\} + \\ &\sum_{J=1}^{q} \left\{ B_{(i)}^{(J)} \left[\hat{L}_y^{(i)}(m-J, n-J, l-J) - \hat{L}_y^{(i)}(m+J, n, l) \right] \right\} = -m \cdot \hat{L}_y^{(i)}(m, n, l) \end{aligned} \tag{5.29}$$

with $m = -w, \ldots, -1, 1, \ldots, w, n = -z, \ldots, 0, \ldots, z$ and $l = -s, \ldots, 0, \ldots, s$ to form the overdetermined system of equations:

$$\hat{P}(i) \cdot \hat{a}(i) = \hat{p}(i) \qquad i = 0, 1, 2, \ldots \tag{5.30}$$

where $\hat{P}(i)$ is $[N_p \times (p+q)]$ (where $N_p = 2w \times (2z+1) \times (2s+1)$) matrix with entries of the form $\{\hat{L}_y^{(i)}(m, n, l) - \hat{L}_y^{(i)}(\sigma, \tau, \lambda)\}$; $\hat{a}(i) = [\hat{A}_{(i)}^{(1)}, \ldots, \hat{A}_{(i)}^{(p)}, \hat{B}_{(i)}^{(1)}, \ldots, \hat{B}_{(i)}^{(q)}]^T$ (T denotes transpose)

is the $(p+q) \times 1$ vector of unknown cepstral parameters; $\hat{p}(i)$ is the $N_p \times 1$ vector with entries of the form $\{-m \cdot \hat{L}_y(m,n,l)\}$.

Step 4

Assume that initially $\hat{a}(0) = [0,\ldots,0]^T$. Update $\hat{a}(i) = [\hat{A}(1),\ldots,\hat{A}_{(i)}^{(p)}, \hat{B}(1),\ldots,\hat{B}_{(i)}^{(q)}]^T$ as follows

$$\hat{a}(i+1) = \hat{a}(i) + \mu(1) \cdot \hat{P}^H(i) \cdot \hat{e}(i), \tag{5.31}$$

$$\hat{e}(i+1) = \hat{p}(i) - \hat{P}(i) \cdot \hat{a}(i), \quad 0 < \mu(i) < 2/tr\{\hat{P}^H(i) \cdot \hat{P}(i)\} \tag{5.32}$$

Step 5

Calculate the equalizer normalized coefficients. Initialize $\hat{i}_{inv}(i,0) = \hat{o}_{inv}(i,0) = 1$ and the estimate:

$$\hat{i}_{inv}(i,k) = \frac{1}{k}\sum_{n=2}^{k+1}[\hat{A}_{(i)}^{(n-1)}] \cdot \hat{i}_{inv}(i,k-n+1)$$

$$k = 1,\ldots,N_1 \tag{5.33}$$

$$\hat{o}_{inv}(i,k) = \frac{1}{k}\sum_{n=k+1}[-\hat{B}_{(i)}^{(1-n)}] \cdot \hat{o}_{inv}(i,k-n+1)$$

$$k = -1,\ldots,-N_2 \tag{5.34}$$

where (i) is the iteration index taking values $i = 1,2,3\ldots$ Then,

$$\hat{u}_{norm}(i,k) = \hat{i}_{inv}(i,k) * \hat{o}_{inv}(i,k), k = -N_2,\ldots,0,\ldots,N_1 \tag{5.35}$$

where $\{*\}$ denotes linear convolution.

Step 6

Estimate the gain factor $\hat{A}(i)$ as follows: In step (1) we have already calculated:

$$\hat{L}_y^{(i)}(0,0,0) \simeq \gamma_x \cdot \sum_k (f(k))^4$$
$$\hat{M}_2^{(i)}(0) \simeq Q_x \cdot \sum_k (f(k))^2 \tag{5.36}$$

where $Q_x = E\{(x(k))^2\}$, $\gamma_x = E\{(x(k))^4\} - 3 \cdot Q_x^2$ are known. Also:

$$\begin{aligned} \hat{i}(i,k) &= -\frac{1}{k} \sum_{n=2}^{k+1} \hat{A}_{(i)}^{(n-1)} \cdot \hat{i}(i,k-n+1), \quad k = 1,\ldots,p \\ \hat{o}(i,k) &= \frac{1}{k} \sum_{n=k+1}^{0} \hat{B}_{(i)}^{(1-n)} \cdot \hat{o}(i,k-n+1), \quad k = -1,\ldots,q \end{aligned} \tag{5.37}$$

and $\hat{f}(i,k) = \hat{i}(i,k) * \hat{o}(i,k)$, $\{*\}$ denotes convolution, $Q_{\hat{f}}(i) = \sum_k (\hat{f}(i,k))^2$, $\gamma_{\hat{f}}(i) = \sum_k (\hat{f}(i,k))^4$.

Then (the sign of $\frac{1}{\hat{A}(i)}$ cannot be identified):

For L-PAM Signaling:

$$\left| \frac{1}{\hat{A}(i)} \right| \simeq \left(\frac{Q_x \cdot Q_{\hat{f}}(i)}{\hat{M}_2^{(i)}(0)} \right)^{\frac{1}{2}} \tag{5.38}$$

For L^2-QAM Signaling:

$$\frac{1}{\hat{A}(i)} \simeq \left(\frac{\gamma_x \gamma_{\hat{f}}(i)}{\hat{L}_y^{(i)}(0,0,0)} \right)^{\frac{1}{4}} = |\gamma_x|^{\frac{1}{4}} \cdot e^{j\frac{\pi}{4}} \cdot \left(\frac{\gamma_{\hat{f}}(i)}{\hat{L}_y^{(i)}(0,0,0)} \right)^{\frac{1}{4}} \tag{5.39}$$

since $\gamma_x < 0$ for equi-probable L^2-QAM signaling.

Step 7

Let, $\underline{y}(i) = [y(i+N_2),\ldots,y(i-N_1)]^T$ and $[\underline{\hat{u}}_{norm}(i)] = [\hat{u}_{norm}(i,-N_2),\ldots,\hat{u}_{norm}(i,N_1)]^T$. Fi-

nally, the output of the TEA equalizer is:

$$\tilde{x}(i) = \frac{1}{\hat{A}(i)} \cdot [\underline{\hat{u}}_{norm}(i)]^T \cdot \underline{y}(i) \tag{5.40}$$

While most of the Bussgang blind equalization algorithms, which are based on non-MSE cost function minimization, have not been shown to be globally convergent and cases of their mis-convergence have been encountered, the TEA algorithm, designed as described above, is a more reliable alternative, as it guarantees convergence.

Remarks:

1. Since Gaussian noise is suppressed in the fourth-order cumulant domain, the identification of the channel response does not take into account the observation noise. Consequently, the proposed equalizers work under the zero-forcing (ZF) constraint. For the same reason, we expect that the identification of the channel will be satisfactory even in low signal to noise (SNR) conditions.

2. The ability of the tricepstrum method to identify separately the maximum and minimum phase components of the channel makes possible the design and implementation of different equalization structures.

3. In the recursive formulas (5.37) we used the following properties that relate time impulse responses with cepstrum coefficients: (i) a channel and its inverse have opposite in sign cepstrum coefficients, (ii) the cepstrum coefficients of the convolution of two minimum phase or two maximum phase sequences, are equal to the sum of the corresponding cepstrum coefficients of the individual sequences and (iii) two finite impulse response (FIR) sequences with

conjugate roots have also conjugate cepstrum coefficients. These become unique features of the TEA equalizer when is compared with other equalization schemes.

4. The described algorithm is based only on the statistics of the received sequence $\{y(i)\}$ and does not take into account the decisions $\{\hat{x}(i)\}$ at the output of the equalizer. Consequently wrong decisions (and thus error propagation effects) do not affect the convergence of the proposed equalization schemes.

5. Instead of using the LMS algorithm to solve adaptively the system of equations (5.30), one may employ a Recursive Least-Squares (RLS) algorithm [25] which will have a faster convergence at the expense of even more computations.

5.2.4 Power Cepstrum and Tricoherence Equalization Algorithm (POTEA) [7]

5.2.5 Relations of Power Cepstrum and Tricoherence of the Linear Filter Output

The problem is as formulated in Section 5.2.1, the channel output $y(i)$ is the convolution of the non-Gaussian *i.i.d.* random sequence $x(i)$ with the channel impulse response $f(i)$ plus some noise. The cepstrum of the power spectrum of the channel output $y(i)$, can be shown after some algebra to be equal to [7].

$$c_{p_y}(m) = \begin{cases} ln|A_2| & m = 0 \\ -\frac{1}{m}[A^{*(m)} + B^{(m)}] & m > 0 \\ \frac{1}{m}[A^{(-m)} + B^{*(-m)}] & m < 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.41}$$

where $A^{(k)}$, $B^{(k)}$ are the minimum and maximum phase cepstral coefficients of $F(z)$. These are :

$$\begin{aligned} A^{(k)} &= \sum_{i=1}^{L_3} a_i^k - \sum_{i=1}^{L_4} c_i^k \\ B^{(k)} &= \sum_{i=1}^{L_2} b_i^k, \end{aligned} \tag{5.42}$$

where $\{a_i\}$ and $\{b_i\}$ are the zeros of $F(z)$ inside and outside of the unit circle respectively.

Remarks:

1. $A^{(k)}$, $B^{(k)}$ decay exponentially and thus their length can be truncated in practice at $k = p$, so that $A^{(p)}$, $B^{(p)}$ are arbitrarily small.

2. **If the channel $F(z)$ has cepstral coefficients $A^{(k)}$, $B^{(k)}$, its inverse filter, $U(z)$, has cepstral coefficients $-A^{(k)}$, $-B^{(k)}$.** It is also shown in [7] that if we define $S^{(k)} \triangleq A^{(k)} + B^{*(k)}$ and $r_y(k) \triangleq E\{y(i+k)y^*(i)\}$, then the following relations holds:

$$\sum_{k=1}^{p} S^{*(k)}[-r_y(m-k)] + \sum_{k=1}^{p} S^{(k)}[r_y(m+k)] = m r_y(m), \quad m = 1, \cdots 2p \tag{5.43}$$

where p is some integer, the choice of which is discussed in [24]. Now let us consider the cepstrum of the tricoherence.

$$R_y(z_1, z_2, z_3) = \left[\frac{S_y(z_1, z_2, z_3)}{S_y^*(z_1^{-1}, z_2^{-1}, z_3^{-1})} \right]^{\frac{1}{2}} \tag{5.44}$$

It has been shown that the trispectrum of the received data satisfies:

$$S_y(z_1, z_2, z_3) = \gamma_x F^*(z_1^{-1}) F(z_2) F^*(z_3^{-1}) F(z_1^{-1} z_2^{-1} z_3^{-1}) \tag{5.45}$$

Therefore,

$$R_y(z_1, z_2, z_3) = \left[\frac{F^*(z_1^{-1})F(z_2)F^*(z_3^{-1})F(z_1^{-1}z_2^{-1}z_3^{-1})}{F(z_1^{-1})F^*(z_2)F(z_3^{-1})F^*(z_1 z_2 z_3)} \right]^{\frac{1}{2}} \tag{5.46}$$

After some algebra, we obtain

$$R_y(m,n,l) = \frac{1}{2} \begin{cases} ln|A_1| & m=0, n=0, l=0 \\ -\frac{1}{m}[A^{*(m)} - B^{(m)}] & m>0, n=0, l=0 \\ -\frac{1}{m}[A^{*(-m)} - B^{*(-m)}] & m<0, n=0, l=0 \\ -\frac{1}{n}[A^{*(n)} - B^{*(n)}] & m=0, n>0, l=0 \\ -\frac{1}{n}[A^{*(-n)} - B^{*(-n)}] & m=0, n>0, l=0 \\ \frac{1}{m}[A^{*(m)} - B^{(m)}] & m=n=l>0 \\ \frac{1}{m}[A^{(-m)} - B^{*(-m)}] & m=n=l<0 \\ -\frac{1}{l}[A^{*(l)} - B^{(l)}] & m=0, n=0, l>0 \\ -\frac{1}{l}[A^{(-l)} - B^{*(-l)}] & m=0, n=0, l<0 \\ 0 & \text{otherwise} \end{cases} \tag{5.47}$$

Taking the logarithm of both sides of (5.44), we obtain,

$$R_y(z_1, z_2, z_3) = \frac{1}{2}\left[lnS_y(z_1, z_2, z_3) - lnS_y^*(z_1^{-1}, z_2^{-1}, z_3^{-1}) \right] \tag{5.48}$$

Differentiating with respect to Z_1 and performing inverse Z-transform, we obtain

$$\begin{aligned} &2L_y(m,n,l) * L_y^*(-m,-n,-l) * [-mR_y(m,n,l)] \\ &\quad = L_y^*(-m,-n,-l) * [-mL_y(m,n,l)] + L_y(m,n,l) * [mL_y^*(m,n,l)] \end{aligned} \tag{5.49}$$

By defining the following functions:

$$\begin{aligned}\theta_1(m,n,l) &\triangleq L_y^*(-m,-n,-l) * L_y(m,n,l)\\ \theta_2(m,n,l) &\triangleq L_y^*(-m,-n,-l) * mL_y(m,n,l)\end{aligned} \tag{5.50}$$

are combining (5.49) and (5.50), we obtain:

$$2\theta_1(m,n,l) * [mR_y(m,n,l)] = \theta_2(m,n,l) + \theta_2^*(-m,-n,-l) \tag{5.51}$$

Defining $D^{(k)} = A^{(k)} - B^{*(k)}$ and combining (5.47), we obtain:

$$\begin{aligned}&\sum_{k-=1}^{p} D^{*(k)}[\theta_1(m-k,n-k,l-k) - \theta_1(m-k,n,l)]\\ &\quad + \sum_{k-=1}^{p} D^{(k)}[\theta_1(m+k,n+k,l+k) - \theta_1(m+k,n,l)]\\ &\quad = \theta_2(m,n,l) + \theta_2^*(-m,-n,-l)\end{aligned} \tag{5.52}$$

A rule of thumb is to define $w = p$, $z \leq w/2$, $h \leq z$ and then take $m = -w,\ldots,-1,1,\ldots,w$, $n = -z,\ldots z$, $l = -h,\ldots,h$ to form a linear overdetermined system to equations.

5.3 The POTEA Algorithm

In this section the POTEA algorithm is given in detail.

Let

N_1, N_2: Lengths of minimum and maximum phase components of the equalizer.

p: Length minimum and maximum phase cepstral parameters,

At iteration $i = 1,2,\ldots$.

Step 1 Estimate adaptively the $L_y^{(1)}(m,n,l)$ for $-M_1 \leq m,n,l \leq M_1$, and $r_y^{(i)}(m)$ for $-M_2 \leq m \leq M_2$ from a finite length window of $\{y(n)\}$, and then generate the following functions:

$$\begin{aligned}\theta_1^{(i)}(m,n,l) &= L_y^{*(i)}(-m,-n,-l) * L_y^{(i)}(m,n,l)\\ \theta_2^{(i)}(m,n,l) &= L_y^{*(i)}(-m,-n,-l) * mL_y^{(i)}(m,n,l)\end{aligned}$$

Step 2 Choose p arbitrarily such that $A^{(p+1)} \approx 0$, $B^{(p+1)} \approx 0$ and define $w = p$, $z \leq \frac{w}{2}$, $h \leq z$.

Step 3 Form the equations

$$\sum_{k=1}^{p} S^{*(k)}[-r_y(m-k)] + \sum_{k=1}^{p} S^{(k)}[r_y(m+k)] = mr_y(m), \quad m = 1,\cdots,2p \tag{5.53}$$

where $S^{(k)} = A^{(k)} + B^{*(k)}$, $k = 1,\ldots,p$.

$$\begin{aligned}&\sum_{k=1}^{p} D^{*(k)}[\theta_1(m-k,n-k,l-k) - \theta_1(m-k,n,l)]\\ &+\sum_{k=1}^{p} D^{(k)}[\theta_1(m+k,n+k,l+k) - \theta_1(m+k,n,l)]\\ &\qquad = \theta_2(m,n,l) + \theta_2^*(-m,-n,-l)\end{aligned} \tag{5.54}$$

and the following system of equations

$$\hat{P}\hat{a} = \hat{p} \tag{5.55}$$

$$\hat{Q}\hat{b} = \hat{q} \tag{5.56}$$

where the matrices $\hat{P}, \hat{a}, \hat{p}, \hat{Q}, \hat{b}$ and $\hat{q}$ are defined above.

Step 4 Solve adaptively the above systems employing LMS-type adaptation as follows:

$$\hat{a}(i+1) = \hat{a}(i) + \mu(i)\hat{P}^H(i)\hat{e}(i) \tag{5.57}$$

$$\hat{b}(i+1) = \hat{b}(i) + \mu'(i)\hat{Q}^H(i)\hat{e}'(i) \tag{5.58}$$

where

$$\begin{aligned} \hat{e}(i) &= \hat{p}(i) - \hat{P}(i)\hat{a}(i) \\ \hat{e}'(i) &= \hat{q}(i) - \hat{Q}(i)\hat{b}(i) \\ 0 &< \mu(i) < \frac{2}{tr(\hat{P}^H\hat{P})} \\ 0 &< \mu'(i) < \frac{2}{tr(\hat{Q}^H\hat{Q})} \end{aligned}$$

The algorithm at instant i minimizes the mean square error:

$$\begin{aligned} \hat{J}(i) &= E\{\hat{e}^H(i)\hat{e}(i)\} \\ \hat{J}'(i) &= E\{\hat{e}'^H(i)\hat{e}'(i)\} \end{aligned}$$

Step 5 Calculate $A^{(k)}$ and $B^{(k)}$ as follows:

$$A^{(k)} = \frac{S^{(k)} + D^{(k)}}{2}, B^{(k)} = (\frac{S^{(k)} - D^{(k)}}{2})^* \tag{5.59}$$

Step 6 Calculate

$$\hat{i}_{eq}(i,k) = \frac{1}{k}\sum_{n=2}^{k+1}[A_{(i)}^{(n-1)}]\hat{i}_{eq}(i,k-n+1), k = 1,\ldots,N_1 \tag{5.60}$$

$$\hat{o}_{eq}(i,k) = \frac{1}{k}\sum_{n=k+1}^{0}[-B_{(i)}^{(1-n)}]\hat{o}_{eq}(i,k-n+1), k=-1,\ldots,-N_2 \qquad (5.61)$$

with initialization : $\hat{i}_{eq}(i,0) = \hat{o}_{eq}(i,0) = 1$. The normalized $(A=1)$ estimate $\hat{u}_{norm}(i,k)$ at iteration (i) is given by:

$$\hat{u}_{norm}(i,k) = \hat{i}_{eq}(i,k) * \hat{o}_{eq}(i,k) \qquad (5.62)$$

Step 7 Estimate gain factor $\hat{A}(i)$

Step 8 The reconstructed transmitted sequence at iteration(i) is:

$$\tilde{x}(i) = \frac{1}{\hat{A}(i)}\sum_{k=-N_2}^{N_1}\hat{u}_{norm}(i,k)y(i-k) \qquad (5.63)$$

Computational Complexity

In this section the computational complexity of POTEA is presented and compared with the computational complexity of TEA.

PAM

POTEA: $\frac{3(2M+1)^3}{8} + 3(2M+1) + 2p(N_p+p+1) + \frac{N^2+8N+3}{4} + (4M)^3\log_2 4M$

TEA: $\frac{3(2M+1)^3}{8} + 3(2M+1) + (p+q)(2N_p+1) + \frac{N^2+8N+3}{4}$

QAM

POTEA: $4[\frac{3(2M+1)^3}{8} + 2(2M+1) + 2p(2N_p+4p+2) + \frac{N^2+8N+3}{4} + (4M)^3\log_2 4M]$

TEA: $4[\frac{(2M+1)^3}{6} + (p+q)(2N_p+1) + \frac{N^2+8N+3}{4}]$

5.4 Cross-Tricepstrum Equalization Algorithm (CTEA) [8]

5.4.1 Problem Formulations

Assume we have n measurements at each time index k, $y_i(k), i = 1, 2, \ldots n$, where

$$y_i(k) = f_i(k) * x(k) + n_i(k) \tag{5.64}$$

(shown in Figure 5.1 for $n = 4$) and

1. $f_i(k)$ is the impulse response of a discrete time linear time invariant system,
2. $x(k)$ is a non-Gaussian, nth order white process with cumulant $\gamma_x \neq 0$,
3. $n_i(k)$ is zero-mean additive noise, with $n_i(k)$ independent of $n_j(k)$ for $i \neq j$ and independent of $x(k)$. No assumptions are made about *pdf* for whiteness (in time) of $n_i(k)$.

We also assume that each impulse response $h_i(k)$ is stable with no zeros on the unit circle and that its Z transform $F_i(z)$ can be written as [8]

$$F_i(z) = A_i I_i(z^{-1}) O_i(z) \tag{5.65}$$

where the A_i are gain constants, the r_i are integer linear phase factors,

$$I_i(z^{-1}) = \frac{\prod_{j=1}^{L_{i3}}(1 - a_{ij}z^{-1})}{\prod_{j=1}^{L_{i4}}(1 - c_{ij}z^{-1})}$$

is the minimum phase component and

$$O_i(z) = \prod_{j=1}^{L_{i2}}(1 - b_{ij}z)$$

is the maximum phase component, with zeros a_{ij} and poles c_{ij} inside and zeros b_{ij} outside the unit circle (*i.e.* $|a_{ij}| < 1, |b_{ij}| < 1$, and $|c_{ij}| < 1$).

5.4.2 Relation of Cross-Tricepstrum of the Linear Filter Output

With the above assumptions, the nth-order cross-spectrum of the $y_i(k)$ can be written as

$$S_{y,1,2,\ldots,n}(z_1, z_2, \cdots, z_{n-1}) = \gamma_x F_1(z_1) F_2(z_2) \cdots F_{n-1}(z_{n-1}) F_n(\prod_{i=1}^{n-1} z_i^{-1}) \tag{5.66}$$

Taking the logarithm and performing inverse Z-transform on both sides, we obtain after some algebra the following results:

$$c_{y,1,2,\cdots,n}(m_1, m_2, \cdots, m_{n-1}) = \begin{cases} ln\gamma_x & m_1 = m_2 = \ldots = m_{n-1} = 0, \\ -(1/m_i)A_i(m_i) & m_i > 0, m_j = 0, j \leq i, \\ & i = 1, 2, \ldots, n-1, \\ (1/m_i)B_i(-m_i) & m_i < 0, m_j = 0, j \neq i, \\ & i = 1, 2, \ldots, n-1, \\ -(1/m_n)A_n(-m_n) & m_1 = m_2 = \ldots = m_{n-1} < 0, \\ -(1/m_n)B_n(m_n) & m_1 = m_2 = \ldots = m_{n-1} > 0, \\ 0 & \text{otherwise} \end{cases} \tag{5.67}$$

with

$$\begin{aligned} A_i(k) &\triangleq \sum_{j=1}^{L_{i3}} (a_{ij})^k - \sum_{j=1}^{L_{i4}} (c_{ij})^k \\ B_i(k) &\triangleq \sum_{j=1}^{L_{i2}} (b_{ij})^k. \end{aligned} \tag{5.68}$$

This results means that the n-th order cross-cepstrum is non-zero on n lines only in its domain and that on each of these lines we find the complex cepstrum of a zero-linear phase, scaled version of *one* of the n impulse responses.

Now, to develop a least squares solution for the A_i and B_i, we take first partial derivatives of the logarithm of (5.66), independently with respect to each of its variables, followed by inverse $\mathcal{Z}$ transforms. Letting $S_{y,1,2,\ldots,n}(m_1, m_2, \ldots, m_{n-1})$ denote the n-th order cross cumulants of the y_i, we get the following $n-1$ equations relating the cross cumulants to the cepstral coefficients:

$$\begin{aligned} & S_{y,1,2,\ldots n}(m_1, m_2, \ldots, m_n) * (m_i \; c_{y,1,2,\ldots,n}(m_1, m_2, \ldots, m_{n-1})) \\ & = -m_i \; S_{y,1,2,\ldots n}(m_1, m_2, \ldots, m_{n-1}) \end{aligned}$$

for $i = 1, 2, \ldots n-1$. Each equations involves an $(n-1)$ dimensional convolution. However, plugging in (5.67) reduces each equation to a single finite summation:

$$\begin{aligned} & \sum_{k=1}^{\infty} A_i(k) S_{y,1,2,\ldots,n}(t_1, t_2, \ldots, t_{n-1}) - B_i(k) S_{y,1,2,\ldots,n}(u_1, u_2, \ldots, u_{n-1}) \\ & \quad -A_n(k) S_{y,1,2,\ldots,n}(m_i + k, m_i + k, \ldots, m_i + k) \\ & \quad +B_n(k) S_{y,1,2,\ldots,n}(m_i - k, m_i - k, \ldots, m_i - k) \\ & \quad = -m_i S_{y,1,2,\ldots,n}(m_1, m_2, \ldots, m_{n-1}) \end{aligned} \tag{5.69}$$

where

$$\begin{aligned} t_i &= m_i - k \\ u_i &= m_i + k \\ t_j &= u_j = m_j \quad j \neq i. \end{aligned}$$

From equation (5.68) the sums in (5.69) decay, so we can truncate them to p_i and q_i for the terms involving A_i and B_i respectively (see [8]) and rewrite (5.69) as a finite dimensional vector dot product equation. Writing $M > p_i + q_i + p_n + q_n$ equations at M points in the $n-1$ dimensional domain of $S_{y,1,2,\ldots n}$ we can form the overdetermined system

$$\underline{R}_{in} \cdot \underline{C}_{in} = \underline{\gamma}_{in} \tag{5.70}$$

5.4.3 Cross-TEA (CTEA) Algorithm

In this section we describe the CTEA algorithm for blind equalization of QAM signals with four receivers. The algorithm has two stages at each iteration:

1. Channel identification and deconvolution
2. Combining by use of a decision rule

Channel Identification and Deconvolution

Step 1. Estimate the cross-cumulants and kurtoses of the received data recursively.

Step 2. Form the systems of equations (5.70) and solve each system in turn to get the cepstral coefficients for each channel[1]

Step 3. From the results of the previous step, estimate the forward and inverse channel impulse responses up to a desired length.

Step 4. From the estimated forward impulse response and the kurtoses, estimate the gains $A_i^{(j)}$ for each channel.

[1]The cepstral coefficients for channel four can be estimated from the solution of one of the three systems or an average of all three.

Step 5. With the estimated inverse response, $f_{i,\text{inv}}^{(j)}(k)$, and the estimated gain for each channel, deconvolve to estimate the input symbol as

$$\hat{x}_i(j) = \frac{1}{A_i^{(j)}} y_i(j) * f_{i,\text{inv}}^{(j)}(k)$$

Combining Decision Rules

As illustrated in Figure 5.1, from the four estimates $\tilde{x}_i(j)$ we need to form a single quantized decisions $\hat{x}(j)$. We describe here an optimal combining rule in the case of a perfect equalizer, as well as three sub-optimal schemes, arithmetic mean, majority rule, and median (which for $n = 4$ channels is equivalent to α-trimmed mean with $\alpha = 1$).

Optimal Decision for the Perfect Equalizer [8]

We consider the following assumptions:

1. $x(k)$ is complex and uniformly distributed,
2. $u_i(k)$ is the perfect equalizer for $f_i(k)$, *i.e.* $f_i(k) * u_i(k) = \delta(k)$, and
3. $n_i(k)$ are zero-mean, complex Gaussian variables with known variance σ_i^2 and are independent across channels.,

Since we will do symbol by symbol detection, we will drop the time index k for simplicity. With these assumptions,

$$\tilde{x}_i = x + n_i * u_i \triangleq x + \tilde{n}_i.$$

Therefore, the conditional probability density of $\hat{x}$ given X, $p(\hat{x}|x)$, is complex Gaussian with mean x and variance

$$\tilde{\sigma}_i^2 = \sigma_i^2 \sum_k |u_i(k)|^2.$$

Since the noise in each channel is independent, the maximum likelihood estimate $\hat{x}$ of x given the four observations $\tilde{x}_i$ (assuming x to be from a continuous distribution) is

$$\begin{aligned} \hat{x}_R &= \frac{\frac{1}{4}\sum_i \tilde{\sigma}_i^{-2}\tilde{x}_{i,R}}{\sum_i \tilde{\sigma}_i^{-2}} \\ \hat{x}_I &= \frac{\frac{1}{4}\sum_i \tilde{\sigma}_i^{-2}\tilde{x}_{i,I}}{\sum_i \tilde{\sigma}_i^{-2}} \end{aligned}$$

where the subscript R and I denote real and imaginary parts respectively. Note that if the noise has the same variance in all channels then this result reduces to the arithmetic man. If, on the other hand, we assume that x belongs to a known discrete set $\mathcal{D}$ then we need to find $\hat{x} \in \mathcal{D}$ which satisfies

$$\min_{\hat{x}\in\mathcal{D}} \sum_i \tilde{\sigma}_i^{-2}|\tilde{x}_i - \hat{x}|^2$$

or equivalently

$$\min_{\hat{x}\in\mathcal{D}} \sum_i \tilde{\sigma}_i^{-2}\left(|\hat{x}|^2 - 2(\hat{x}_R\tilde{x}_{i,R} + \tilde{x}_I\hat{x}_{i,I})\right).$$

Of course the assumptions of perfect equalization and known noise variance are not realistic in practice so we describe below three sub-optimal combining rules which we tested in our simulations.

Arithmetic Mean

Step 1. Form a soft decision statistic

$$\tilde{x}(j) = \frac{1}{4}\sum_{i=1}^{4} \tilde{x}_i(j).$$

(If information is available about the relative quality of the channels then a weighted mean could be used.)

Step 2. Put $\tilde{x}(j)$ through a decision device to get $\hat{x}(j)$.

Majority Rule

Step 1. Put each estimate through a decision device to form four decision statistics $\hat{x}_i(j)$.

Step 2. If there is a plurality among the $\hat{x}_i(j)$ in one region of the decision space then that is the decision. If there is a tie (all four different or two votes for each of two decisions) use a tie-breaking procedure. One method would be to pick the decision region that has the smallest average squared decision error. For example, if $\hat{x}_i(j) = \hat{x}_2(j) \neq \hat{x}_3(j) = \hat{x}_4(j)$:

$$\text{Let } d_1 = \sum_{i=1}^{2} |\hat{x}_i(j) - \tilde{x}_i(j)|^2$$

$$\text{Let } d_2 = \sum_{i=3}^{4} |\hat{x}_i(j) - \tilde{x}_i(j)|^2$$

Then

$$\text{Choose} \quad \hat{x}_1(j) \quad d_1 \leq d_2$$

$$\hat{x}_2(j) \quad d_2 > d_1.$$

Median

Step 1. Order the real and imaginary parts of the $\tilde{x}_i(j)$ separately.

Step 2. Set

$$\begin{aligned} \text{REAL}\{\tilde{x}(j)\} &= \text{median}\{\text{REAL}\{\tilde{x}_i(j)\}\} \\ \text{IMAG}\{\tilde{x}(j)\} &= \text{median}\{\text{IMAG}\{\tilde{x}_i(j)\}\} \end{aligned}$$

Step 3. Put $\tilde{x}(j)$ through the decision to get $\hat{x}(j)$.

5.5 Computer Simulations

Computer simulations has been employed to compare the performance of the blind equalization algorithms. The performance metric used are those in Sections 2. And the following issues are addressed.

5.5.1 TEA vs. Bussgang-type Algorithms

Fig. 5.2-5.4 show the performance of the TEA algorithm, compared with that of Bussgang-type algorithms, such as Godard, Benveniste-Goursat, Stop-and-Go algorithms. We see that the TEA algorithm opens the eye much faster than the Bussgang-type algorithms. This performance improvement is achieved at the expense of larger computational complexity.

5.5.2 POTEA vs. TEA

Fig. 5.5-5.6 show the performance of the POTEA algorithm, compared with that of TEA. We see that the POTEA algorithm converges faster than the TEA algorithm. The performance improvement is achieved at the expense of further increase in computational complexity.

5.5.3 CTEA vs. TEA

Fig. 5.7-5.8 show the performance of the CTEA algorithm compared with that of TEA algorithm. We see that the CTEA algorithm converges faster than the TEA algorithm for some channels. The performance improvement is achieved at the expense of further increase in computational complexity.

6 ALGORITHM WITH NONLINEARITY INSIDE THE EQUALIZATION FILTER

Still another class of bind equalization algorithms are those algorithms which use Volterra filters [9], [10] or neural networks [20], [26], [27]. This class of algorithms perform nonlinear operations inside the equalization filter. It is therefore also be able to correctly extract the phase information of the unknown channel from its output only. In this section, we will concentrate on those algorithms based on neural network.

6.1 Review of Equalization Techniques Based on Neural Networks

Equalization is a technique which is used to combat the intersymbol interference caused by non-ideal channels. Usually, equalizers are implemented using linear transversal filters [17], [18], [30], [31]. However, when the unknown channel has deep spectral nulls or some severe nonlinear distortions, such as phase jitter and frequency offset, linear equalizers are not powerful enough to compensate all of these. That is why nonlinear filters, such as those implemented by Volterra filter or neural network, come in and play an important role.

Neural Networks (NNWs) are mathematical models of theorized mind and brain activities. The fundamental idea of NNWs is to organize many simple identical processing elements into

layers to perform more sophisticated tasks. The properties of NNWs include: massive parallelism; high computation rates; great capability for non-linear problems, continuous adaptation; inherent fault tolerance and ease for VLSI implementation, etc. All these properties make NNWs attractive to various applications. Several neural network based algorithms have been proposed for equalization problems.

1 Multi-Layer Perceptron

The multi-layer Perceptron (MLP) [39], [40] is one of the most widely used implementations of NNWs. It comprises a number of nodes which are arranged in layers, as shown in Figure 6.1. A node receives a number of inputs $x_1, x_2, \cdots, x_n$, which are then multiplied by a set of weights $w_1, w_2, \cdots, w_n$ and the resultant values are summed up. A constant v is added to this weighted sum of inputs, known as the node threshold, and the output of the node is obtained by evaluating a nonlinear (sigmoid) function, $f(.)$, which is called activation function.

The architecture of a perceptron can be described by a sequence of integers $n_0, n_2, \cdots, n_k$. where n_0 is the dimension of the input to the network, and the number of nodes in each successive layer, ordered from input to output, is $n_1, n_2, \cdots, n_k$. In this notation, the MLP produces a nonlinear mapping $g = R^{n_0} \rightarrow R^{n_k}$.

The updating of the connection coefficients of the MLP is done iteratively by using back-propagation (BP) algorithm with the following formula:

$$(w_{i+1}, v_{i+1}) = (w_i, v_i) + \Delta_i \tag{6.1}$$

and

$$\Delta_i = -(\beta, \eta) \cdot \frac{de^2}{d(w_i, v_i)} + \alpha \cdot \Delta_{i-1}. \tag{6.2}$$

2 Self-Organizing Feature Maps

The topology by self-organizing feature map (SOM), which is introduced by Kohonen [26], [27] consists of two layers of nodes, referred to as input layer and output layer, which are fully connected with different connection weights. The inputs to the SOM can be any continuous values, whereas each of the output-layer node represent a pattern class that the input vector may belong to. That means the outputs of SOMs are discrete values, and therefore, the SOM is sometimes also referred to as learning vector quantizer.

The SOM works iteratively as follows. First, find the set of connection coefficients W_g which is the closest to the input vector A_k,

$$\| A_k - W_g \| = \min_{j=1}^{p} \| A_k - W_j \| . \tag{6.3}$$

Second, perform the following quantization of the output-layer node:

$$b_g = \begin{cases} 1, & \text{if } \| A_k - W_g \| = \min \| A_k - W_j \| \\ 0, & \text{otherwise.} \end{cases} \tag{6.4}$$

and then move W_g closer to A_k using the equation

$$\Delta W_{ij} = \begin{cases} \alpha(k) \cdot [a_i^k - W_{ig}], & j = g \\ \beta(k) \cdot [a_i^k - W_{ij}], & j \in N_r, j \neq g \\ 0, & j \notin N_r, \end{cases} \tag{6.5}$$

where N_r is the topological neighborhood of the winning node b_g which consists b_g itself and its direct neighbors up to the depth $1, 2, \cdots$, and $\alpha(k)$ and $\beta(k)$ are the learning rate at time k.

6.2 The MLPs Equalization Algorithm for PAM and QAM Signals

The applications of MLP in equalization problems so far, have been limited to binary $\{0, 1\}$ or bipolar $\{-1, 1\}$ valued data and real valued channel models [11], [20], [49]. In this section, we introduce for the first time a **new implementation structure** of MLP which works well with L-PAM ($L > 2$) and N-QAM (N¿4) signals.

Looking into a MLP structure, we find out that it is the sigmoid function of the output layer nodes that confines the network outputs to the range $[-1, 1]$. In our equalization problem, the signals are equally spaced and symmetric with respect to either the original point of the coordinate, or to the x and y axes. Thus **we can just scale up the node function of the output layer by a constant factor C which is large enough to cover our maximum signal range**, e.g., [-15,15] for 16-PAM or 256-QAM signals. So, for the output layer, we have [30], [40]

$$f_M(x) = C \cdot \frac{1 - e^{\alpha x}}{1 + e^{\alpha x}}, \qquad (c \geq 1) \tag{6.6}$$

as the activation function. For the hidden layers, we still use the sigmoid function

$$f_i(x) = \frac{1 - e^{\alpha x}}{1 + e^{\alpha x}}. \tag{6.7}$$

The idea of adding another constant α comes form the thought that a smaller α, equivalently, a lower slope in Figure 6.2, would avoid high vibration, and in turn, decrease the chance of

divergence in the course of weight adjustment.

For complex channel models and QAM signals, we use complex connection coefficients to get the weighted sum to which a complex threshold is added. Then the sigmoid functions of the real and the imaginary parts of the threshold added weighted sum are evaluated separately. Again, for the output layer nodes, the outputs are multiplied by a constant C. Using the steepest descent formula (Eq. 6.1, 6.2), we get the adaptation algorithm of our new MLP equalizer which is described in Table 6.1 [30], [40].

Simulation are conducted to examine the performance of MLP equalizers. The equalizer is implemented by the new MLP structure with only one output node. The input data to the system x_i are assumed to be independent of each other. The delayed input sequence x_{i-d}, where d is channel dependent, is used as the training sequence. The performance of MLP equalizers is evaluated by calculating the mean square error (MSE) $E[(x-\tilde{x})^2]$ and the average symbol error rate (SER) of the quantizer output. The eye pattern of equalizer outputs around certain number of iterations is shown in Figure 6.3.

Figure 6.4 illustrates the performance comparison between MLP and LMS-based linear transversal equalizer with the same number of inputs. The structure (the number of nodes in the hidden layer) of the MLP has been fine-tuned through experiment. The step size μ of the LMS-based equalizer is also optimized (the biggest value without causing divergence). From Fig. 6, it appears that the new structure of MLP works no much better, as a channel equalizer, than the simple linear adaptive equalizer. As a matter of fact, both methods end giving similar results.

7 CONCLUSIONS

The purpose of this paper is to provide a tutorial review of existing blind equalization algorithms for digital communications. Three families of techniques have been described, namely, the Bussgang techniques, the polyspectra-based techniques, and methods based on nonlinear equalization filters or neural networks. The complexity of the Bussgang techniques is approximately $2N$ multiplications per iteration, where N is the order of the linear equalization filter. On the other hand, the polyspectra-based techniques require approximately $\frac{1}{2}N^3$ multiplications per iteration. However, as it has been demonstrated in the paper, the polyspectra-based techniques achieve significantly faster convergence rate than the Bussgang techniques. Finally, it is pointed out in the paper that blind equalizers based on nonlinear filters or neural networks are better suited for equalization of channels with nonlinear distortions.

List of Figures

List of Tables

REFERENCES

[1] Austin, M.E., "Decision-Feedback Equalization for Digital Communication over Dispersive Channels," *IEEE Trans. Comm.*, Vol. COM-30, pp. 2421-2433, November 1982.

[2] Belfiori, C.A. and J.H. Park, Dr., "Decision-Feedback Equalization," *Proc. IEEE*, Vol 67, pp; 1143-1156, August 1979.

[3] Bellini, S. "Bussgang Techniques for Blind Equalization," *Proc. IEEE-Globecom'86*, pp. 46.1.1-46.1.7.

[4] Bellini, S. and F. Rocca, "Blind Equalization: Polyspectra or Bussgang Technbiques," in Digital Communications, E. Biglieri and G. Prati Eds., Northholland, 1986, pp. 251-262.

[5] Benveniste, A. and M. Goursat, "Blind Equalizers," *IEEE Trans. on Communications*, Vol. COM-32, pp. 871-883, August 1984.

[6] Benveniste, S., M. Goursat and G. Ruget, "Robust Identification of Nonminimum Phase System: Blind Adjustment of a Linear Equalizer in Data Communications," *IEEE Trans. Automat. Contr.*, Vol. AC-25, No. 3, pp. 385-398, 1980.

[7] Bessios, A.G. and C.L. Nikias, "POTEA: The Power Cepstrum and Tricoherence Equalization Algorithm," *Proc. SPIE*, San Diego, CA July 1991.

[8] Brooks, D.H. and C.L. Nikias, "Cross-Bicepstrum and Cross-Tricepstrum Approaches to Multichannel Deconvolution," *Proc. Int. Signal Processing Workshop in Higher-Order Statistics*, Chamrousse, France, July 1991.

[9] Biglieri, S. Barberis, and M. Catena, "Analysis and Compensation of Nonlinearities in Digital Transmission Systems," *IEEE Sel. Areas in Comm.*, Vol. 6, No. 1, pp. 42-51,

January 1988.

[10] Biglieri, E., A. Gersho, R.D. Gitlin, and T.L. Lim, "Adaptive Cancellation of Nonlinear Symbol Interference for Voiceband Data Transmission," *IEEE Sel. Areas in Comm.*, Vol. SAC-2, No. 5, September 1984.

[10] Bussgang, J.J., "Crosscorrelation Functions of Amplitude-Distorted Gaussian Signals", *MIT Technical Report*, No. 216, March 1952.

[11] Chen, S., G.J. Gibson, C.F.N. Cowan , and P.M. Grant, "Adaptive Equalization of Finite Non-Linear Channels Using Multilayer Perceptrons," *Signal Processing*, 20, pp. 107-109, 1990.

[12] Chen, Y., C.L. Nikias and J.G. Proakis, "CRIMNO: Criterion with Memory Nonlinearity for Blind Equalization," *Proc. Intern. Signal Processing Workshop in Higher-Order Statistics*, Chamrousse, France, July 1991.

[13] Chen, Y., C.L. Nikias and J.G. Proakis, "CRIMNO: Criterion with Memory Nonlinearity for Blind Equalization," *Proc. 25th Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, Nov. 1991.

[14] Chiang, H.H. and C.L. Nikias, "Deconvolution and Identification of Nonminimum Phase FIR Systems Based on Cumulants," *IEEE Trans. Automatic Control*, Vol. AC-35, No. 1, January 1990.

[15] Ding, Z., R.A. Kennedy, B.D.O. Anderson, and C.R. Johnson, "Existence and Avoidance of Ill-convergence of Godard Blind Equalizers in Data Communication Systems," 23rd Conference on Information Sciences and Systems, Baltimore, MD, March 1989.

[16] Donoho, D.L., "On Minimum entropy Deconvolution," in D.F. Findley, (ed.), *Applied Time Series Analysis, II*, Academic Press, NY, 1981.

[17] Forney, G.D., Jr., "Maximum-Likelihood Sequence Estimation of Digital Sequences in the Presence on Intersymbol Interferences," *IEEE Trans. Inform. Theory*, Vol. IT-18, pp. 363-378, May 1972.

[18] Foschini, G.J., "Equalizing without Altering or Detecting Data," *AT&T Technical J.*, Vol. 64, No. 8, pp. 1885-1909, October 1985.

[19] Gersho, A. and T.L. Lim, "Adaptive Cancellation of Intersymbol interference for Data Transmission," *The Bell Syst. Tech. Journal*, pp. 1997-2021, November 1981.

[20] Gibson, G.J., S. Siu, and C.F.N. Cowan, "Application of Multilayer Perceptrons as Adaptive Channel Equalizers," *Proceedings IEEE Internatioal Conference ASSP*, Glasgow, Scotland, May 23-26, 1989, pp. 1183-1186.

[21] Gitlin, R.D. and S.B. Weinstein, "Fractionally-Spaced Equalization: An Improved Digital Transversal Equalizer," Bell Syst. Tech. J., Vol. 60, pp. 275-296, February 1981.

[22] Godard, D.N., "Self-Recovering Equalization and Carrier Tracking in Two-Dimensional Data Communication Systems," *IEEE Trans. on Communications*, Vol. COM-28, pp. 1867-1875, November 1980.

[23] Godfrey, R. and F. Rocca, "Zero Memory Nonlinear Deconvolution," *Geophysical Prospecting*, Vol. 29, pp. 189-228, 1981.

[24] Hatzinakos, D. and C.L, Nikias, "Blind Equalization using a Tricepstrum Based Algorithm," *IEEE Trans. Comm.*, Vol. COM-39, May 1991.

[25] Haykin, S., Adaptive Filter Theory, Prentice Hall, Inc., Englewood Cliffs, NJ, 1991.

[26] Kohonen, T., K. Raivio, O. Simula, "Combining Linear Equalization and Self-Organizing Adaptation in Dynamic Discrete-Signal Detection," *IJCNN-90*, Vol. 1, San Diego, CA, 1990.

[27] Kohonen, T., "Self-Organization and Associative Memory," Series in Information Sciences, Vol. 8, Springer-Verlag, 3rd Edition, 1989.

[28] Kolmogorov, A.N., "On the Representation of Continuous Functions of Many Variables by Superposition of Continuous Functions of One Variable and Addition," *Dokl. Adad. Nauka USSR*, Vol. 144, pp. 953-956. 1957.

[29] Lii, K.S. and M. Rosenblatt, "Deconvolution and Estimation of Transfer Function Phase and Coefficients for Non-Gaussian Linear Processes," *Ann. Statistics*, Vol. 10, pp. 1195-1208, 1982.

[30] Lucky, R.W., "Automatic Equalization for Digital Communications," *Bell Systems Tech. J.*, Vol. 44, pp. 547-588, 1965.

[31] Lucky, R.W., "Techniques for Adpative Equalization for Digital Communications," *Bell Systems Tech. J.*, Vol. 45, pp. 555-286.

[32] Macchi, O. and E. Eweda, "Convergence Analysis of Self-Adaptive Equalizers," *IEEE Trans. Inform. Theory*, Vol. IT-30, pp. 161-176, March 1984.

[33] Mazo, J.E., "Analysis of Decision-Directed Equalizer Convergence," *The Bell Systems Tech. J.*, Vol. 59, pp. 1857-1876, 1980.

[34] Mendel, J.M., "Tutorial on Higher-Order Statistics (Spectra) in Signal Processing and System Theory: Theoretical Results and Some Applications," *Proceedings IEEE*, Vol. 9, pp. 278-305, March 1991.

[35] Nikias, C.L., "ARMA Bispectrum Approach to Nonminimum Phase System Identification," *IEEE Trans. ASSP*, Vol. 36, No. 4, pp. 513-525, April 1988.

[36] Nikias, C.L., "Blind Deconvolution using Higher-Order Statistics," *Proc. International Workshop on Higher-Order Statistics*, France, July 1991.

[37] Nikias, C.L., and H.H. Chiang, "Higher-Order Spectrum Estimation via Noncausal Autoregressive Modeling and Deconvolution," *IEEE Transactions ASSP*,Vol. 36, No. 12, pp. 1911-1914, December 1988.

[38] Nikias, C.L. and M.R. Raghuveer, "Bispectrum Estimation: A Digital Signal Processing Framework," *Proc. IEEE*, Vol. 75, No. 7, pp. 869-891, July 1987.

[39] Peng, M., C.L. Nikias and J. Proakis, "Adaptive Equalization for PAM and QAM Signals with Neural Networks," *Proc. 25th Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, Nov. 1991.

[40] Peng, M., C.L. Nikias and J. Proakis, "Adaptive Equalization with Neural Networks : New Multi-layer Perception Structures and their Evaluation," *ICASSP'92*, San Francisco, CA, March 1992.

[41] Picchi, G. and G. Prati, "Blind Equalization and Carrier Recovery Using a 'Stop-and-Go' Decision-Directed Algorithm," *IEEE Trans. on Communications*, Vol. COM-35, No. 9, pp. 877-887, September 1987.

[42] Porat, B. and B. Friedlander, "A Blind SAG-SO-DFD-FS Equalizer," *Proc. of the IEEE Globecom'88*, pp. 30.4.1-30.4.5.

[43] Proakis, J.G., "Advances in Equalization for Intersymbol Interference," on *Advances in Communication Systems*, Vol. 4, A.J. Viterbi (ed.), Academic Press, New York, 1975.

[44] Proakis, J.G.,"Digital Communications," McGraw-Hill, New York, 1989, Second Edition.

[45] Qureshi, S.U.H. and G.D. Forney, Jr., "Performance Properties of a T/2 Equalizer," pp. 11.1.1-11.1.14, Los Angeles, CA, December 1977.

[46] Sato, Y., et al., "Blind Suppression of Time Dependency and its Extension to Multi-Dimensional Equalization," *Proc. of IEEE-ICC'86*, pp. 46.4.1-46.4.5.

[47] Sato, Y., "A Method for Self-Recovering Equalization for Multilevel Amplitude-Modulation Systems," *IEEE Trans. on Communications*, Vol. COM-23, pp. 679-682, June 1975.

[48] Shalvi, O. and E. Weinstein, "New Criteria for Blind Deconvolution of Nonminimum Phase Systems (Channels)," *IEEE Trans. Information Theory*, Vol. 36, pp. 312-321, 1990.

[49] Siu, S., G.J. Gibson, and C.F.N. Cowan, "Decision Feedback Equalization Using Neural Network Structures," *IEEE International Conference on Neural Networks*, London.

[50] Treichler, J.R. and B.G. Agee, "A New Approach to Multipath Correction of Constant Modulus Signals," *IEEE Trans. on Acoustics, Speech and Signal Processing*, Vol ASSP-31, No. 2, pp. 459-471, April 1983.

[51] Ungerbeck, G., "Theory on the Speed of Convergence in Adaptive Equalizers for Digital Communication," *IBM J. Res. Develop.*, pp. 546-555, November 1972.

[52] Van Trees, H.L., Detection, Estimation and Modulation Theory, Part I, J. Wiley and Sons, Inc., New York, 1968.

[53] Widrow, B. and S.D. Stearns, Adaptive Signal Processing, Prentice Hall, Englewood Cliffs, NJ, 1985.

[54] Wiggins, R.A., "Minimum Entropy Deconvolution," *Geoexploration*, Vol. 16, pp.21-35, 1978.

Table 4.1 Nonlinear Functions of Bussgang Iterative Techniques.

$\underline{u}(i) = [u_1,(i), \cdots, u_N(i)]^T$ equalizer taps

$\underline{y}(i) = [y(i), \cdots, y(i-N+1)]^T$ input to the equalizer block of data

At iteration$\{i\}$, $i = 1, 2, \cdots$

$\tilde{x}(i) = \underline{u}^H(i)\ \underline{y}(i)$

$e(i) = g^{(i)}[\tilde{x}(i)] - \tilde{x}(i)$

$\underline{u}(i+1) = \underline{u}(i) + \mu\ \underline{y}(i)\ e^*(i)$

Algorithm	Nonlinear function: $g[\tilde{x}(i)] =$
LMS training mode	$x(i)$ (linear)
Decision Directed Mode	$\hat{x}(i)$
Sato	γ csgn $[\tilde{x}(i)]$
Benveniste-Goursat	$\tilde{x}(i) + k_1(\hat{x}(i) - \tilde{x}(i)) + k_2\lvert\hat{x}(i) - \tilde{x}(i)\rvert \cdot (\gamma\ \text{csgn}[\tilde{x}(i)] - \tilde{x}(i))$
Godard $p, q = 2$	$\frac{\tilde{x}(i)}{\lvert\tilde{x}(i)\rvert} \cdot \{\lvert\tilde{x}(i)\rvert + R_p\lvert\tilde{x}(i)\rvert^{p-1} - \lvert\tilde{x}(i)\rvert^{2p-1}\}$
Stop-and-Go	$\tilde{x}(i) + \frac{1}{2}A(\hat{x}(i) - \tilde{x}(i)) + \frac{1}{2}B(\hat{x}(i) - \tilde{x}(i))^*$ (A,B) = (2,0), (1,1), (1,-1) or (0,0), depending on the signs of DD and Sato errors

Table 4.2 Comparison of Computational Complexity

	Godard	CRIMNO (memory size M)		Adaptive Weight CRIMNO (memory size M)
		Version I	Version II	Version I
Real Multiplication	4N+5	4N+8M+5	MN+8M+4N+5	4N+10M+5

Table 6.1 Complex MLP adaptation algorithm.

1). Assign small random complex numbers to all the connections and thresholds.

2). Forward propagate inputs through the network:

$$\tilde{a}_{i+1,j} = \sum_{l=1}^{n_i} a_{i,l} \cdot w_{i,l,j}^{*} + v_{i,j} = \tilde{a}_{i+1,j}^{I} + j \cdot \tilde{a}_{i+1,j}^{Q},$$

$$\tilde{a}_{i+1,j} = f(\tilde{a}_{i+1,j}^{I}) + j \cdot f(\tilde{a}_{i+1,j}^{Q}),$$

where $i = 1, \cdots M$ (M is the number of layers), $f(\cdot)$ is the sigmoid function, and get the output,

$$\hat{x} = C \cdot a_{M1}.$$

3). Present the training signal to find the output error,

$$\bar{e}_m = e_M^I \left[1 - (\hat{x}^I/C)^2\right]/C + j e_M^Q \left[1 - (\hat{x}^Q/C)^2\right]/C$$

where $e_M = x_{i-d} - \hat{x}$.

4). Find the backpropagation error,

$$\bar{e}_{ij} = \underline{e}_{ij}^I \cdot [1 - (a_{ij}^I)^2] + j \cdot \underline{e}_{ij}^Q \cdot [1 - (a_{ij}^Q)^2],$$

where

$$\underline{e}_{i,j} = \sum_{l=1}^{n_{i+1}} w_{i,j,l} \cdot \bar{e}_{i+1,l}.$$

5). Adjust connections and thresholds:

$$w_{i,j,k}(n+1) = w_{i,j,k}(n) + \eta \cdot \bar{e}_{i+1,j}^{*}(n) \cdot a_{ij}(n),$$
$$v_{ij}(n+1) = v_{ij}(n) + \beta \cdot \bar{e}_{ij}(n).$$

where "$*$" denotes conjugate operator. The momentum term can also be added.

6). Back to Step 2.

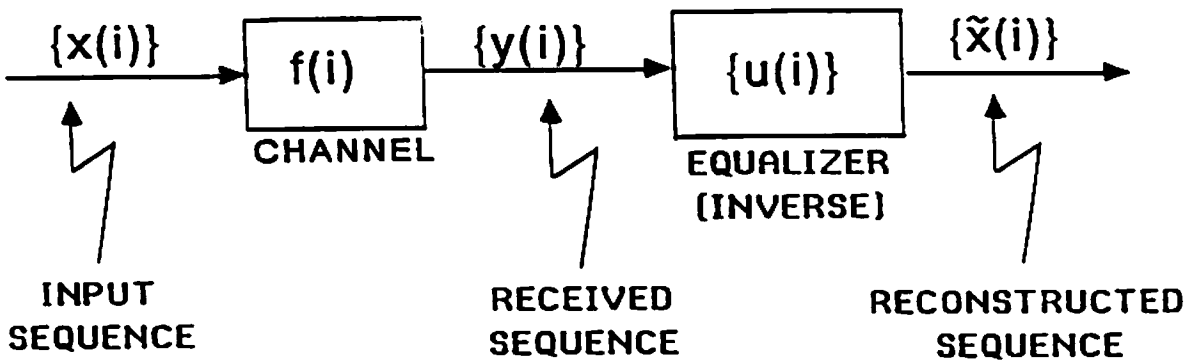


Figure 2.1 Block Diagram of Channel and Equalization Filter

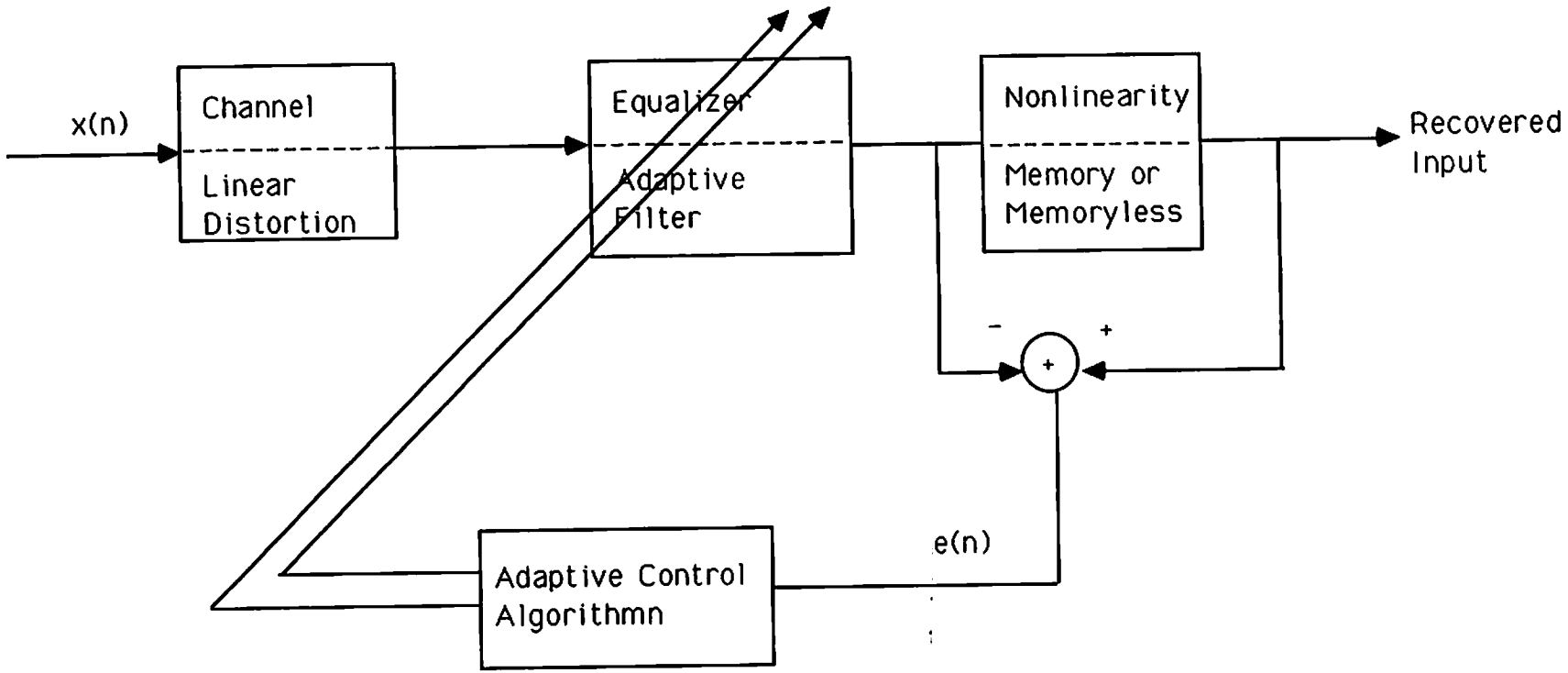


Figure 2. 2 (a) The Bussgang Algorithms: Nonlinearity is in the Output of Equalization Filter.

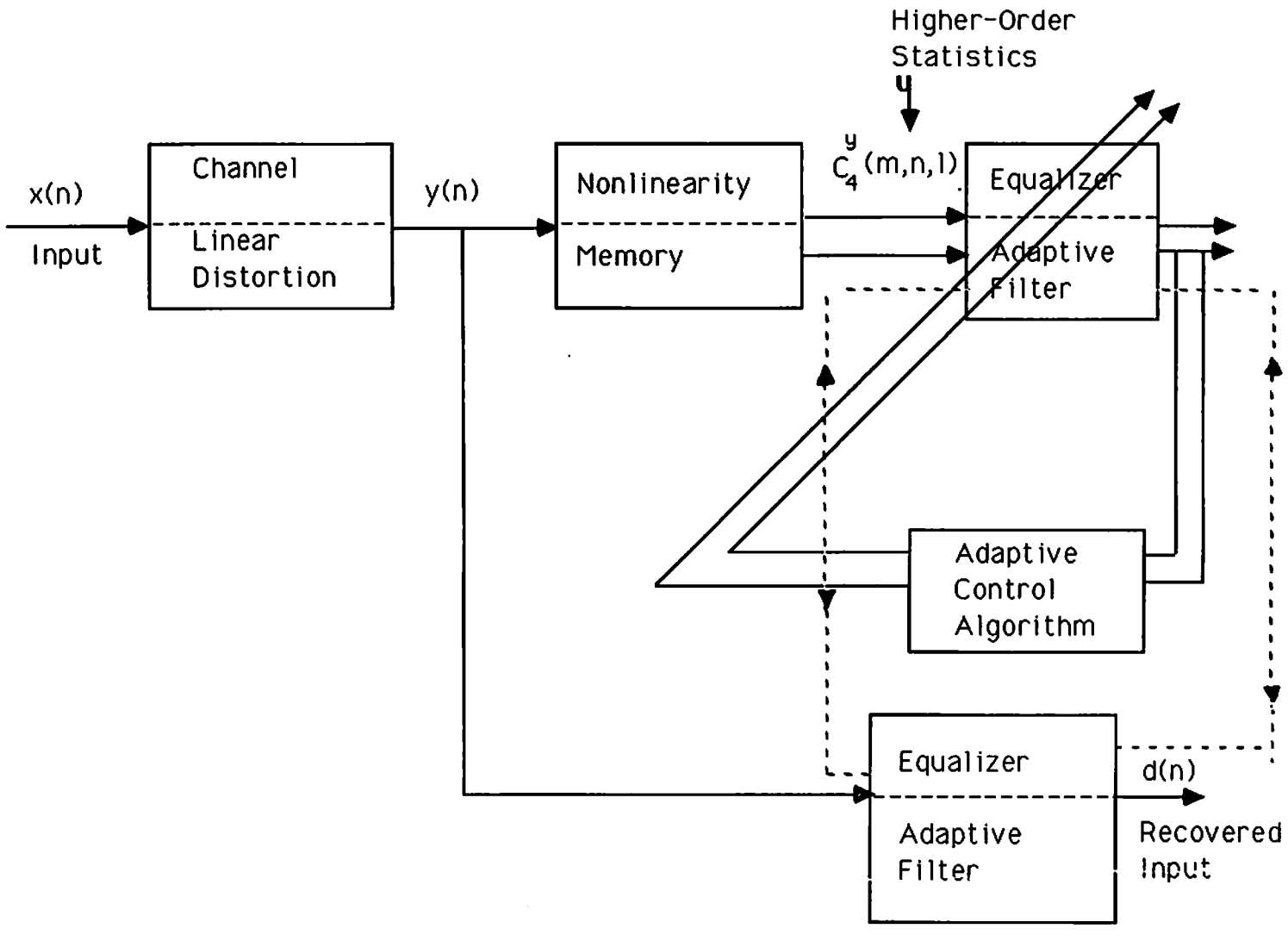


Figure 2. 2 (b) The Polyspectra Algorithms: Nonlinearity is in the Input of Equalization Filter.

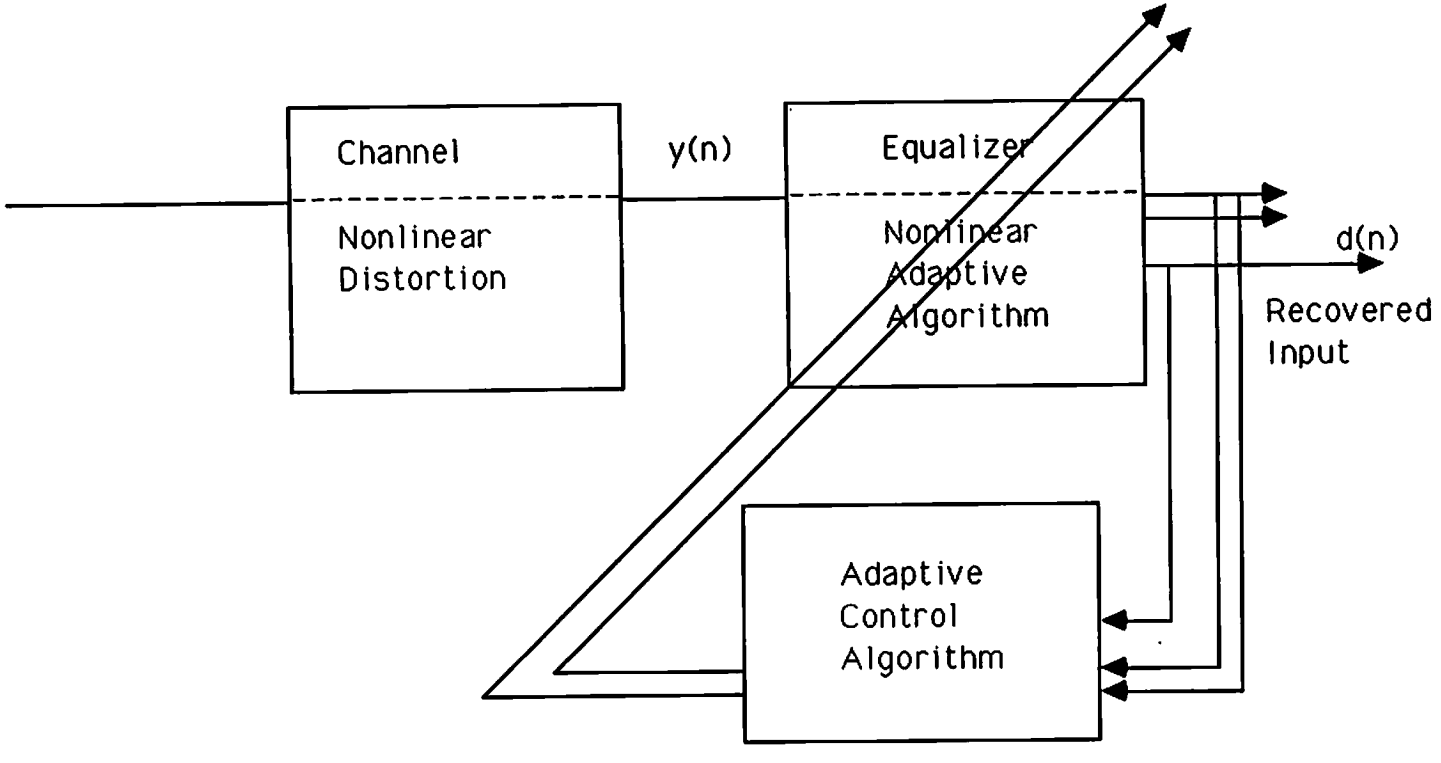


Figure 2. 2 (c) Blind Equalizers with Nonlinearity Inside the Equalization Filter.

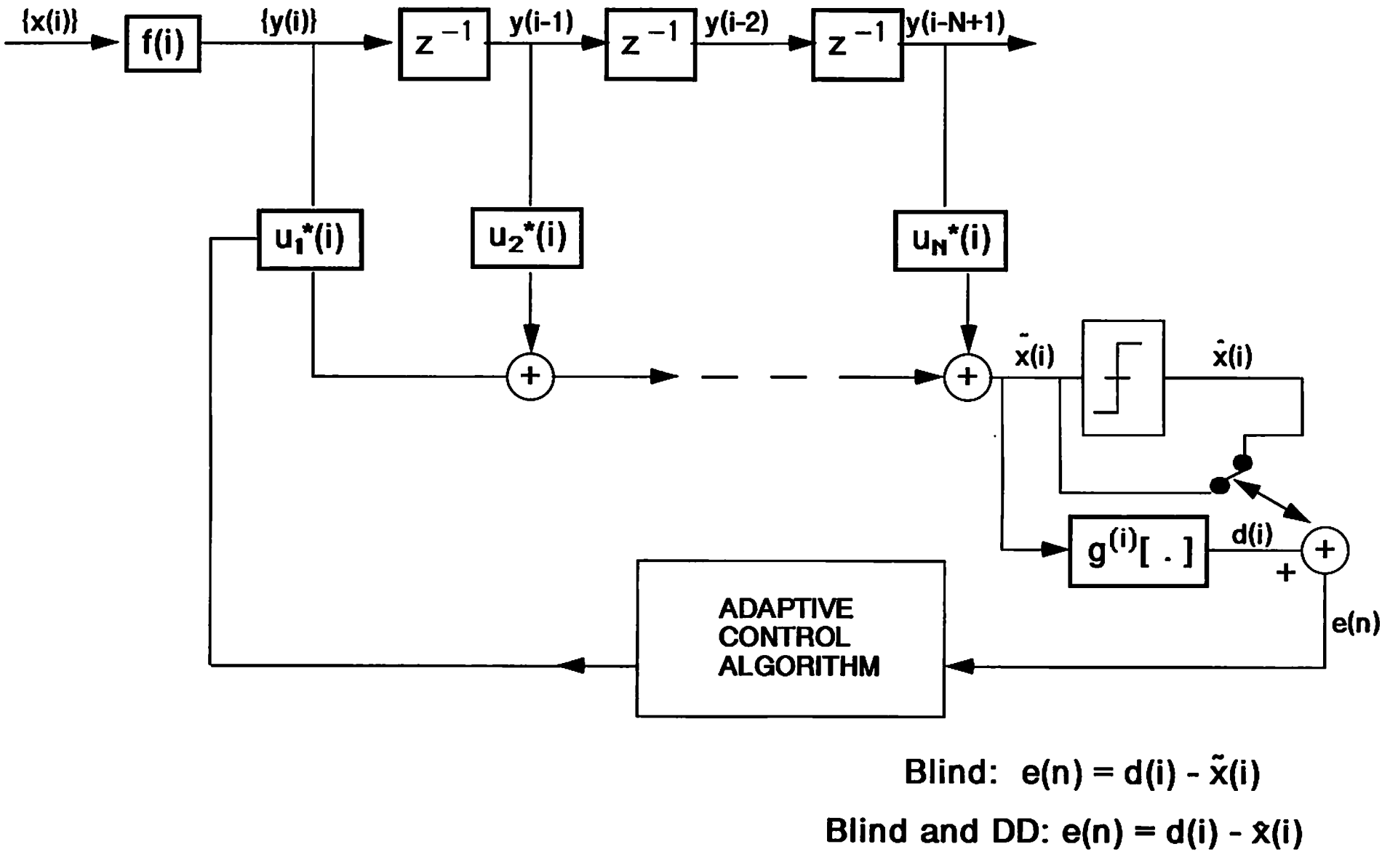


Figure 4.1 Linear Blind Equalization Filter with Nonlinearity in the Output.

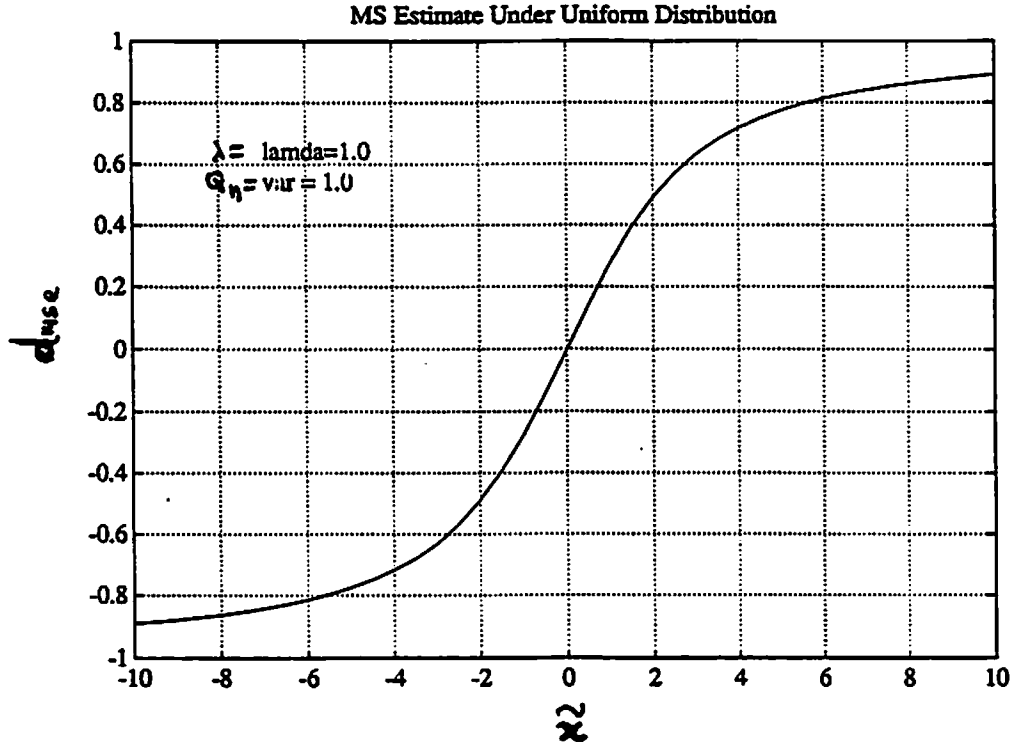


Figure 4.2 MS Estimate Under Uniform Distribution

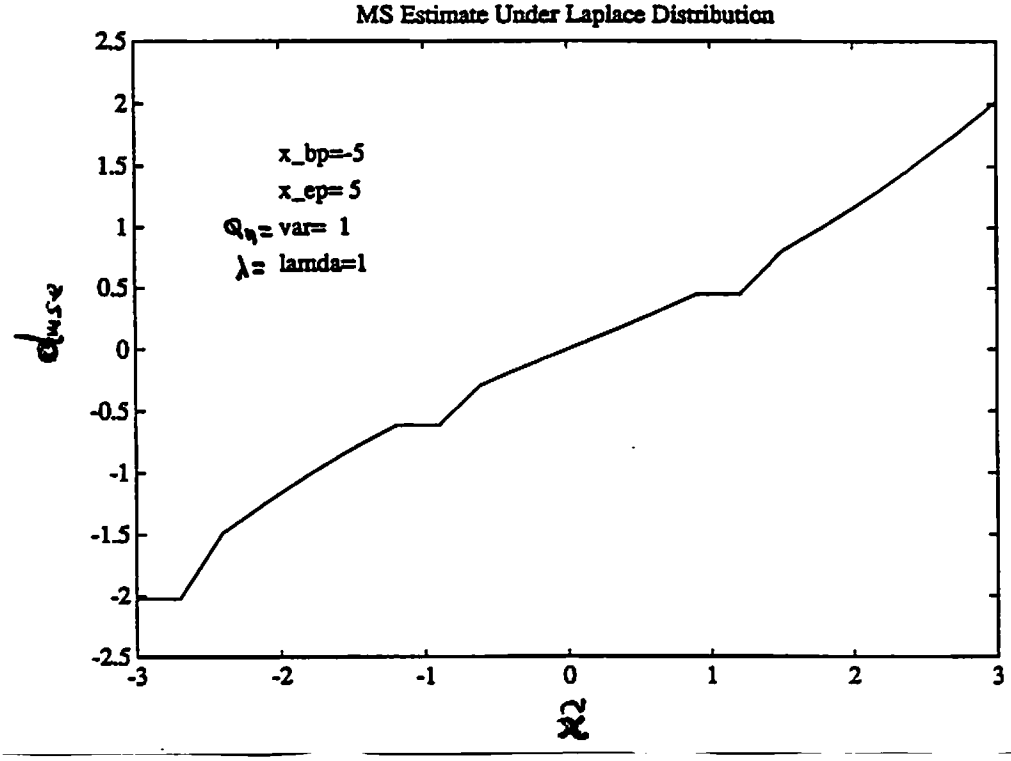
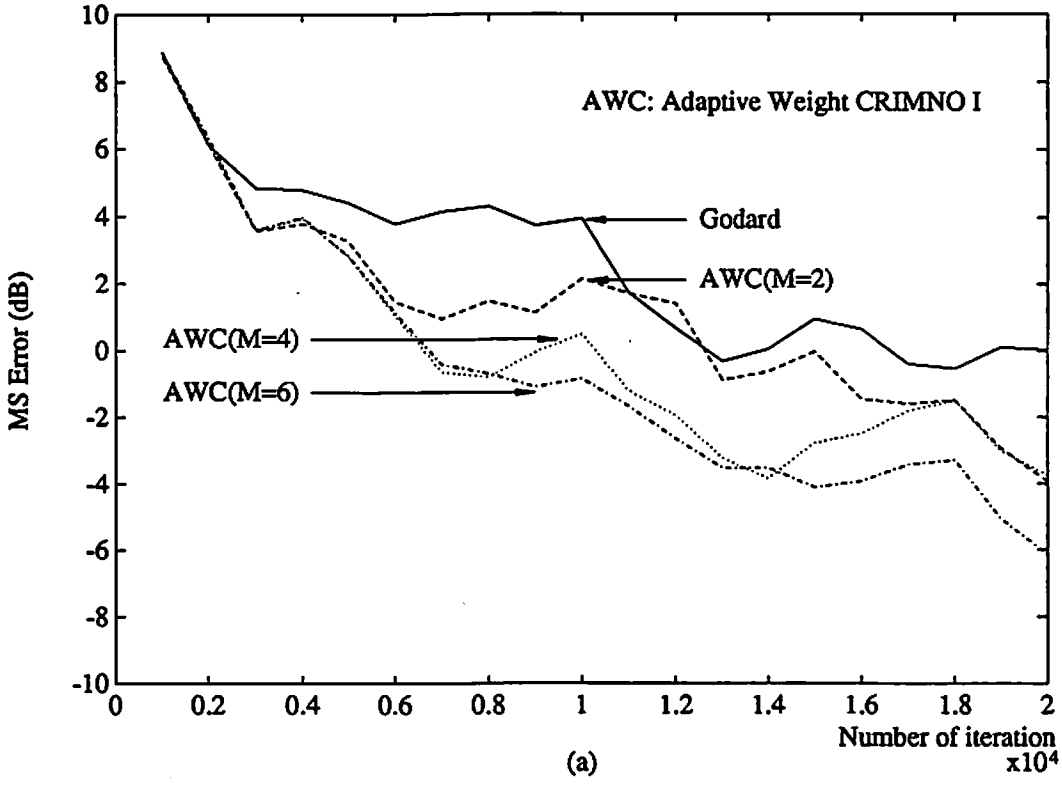
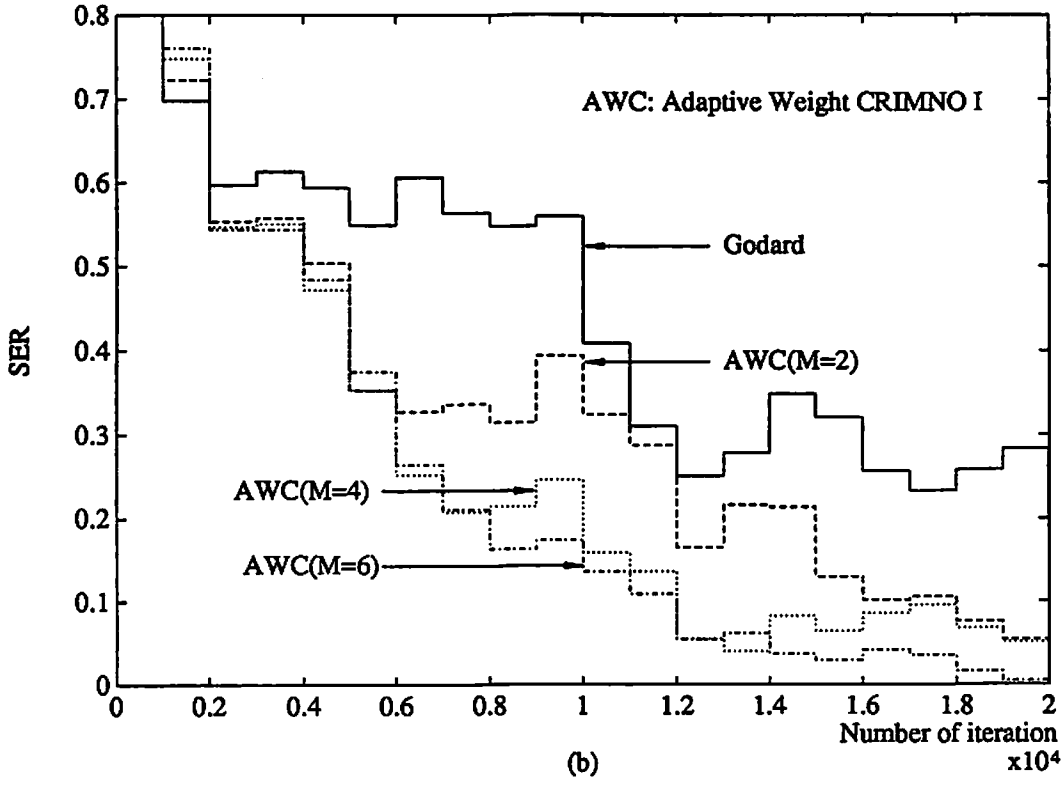


Figure 4.3 MS Estimate Under Laplace Distribution


AWC: Adaptive Weight CRIMNO I
Godard
AWC(M=2)
AWC(M=4)
AWC(M=6)
MS Error (dB)
Number of iteration
x10^4
(a)


AWC: Adaptive Weight CRIMNO I
Godard
AWC(M=2)
AWC(M=4)
AWC(M=6)
SER
Number of iteration
x10^4
(b)

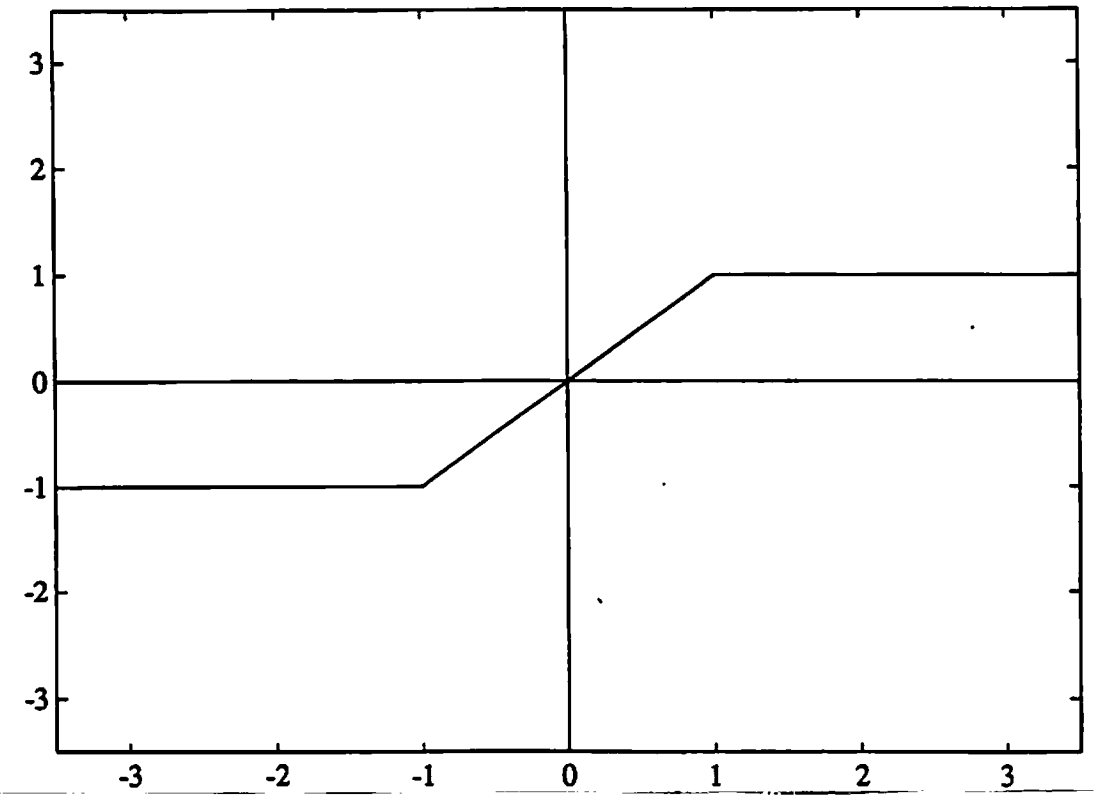


Figure 4.4 MAP Estimate Under Uniform Distribution

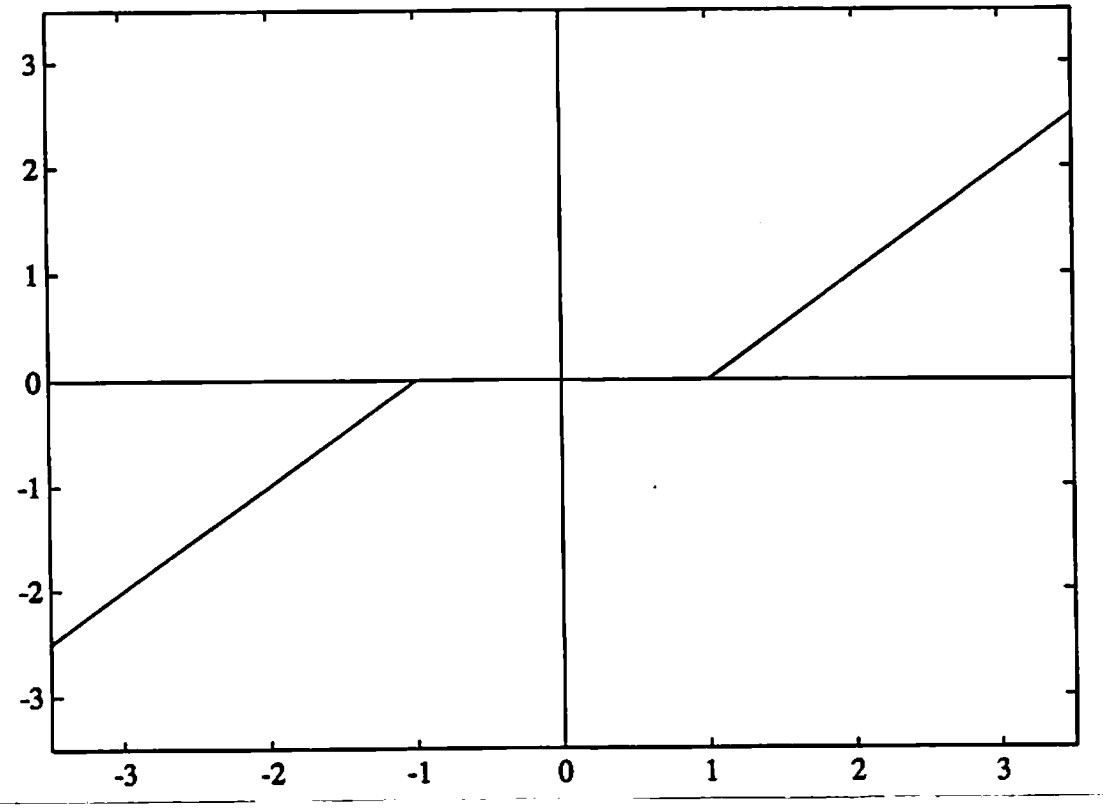


Figure 4.5 MAP Estimate Under Laplace Distribution

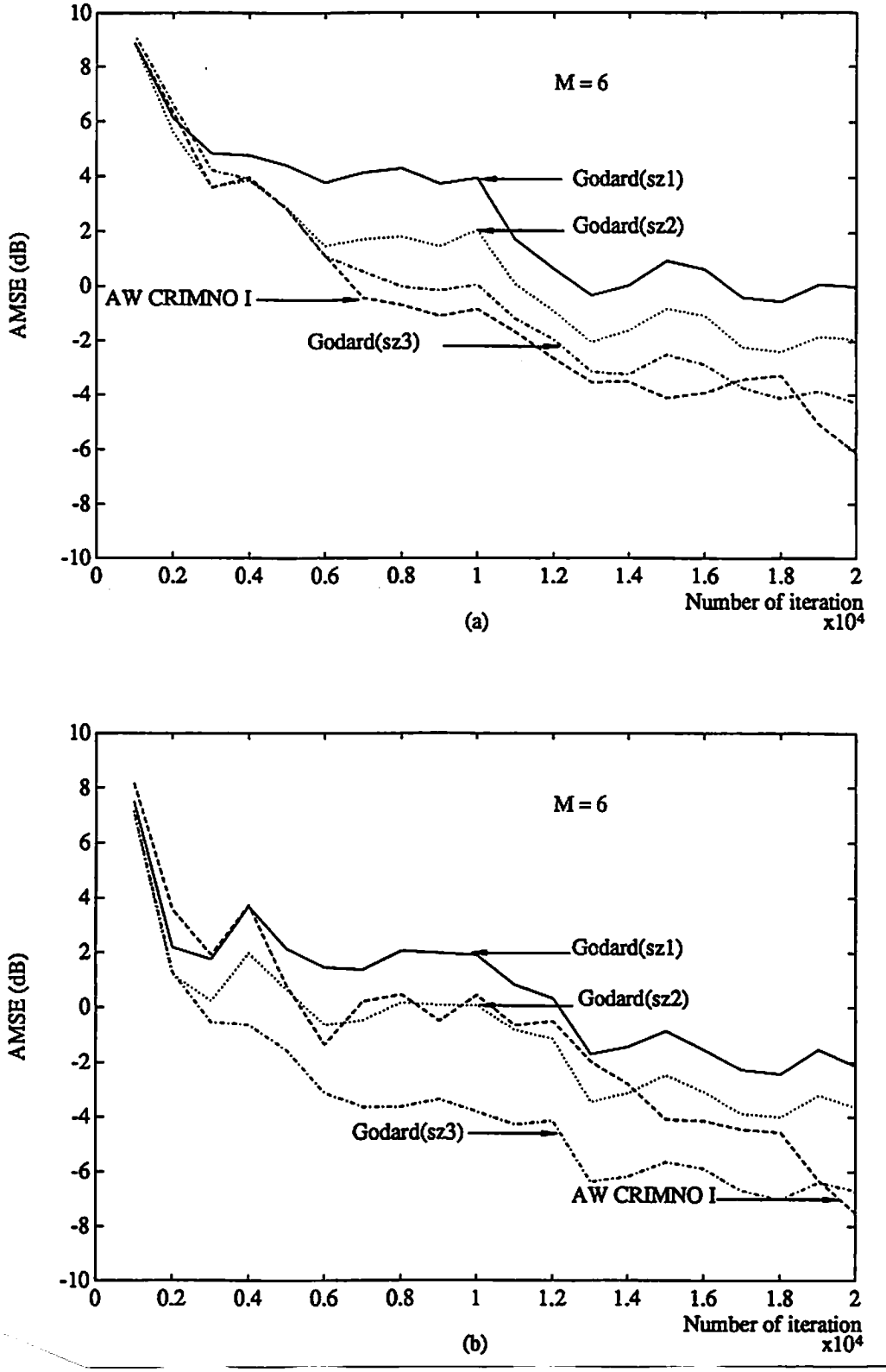


Figure 4.6 Comparison of the adaptive weight CRIMNO algorithm (szl) with Godard's algorithm of different step-size (sz3 is the optimum step-size): (a) the real channel; (b) the synthetic channel.

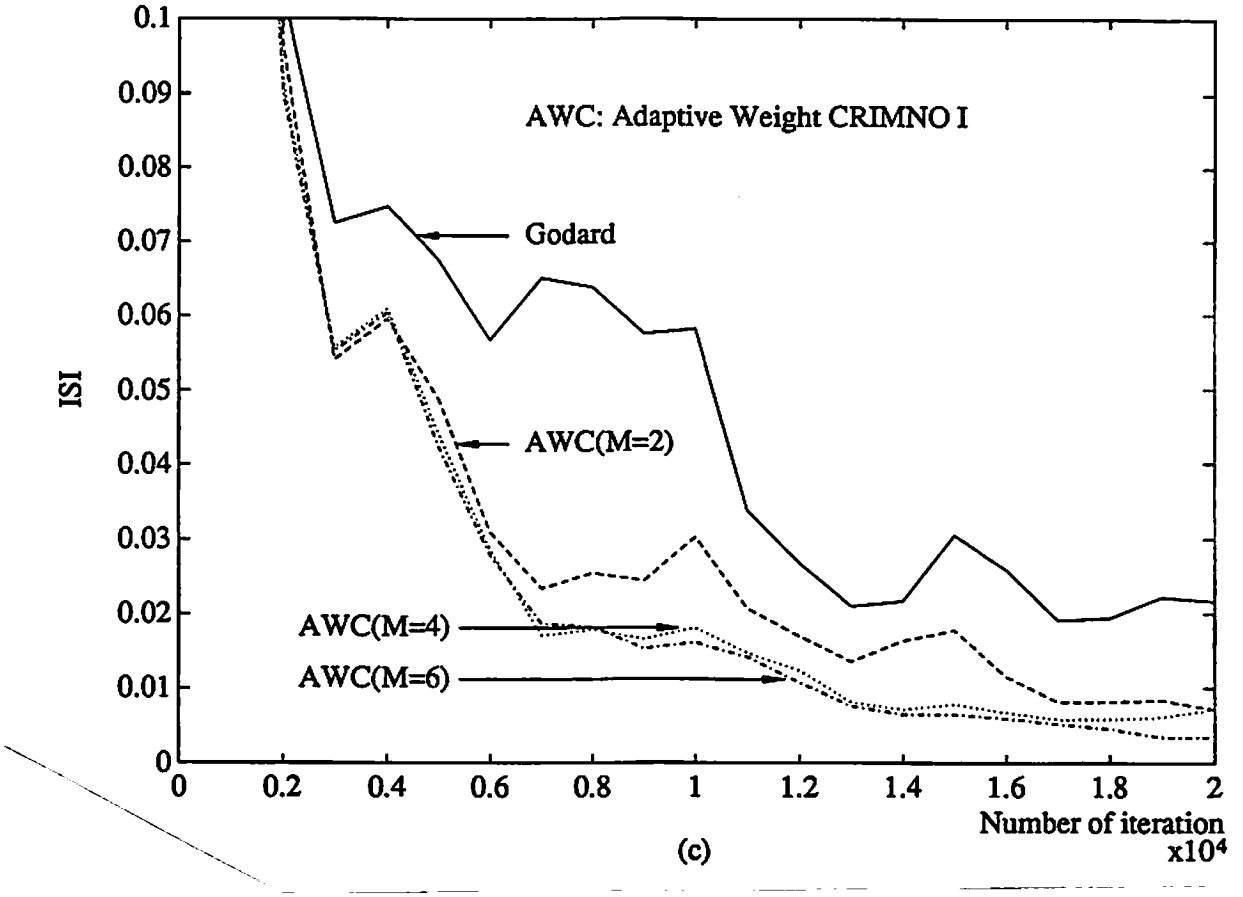


Figure 4.7 Effect of memory size M on the adaptive weight CRIMNO algorithm: (channel 2) (a) Mean square error; (b) Probability of error; (c) Intersymbol interference.

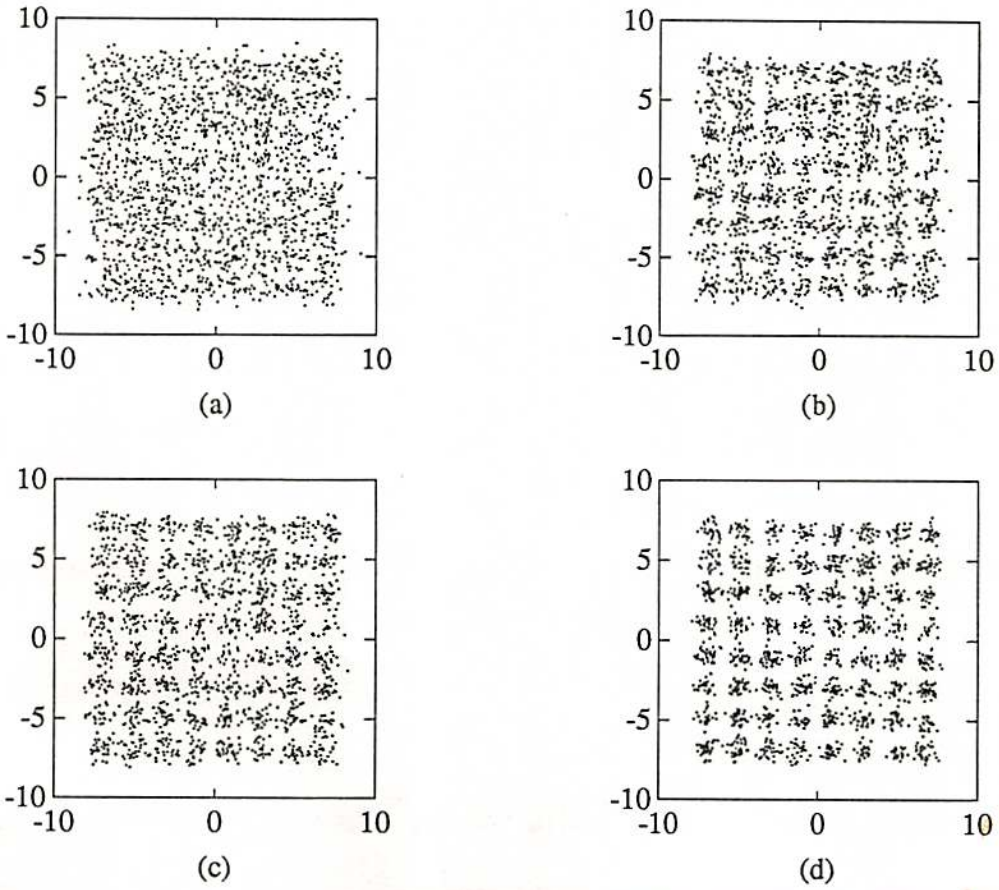


Figure 4.8 Eye pattern of adaptive weight CRIMNO algorithms with different memory size M at iteration 20000. (a) Godard; (b) Adaptive weight CRIMNO ($M=2$); (c) Adaptive weight CRIMNO ($M=4$); (d) Adaptive weight CRIMNO ($M=6$).

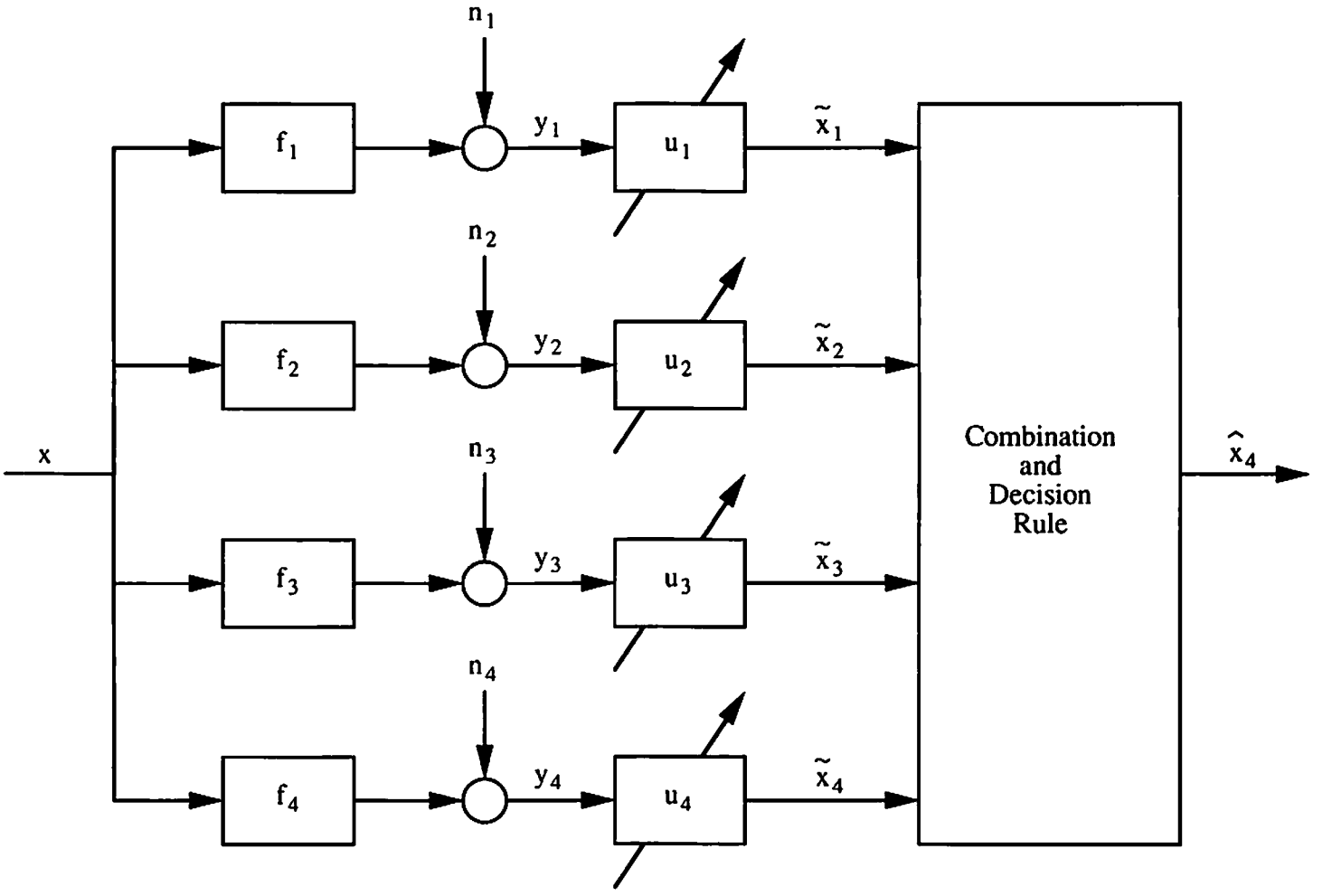


Figure 5.1 Diagram of four parallel equalized systems with additive noise.

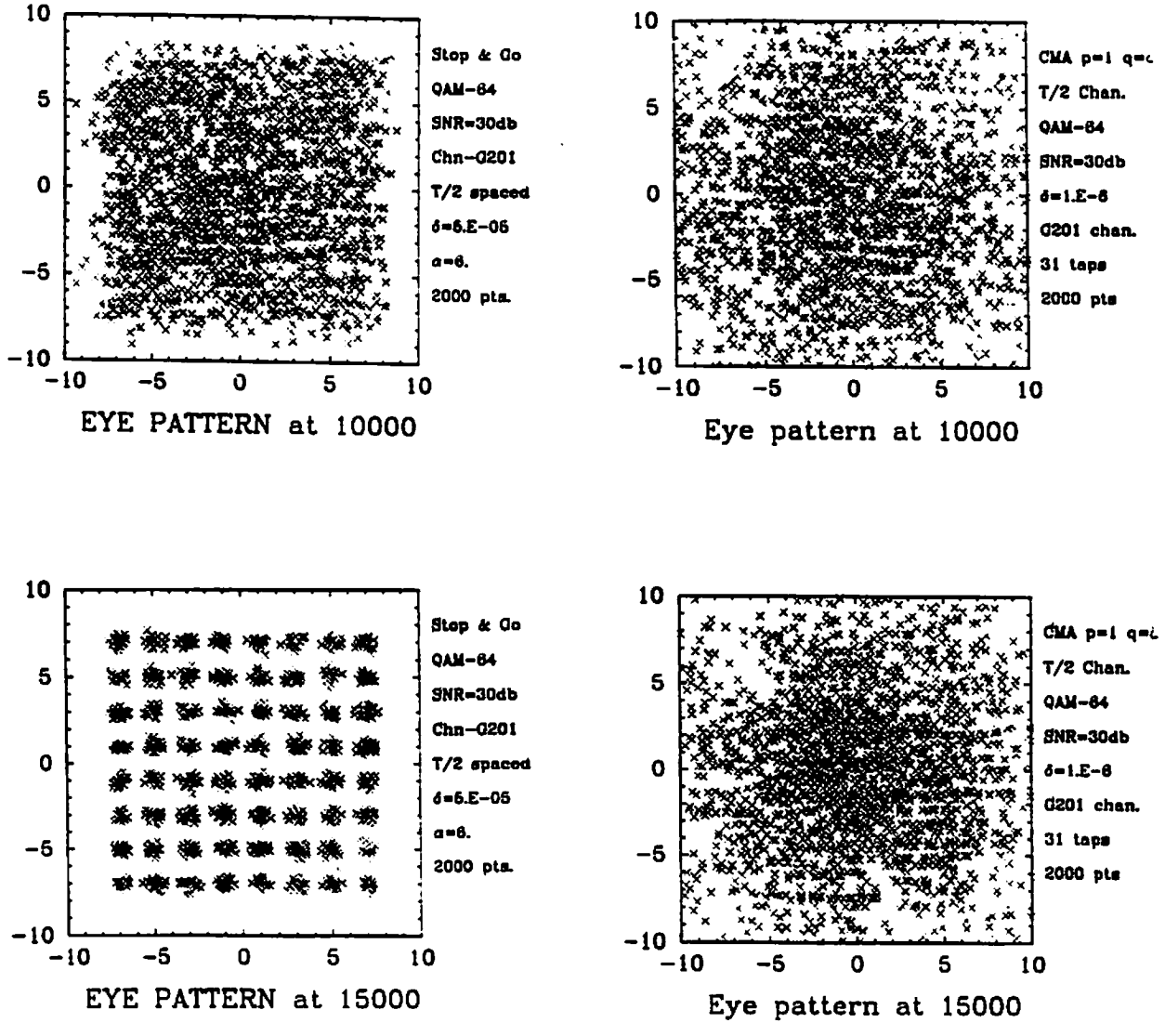


Figure 5.2 Channel G201 with QAM-64: Stop-and-Go and CMA (p=1, q=2) algorithms.

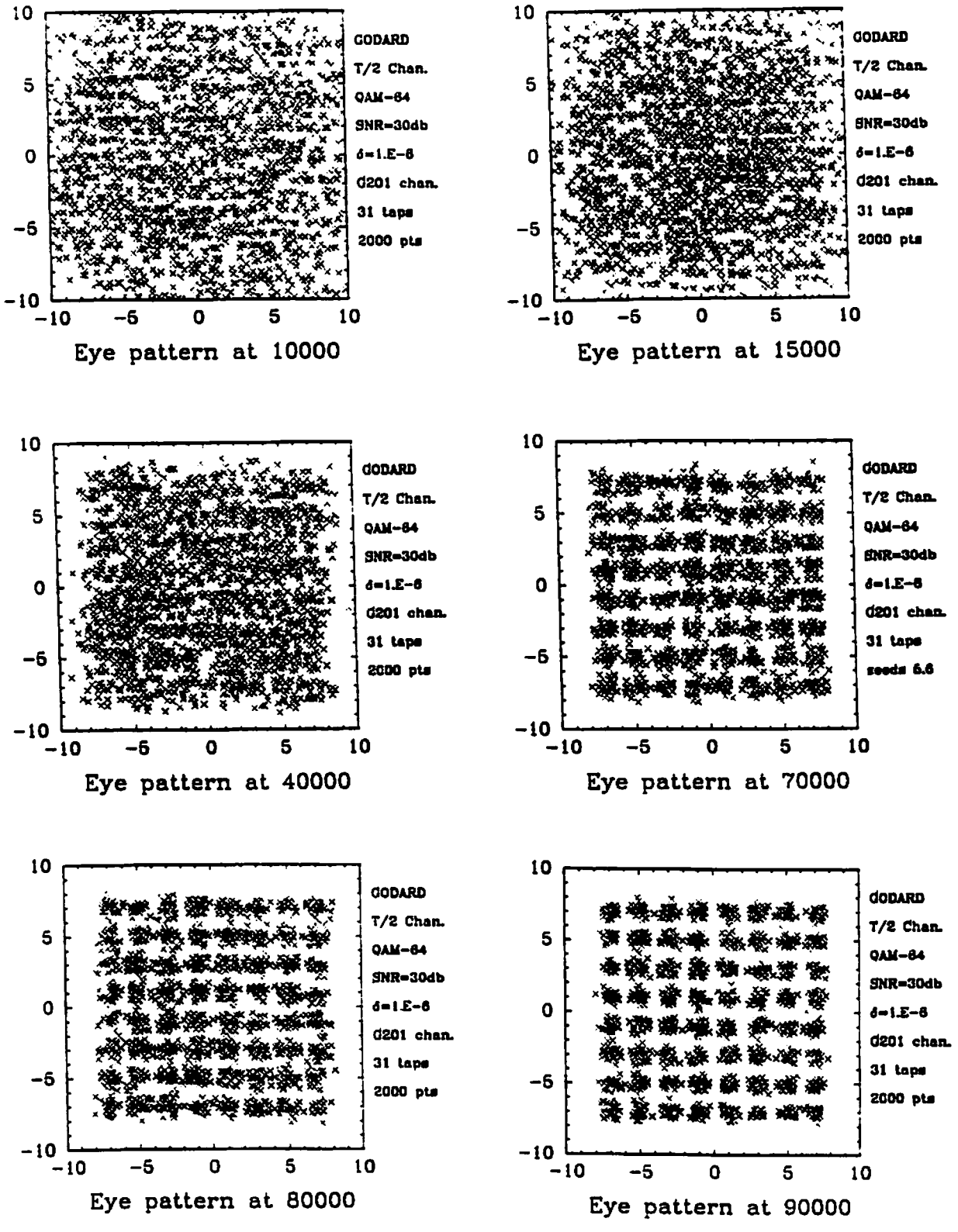


Figure 5.3 Channel G201 with QAM-64: Godard algorithms.

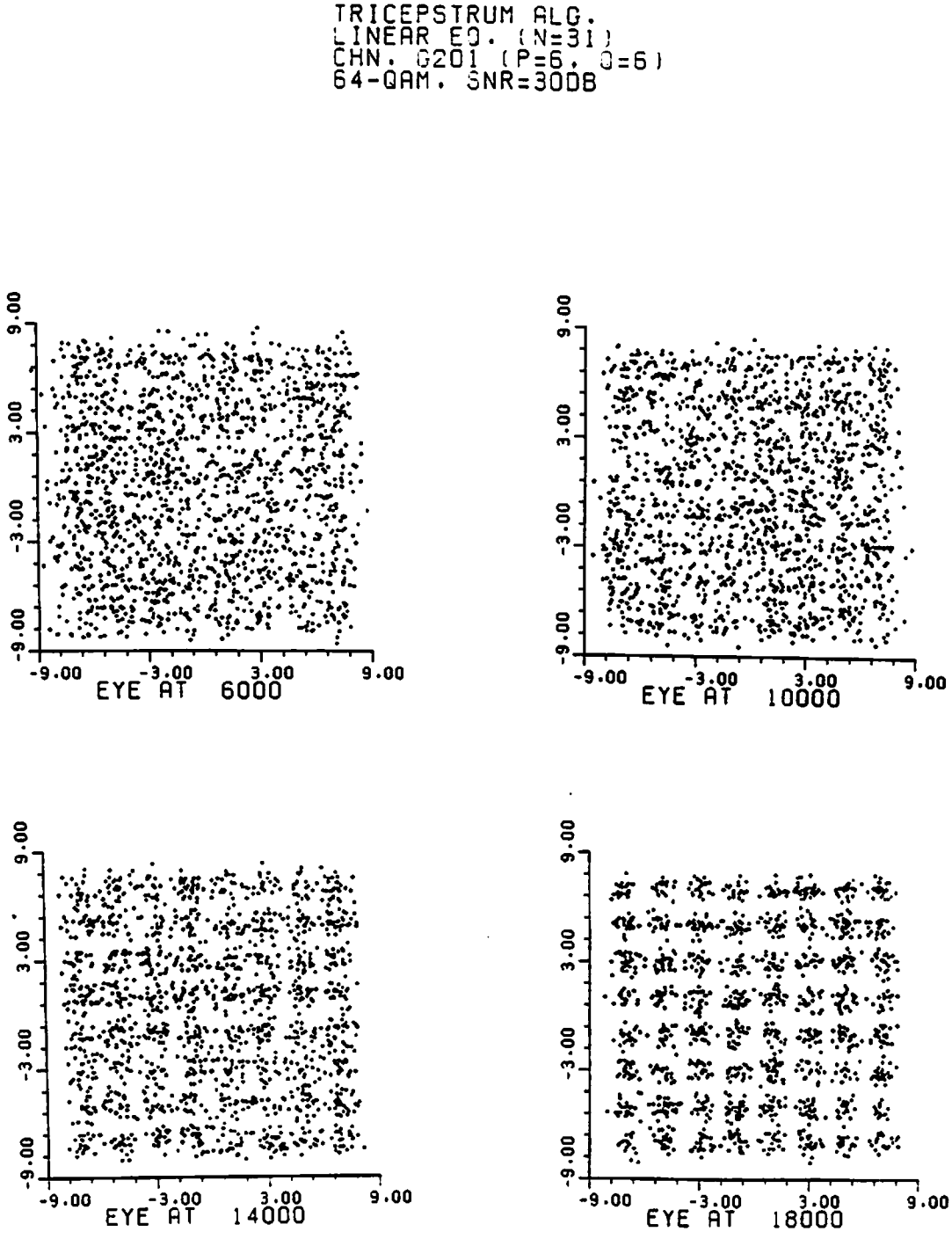


Figure 5.4 Channel G201 with QAM-64: TEA algorithms.

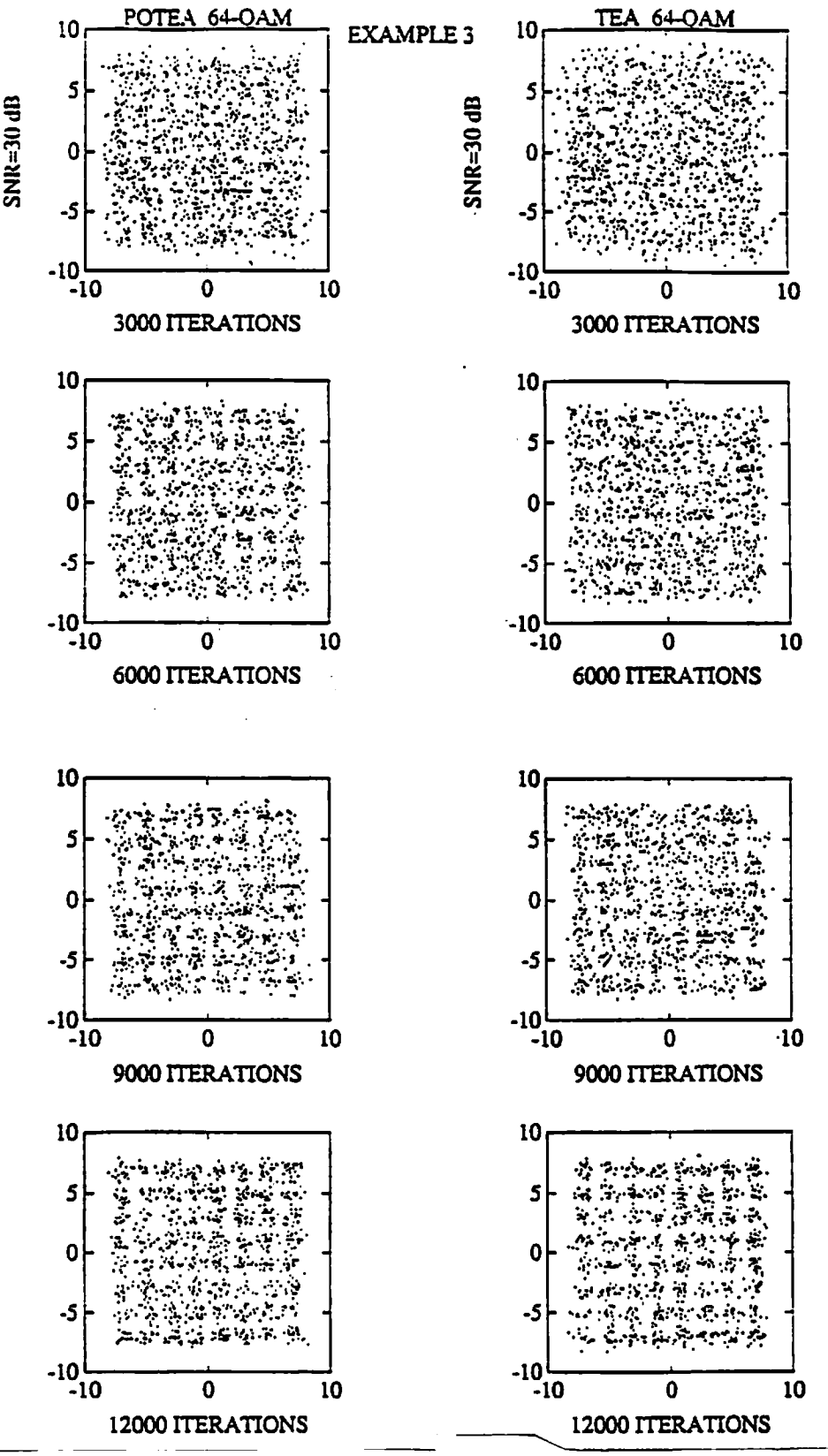


Figure 5.5 Eye-patterns of the Godard algorithm vs. those of the TEA algorithm.

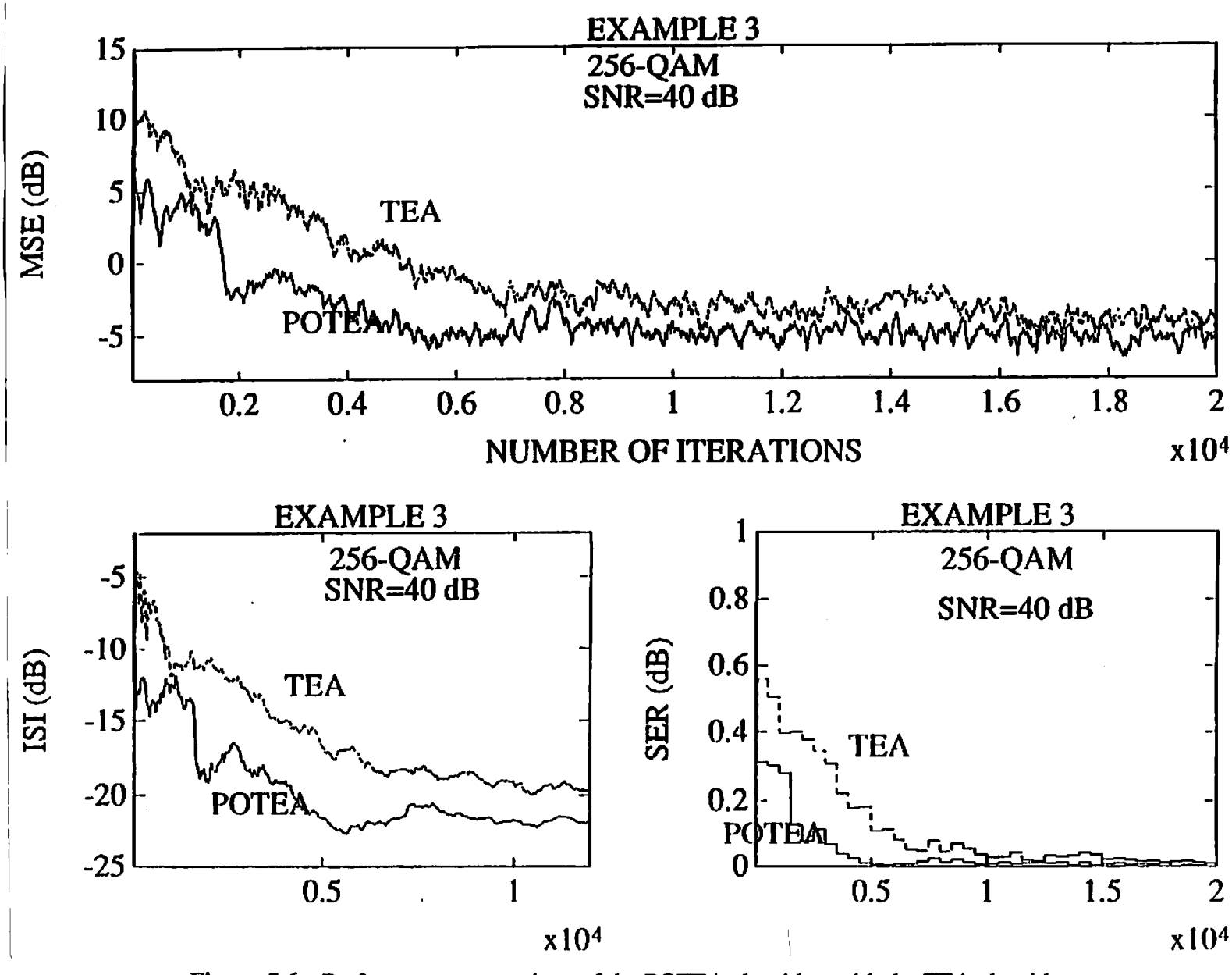


Figure 5.6 Performance comparison of the POTEA algorithm with the TEA algorithm.

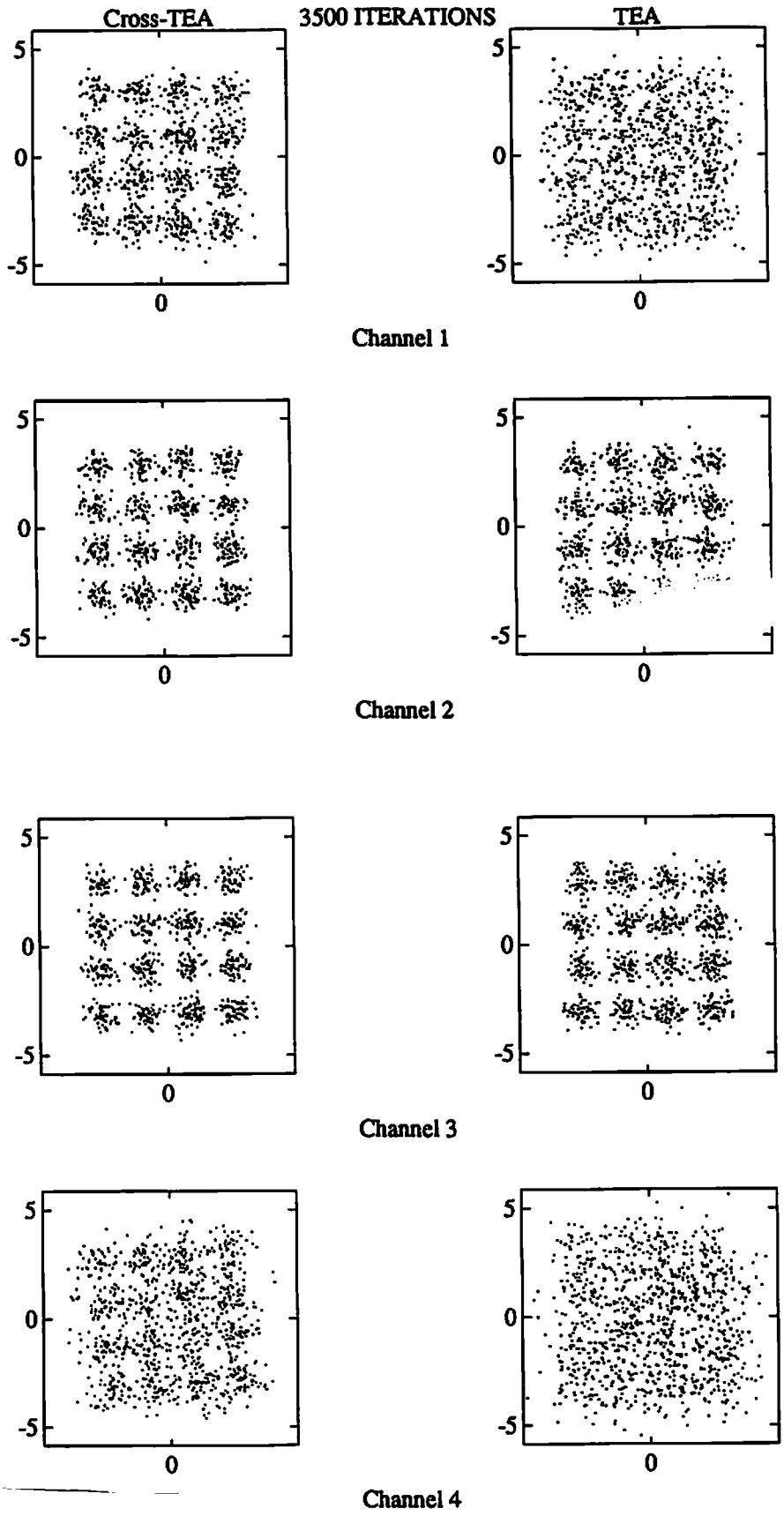


Figure 5.7 Eye Diagrams for CTEA and TEA at 3500 iterations..

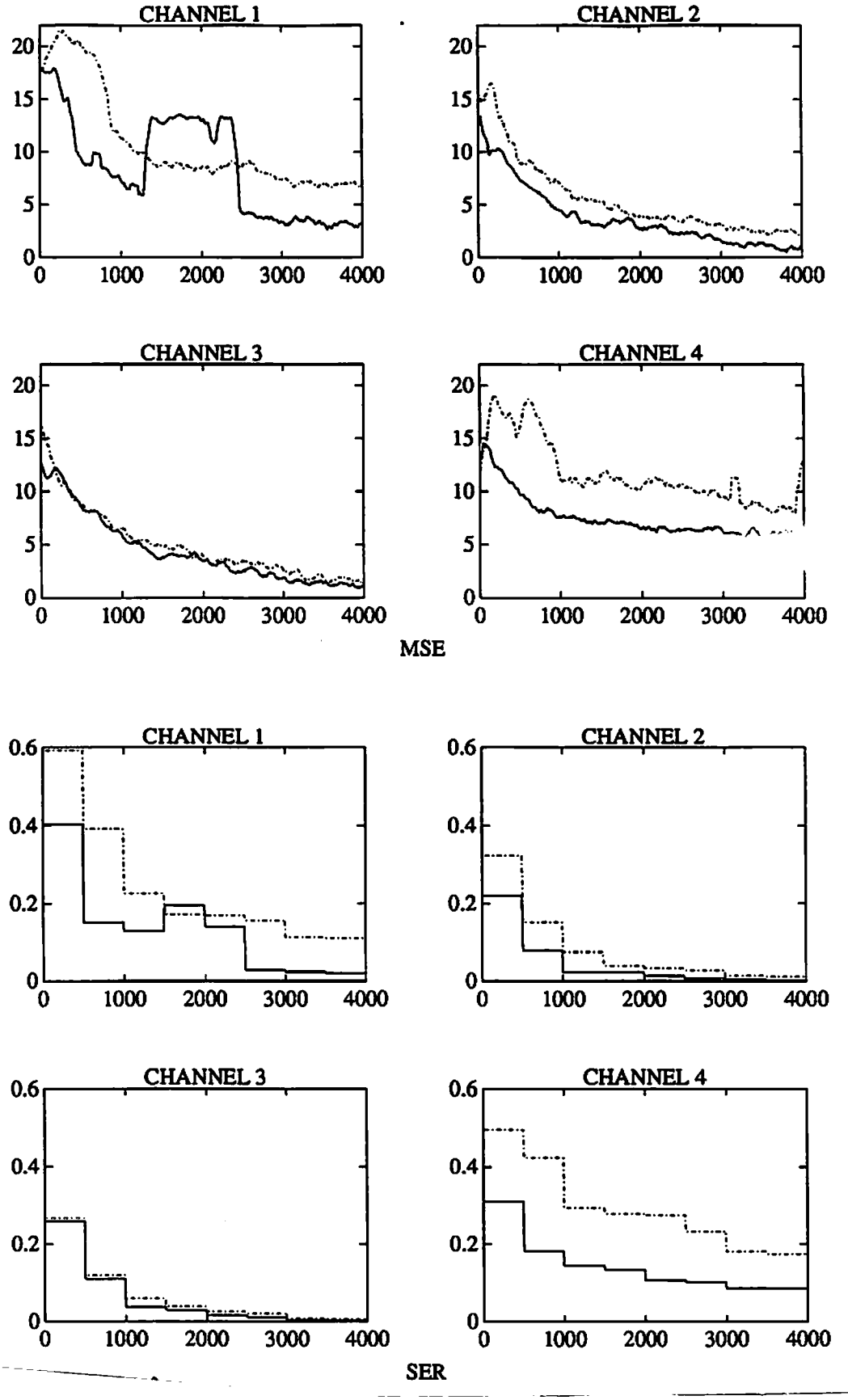


Figure 5.8 MSE and SER plots for all four channels, CTEA vs. TEA.

Table 4.1 Nonlinear Functions of Bussgang Iterative Techniques.

$\underline{u}(i) = [u_1,(i), \cdots, u_N(i)]^T$ equalizer taps

$\underline{y}(i) = [y(i), \cdots, y(i-N+1)]^T$ input to the equalizer block of data

At iteration$\{i\}$, $i = 1, 2, \cdots$

$\tilde{x}(i) = \underline{u}^H(i)\ \underline{y}(i)$

$e(i) = g^{(i)}[\tilde{x}(i)] - \tilde{x}(i)$

$\underline{u}(i+1) = \underline{u}(i) + \mu\ \underline{y}(i)\ e^*(i)$

Algorithm	Nonlinear function: $g[\tilde{x}(i)] =$
LMS training mode	$x(i)$ (linear)
Decision Directed Mode	$\hat{x}(i)$
Sato	γ csgn $[\tilde{x}(i)]$
Benveniste-Goursat	$\tilde{x}(i) + k_1\,(\hat{x}(i) - \tilde{x}(i)) + k_2\lvert\hat{x}(i) - \tilde{x}(i)\rvert \cdot (\gamma\ \text{csgn}[\tilde{x}(i)] - \tilde{x}(i))$
Godard $p, q = 2$	$\frac{\tilde{x}(i)}{\lvert\tilde{x}(i)\rvert} \cdot \{\lvert\tilde{x}(i)\rvert + R_p\lvert\tilde{x}(i)\rvert^{p-1} - \lvert\tilde{x}(i)\rvert^{2p-1}\}$
Stop-and-Go	$\tilde{x}(i) + \frac{1}{2}A\,(\hat{x}(i) - \tilde{x}(i)) + \frac{1}{2}B\,(\hat{x}(i) - \tilde{x}(i))^*$ (A,B) = (2,0), (1,1), (1,-1) or (0,0), depending on the signs of DD and Sato errors

Table 4.2 Comparison of Computational Complexity

	Godard	CRIMNO (memory size M)		Adaptive Weight CRIMNO (memory size M)
		Version I	Version II	Version I
Real Multiplication	4N+5	4N+8M+5	MN+8M+4N+5	4N+10M+5

Table 6.1 Complex MLP adaptation algorithm.

1). Assign small random complex numbers to all the connection weights and thresholds.

2). Forward propagate inputs through the network,

$$\tilde{a}_{i+1,j} = \sum_{l=1}^{n_i} a_{i,l} \cdot w_{i,l,j}^{*} + v_{i,j} = \tilde{a}_{i+1,j}^{I}$$

$$\tilde{a}_{i+1,j} = f(\tilde{a}_{i+1,j}^{I}) + j$$

where $i = 1, \cdots M$ (M is the number of layers) for each sigmoid function, and get the output,

$$\hat{x} = C \cdot a_{M1}.$$

3). Present the training signal to find the output error,

$$\tilde{e}_m = e_M^I \left[1 - (\hat{x}^I/C)^2\right]/C + j e_M^Q \left[1 - (\hat{x}^Q/C\right.$$

where $e_M = x_{i-d} - \hat{x}$.

4). Find the backpropagation error,

$$\tilde{e}_{ij} = \underline{e}_{ij}^I \cdot [1 - (a_{ij}^I)^2] + j \cdot \underline{e}_{ij}^Q \cdot [1 - (a_{ij}^Q)^2],$$

where

$$\underline{e}_{i,j} = \sum_{l=1}^{n_{i+1}} w_{i,j,l} \cdot \tilde{e}_{i+1,l}.$$

5). Adjust connections and thresholds:

$$w_{i,j,k}(n+1) = w_{i,j,k}(n) + \eta \cdot \tilde{e}_{i+1,j}^{*}(n) \cdot a_{ij}(n),$$
$$v_{ij}(n+1) = v_{ij}(n) + \beta \cdot \tilde{e}_{ij}(n).$$

where "$*$" denotes conjugate operator. The momentum term can also be added.

6). Back to Step 2.

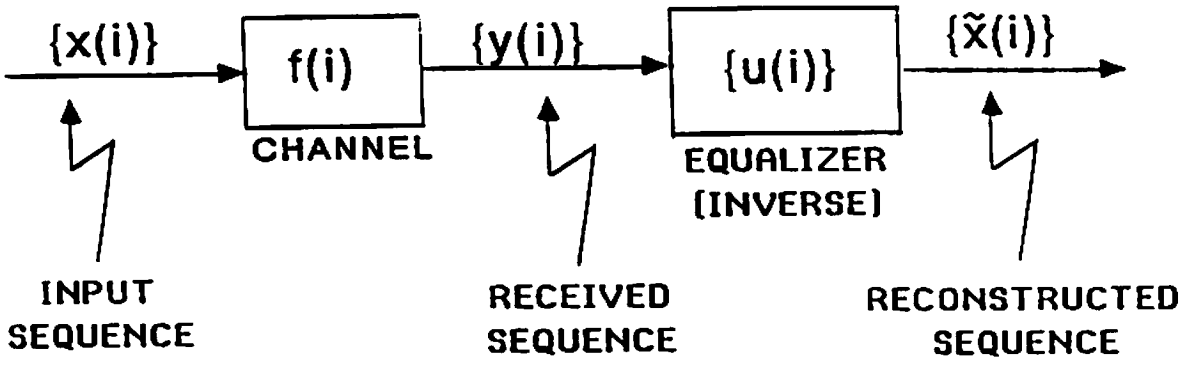


Figure 2.1 Block Diagram of Channel and Equalization Filter

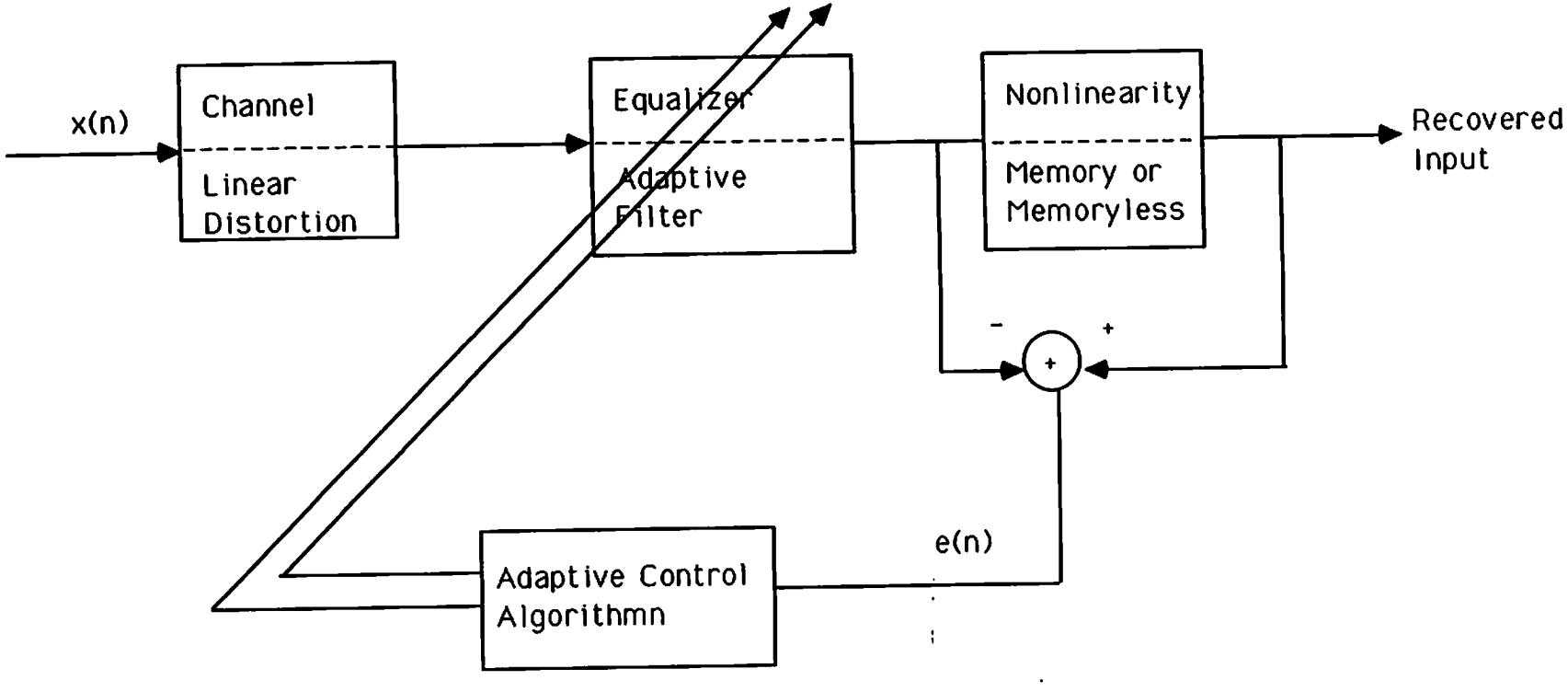


Figure 2. 2 (a) The Bussgang Algorithms: Nonlinearity is in the Output of Equalization Filter.

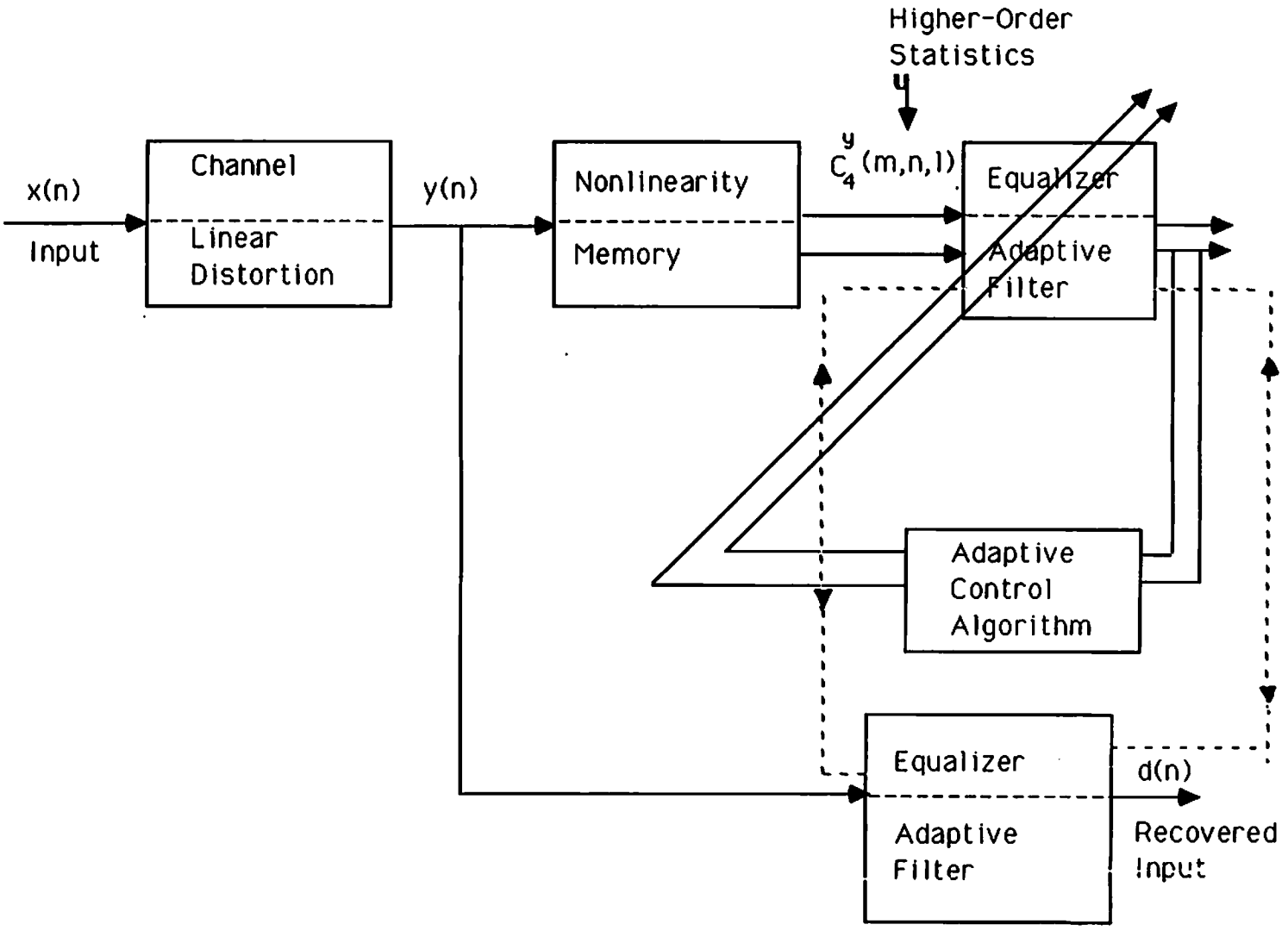


Figure 2. 2 (b) The Polyspectra Algorithms: Nonlinearity is in the Input of Equalization Filter.

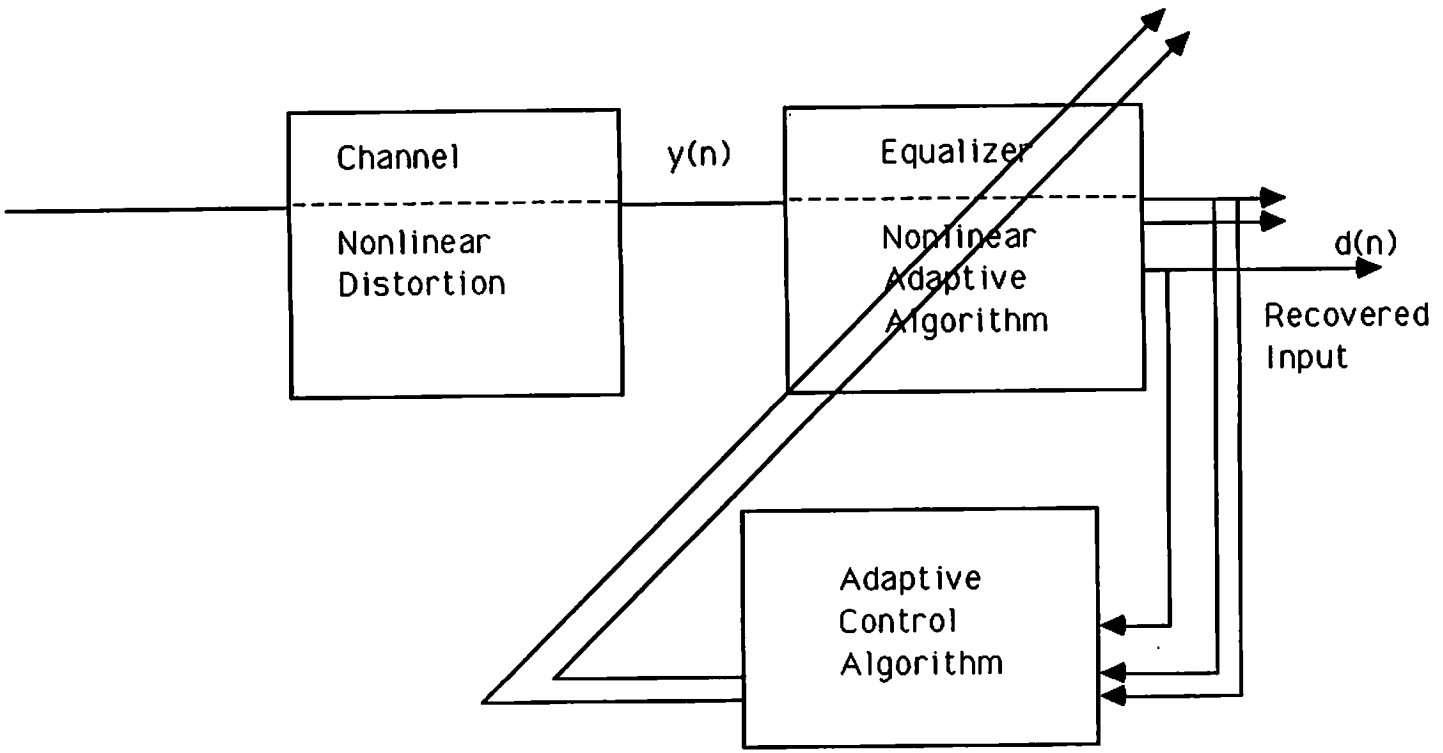


Figure 2. 2 (c) Blind Equalizers with Nonlinearity Inside the Equalization Filter.

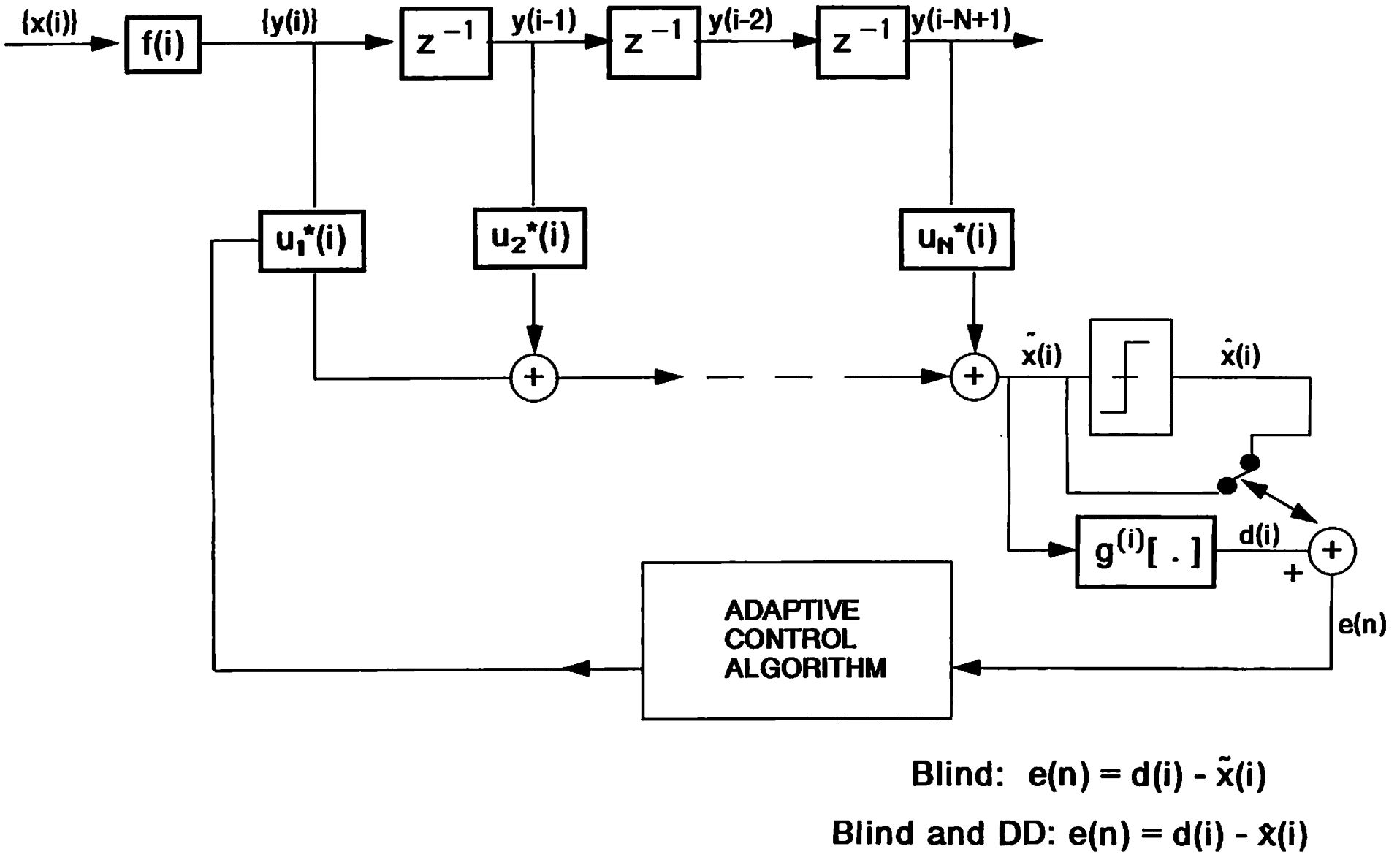


Figure 4.1 Linear Blind Equalization Filter with Nonlinearity in the Output.

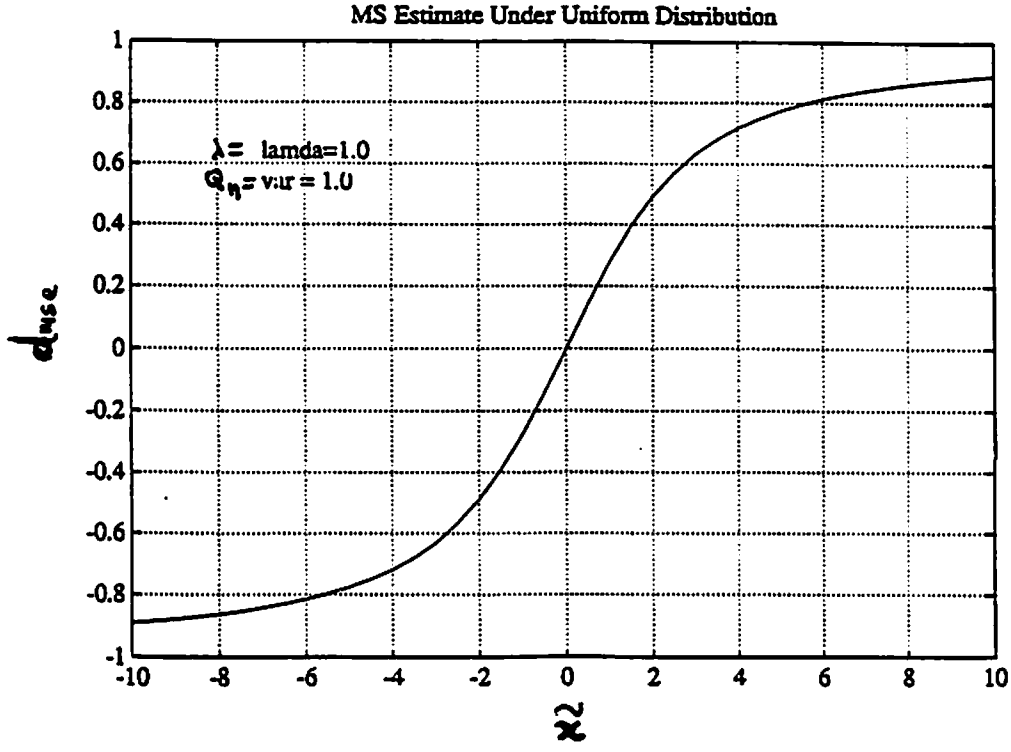


Figure 4.2 MS Estimate Under Uniform Distribution

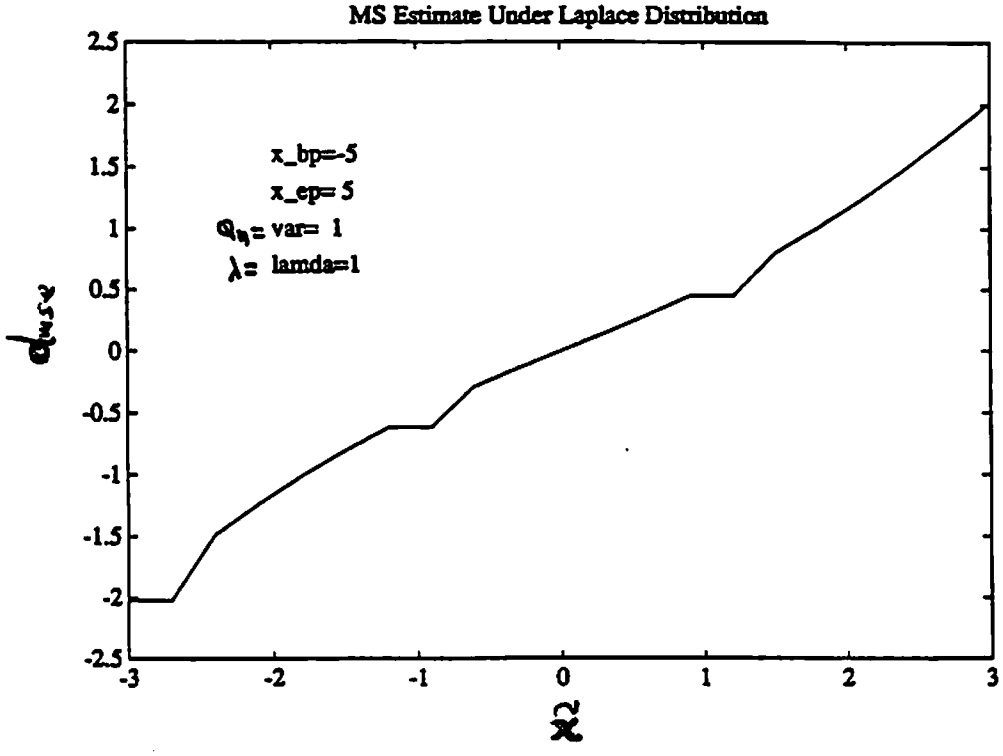


Figure 4.3 MS Estimate Under Laplace Distribution

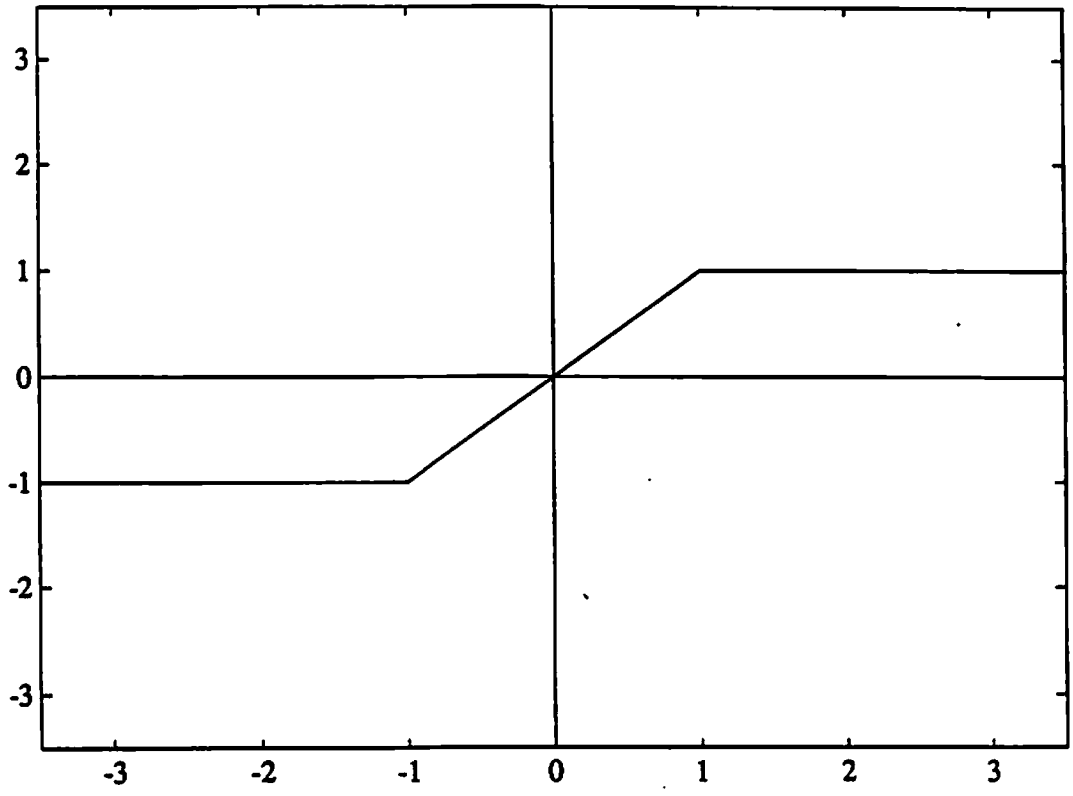


Figure 4.4 MAP Estimate Under Uniform Distribution

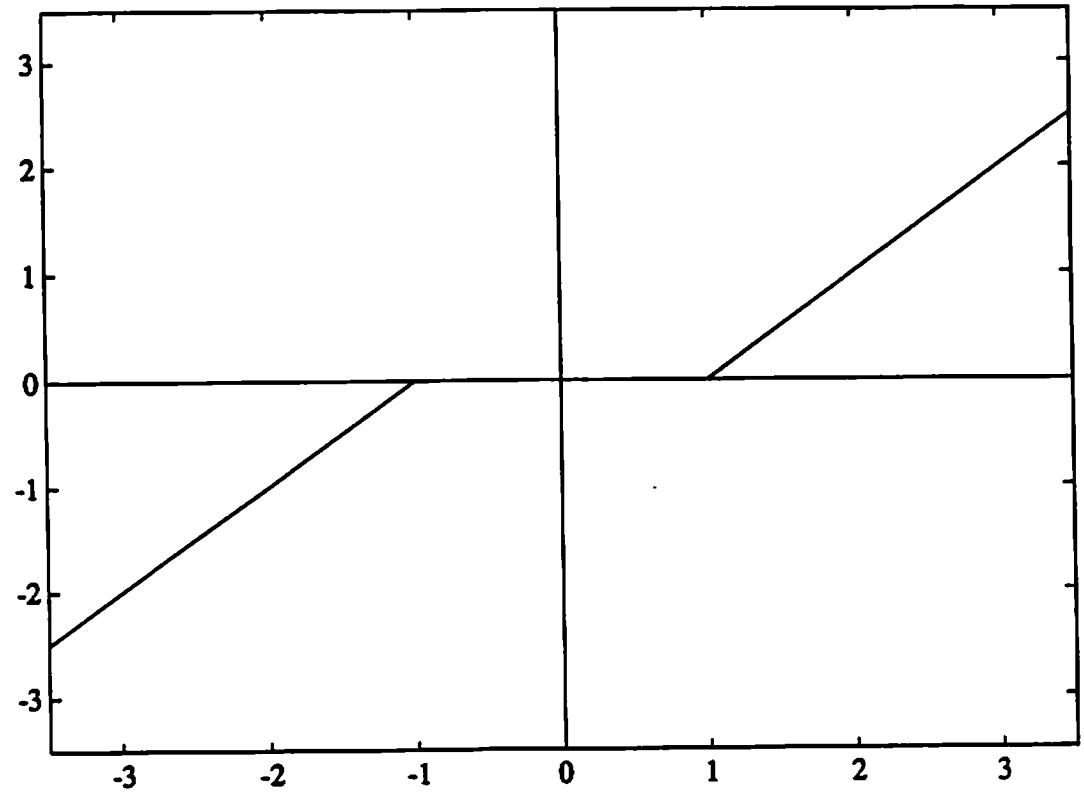


Figure 4.5 MAP Estimate Under Laplace Distribution

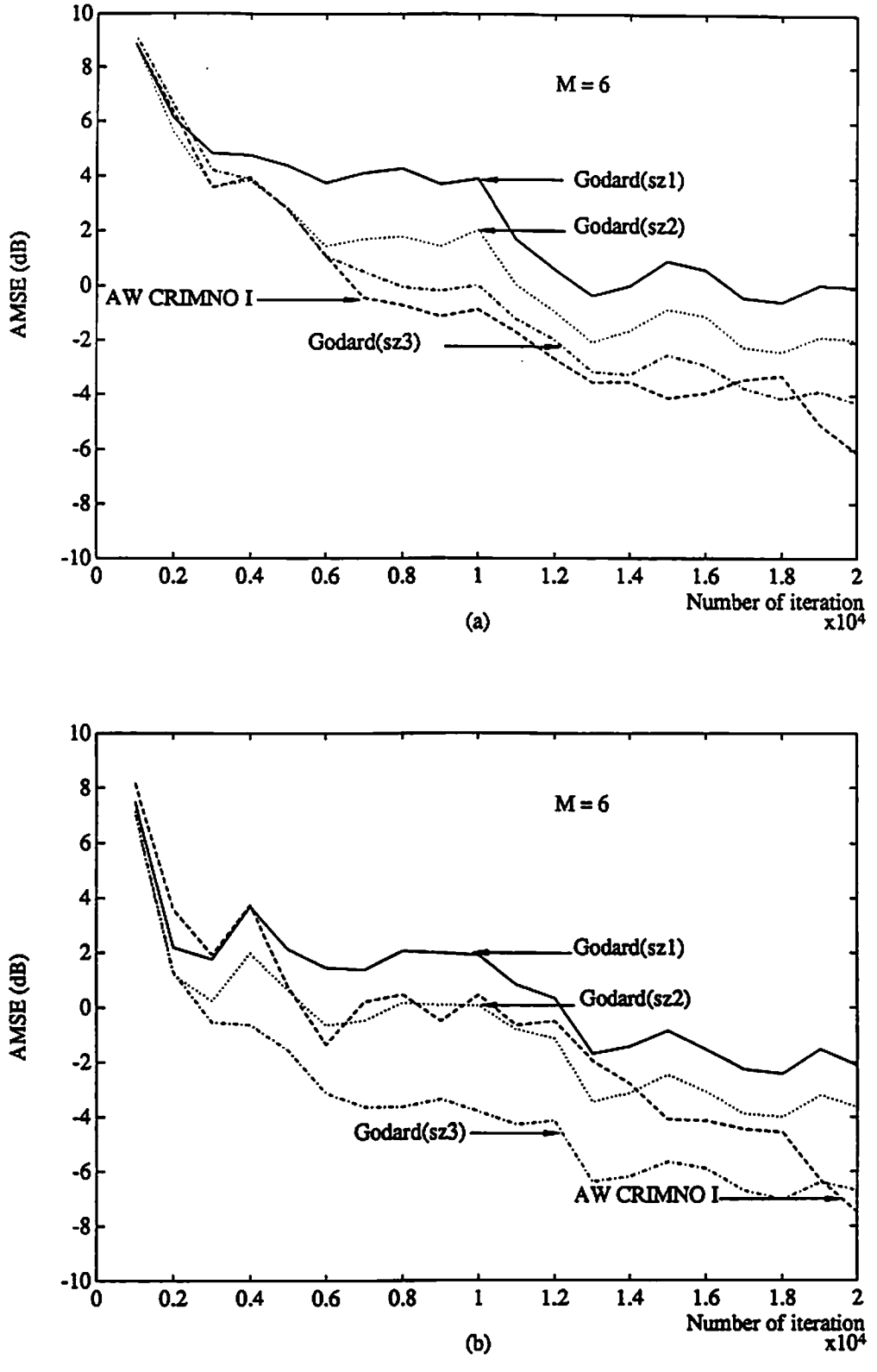
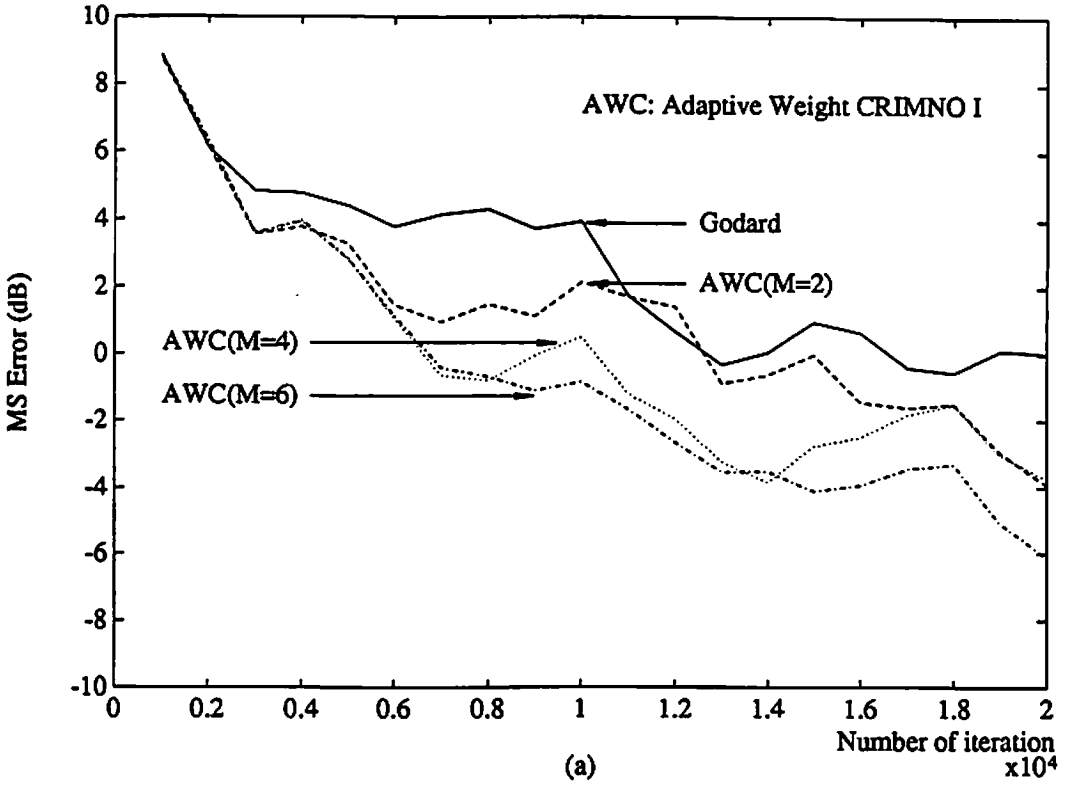
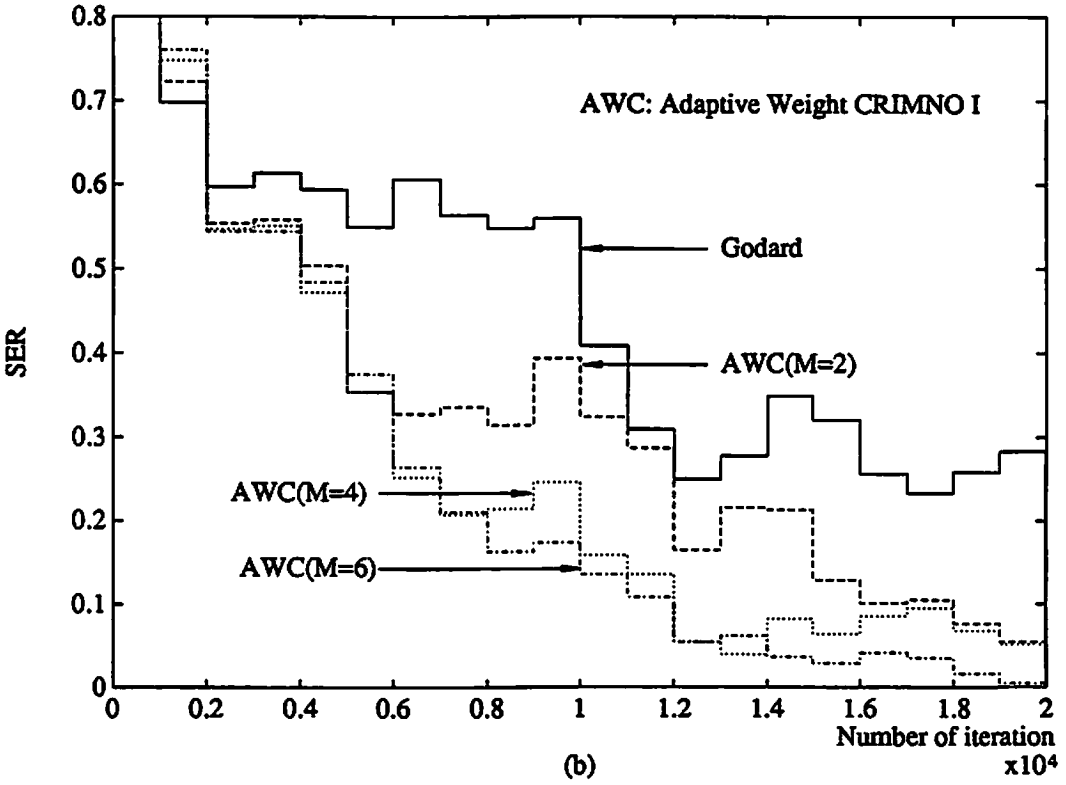


Figure 4.6 Comparison of the adaptive weight CRIMNO algorithm (szl) with Godard's algorithm of different step-size (sz3 is the optimum step-size): (a) the real channel; (b) the synthetic channel.


AWC: Adaptive Weight CRIMNO I
Godard
AWC(M=2)
AWC(M=4)
AWC(M=6)
MS Error (dB)
Number of iteration
x10^4

(a)


AWC: Adaptive Weight CRIMNO I
Godard
AWC(M=2)
AWC(M=4)
AWC(M=6)
SER
Number of iteration
x10^4

(b)

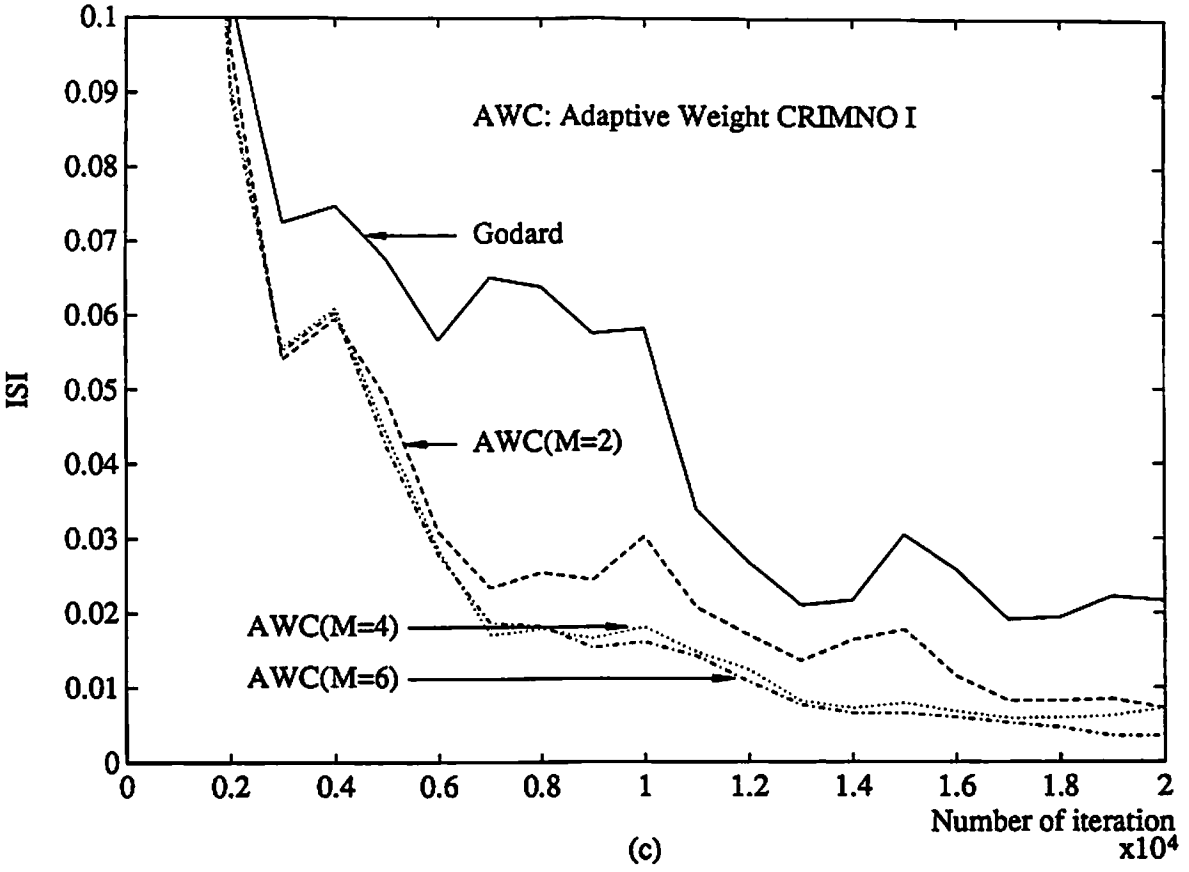


Figure 4.7 Effect of memory size M on the adaptive weight CRIMNO algorithm: (channel 2) (a) Mean square error; (b) Probability of error; (c) Intersymbol interference.

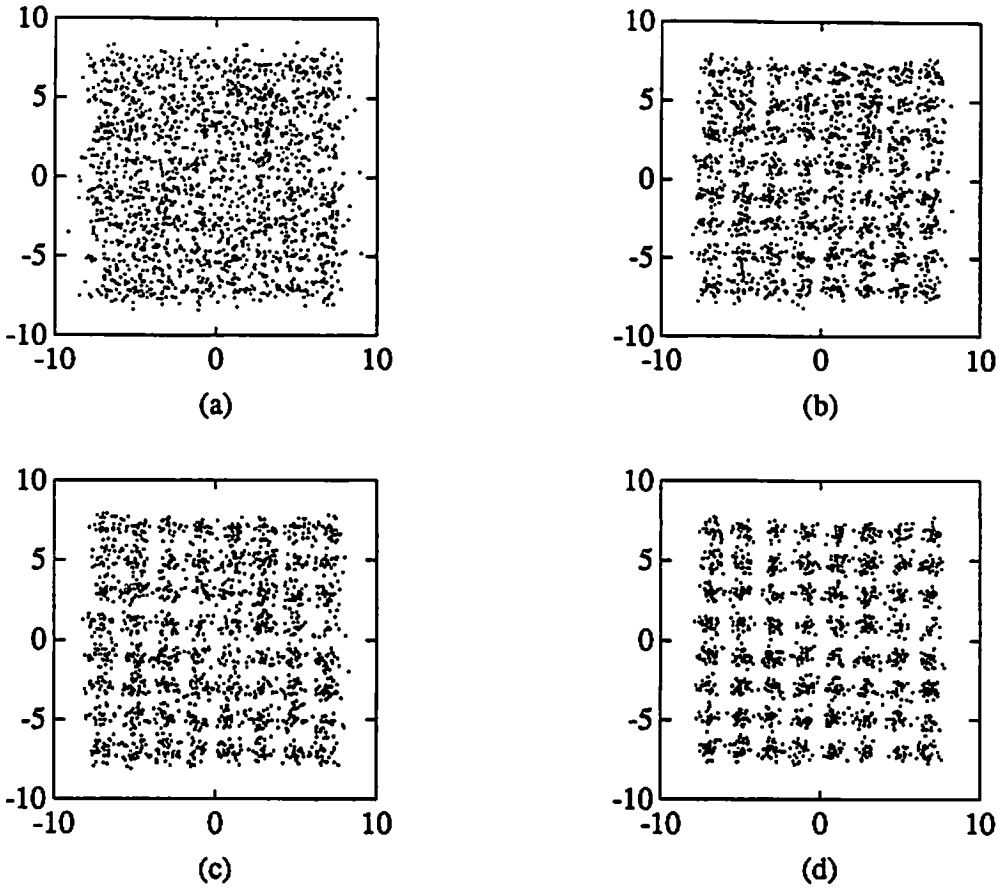


Figure 4.8 Eye pattern of adaptive weight CRIMNO algorithms with different memory size M at iteration 20000. (a) Godard; (b) Adaptive weight CRIMNO ($M=2$); (c) Adaptive weight CRIMNO ($M=4$); (d) Adaptive weight CRIMNO ($M=6$).

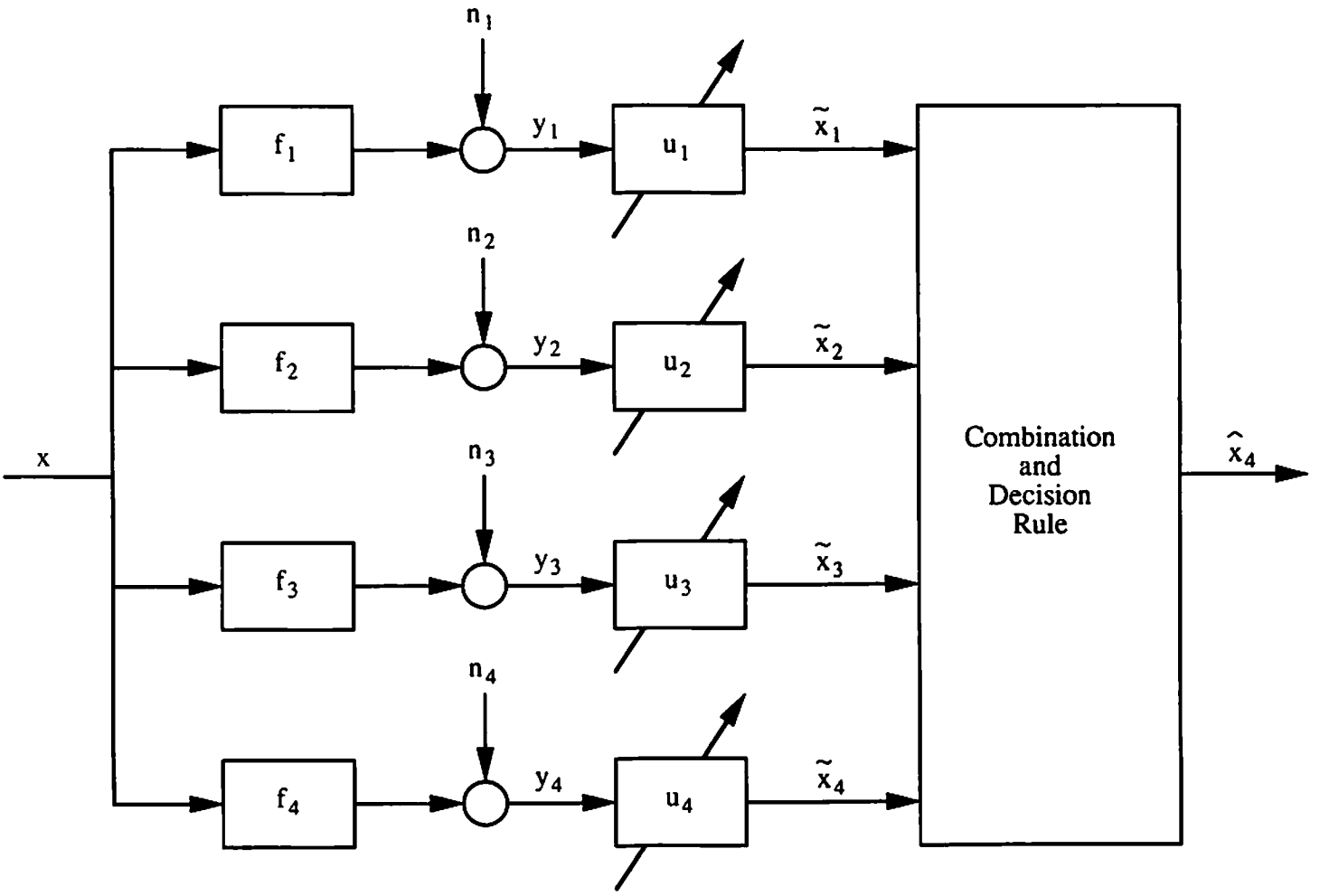


Figure 5.1 Diagram of four parallel equalized systems with additive noise.

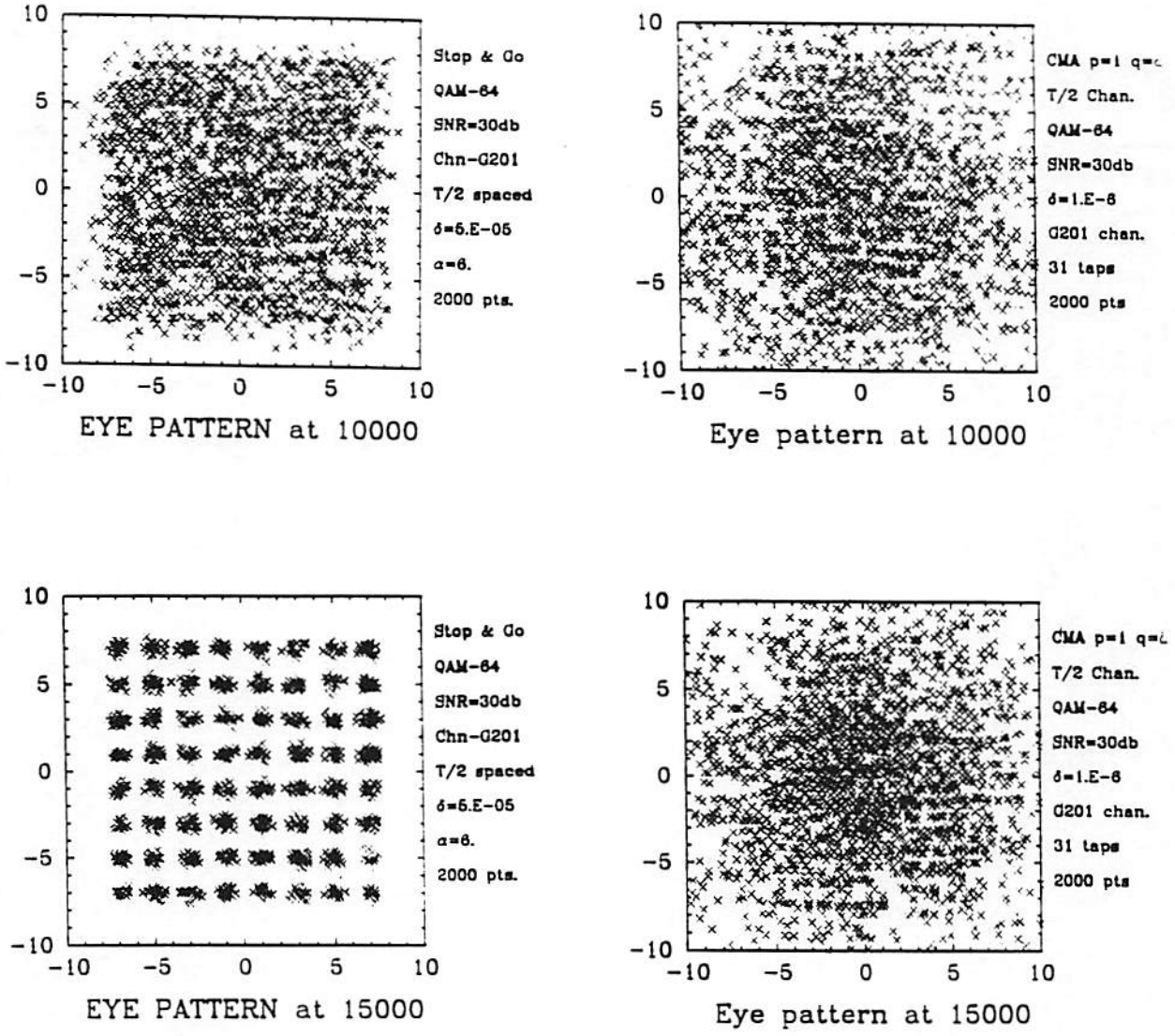


Figure 5.2 Channel G201 with QAM-64: Stop-and-Go and CMA (p=1, q=2) algorithms.

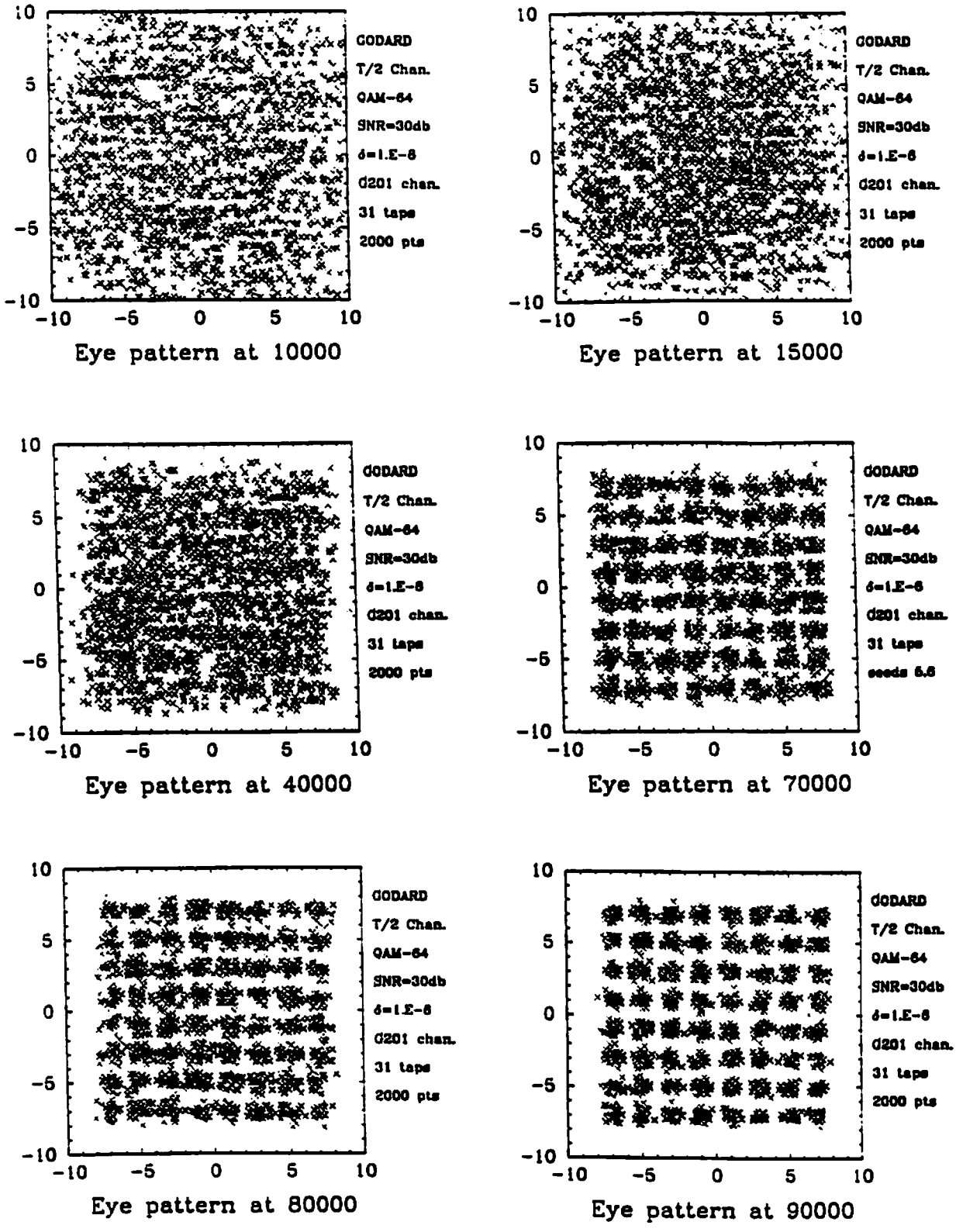


Figure 5.3 Channel G201 with QAM-64: Godard algorithms.

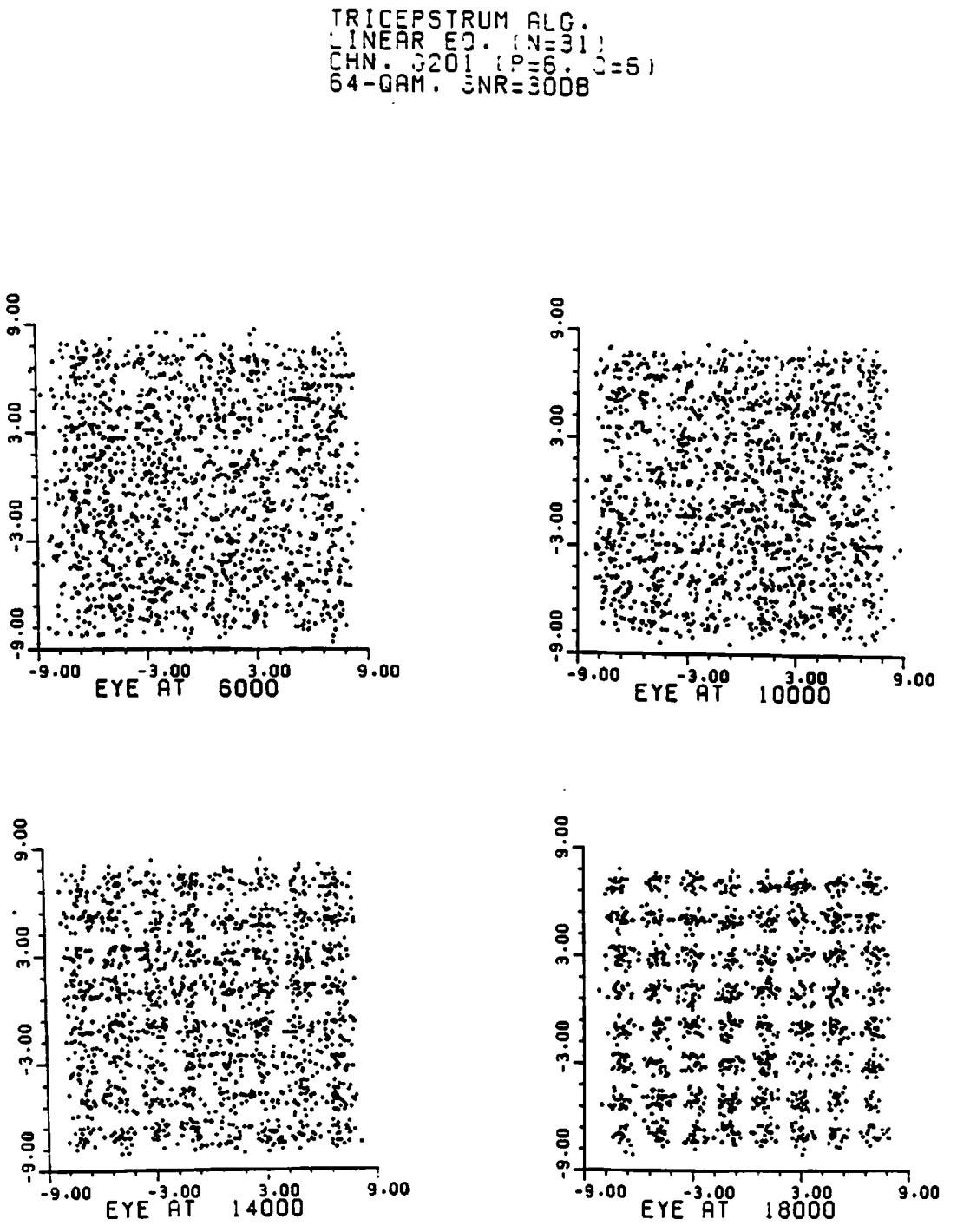


Figure 5.4 Channel G201 with QAM-64: TEA algorithms.

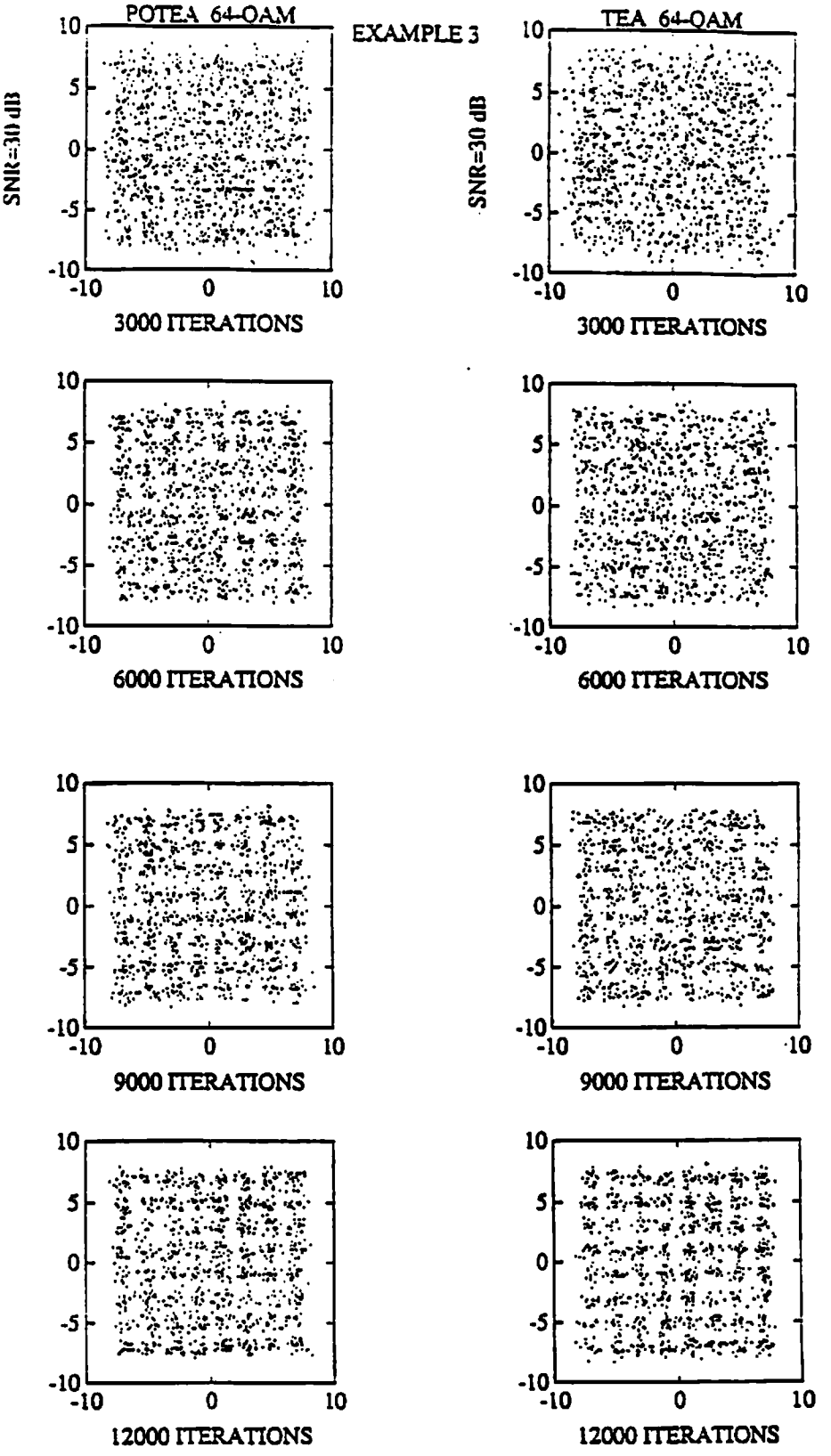


Figure 5.5 Eye-patterns of the Godard algorithm vs. those of the TEA algorithm.

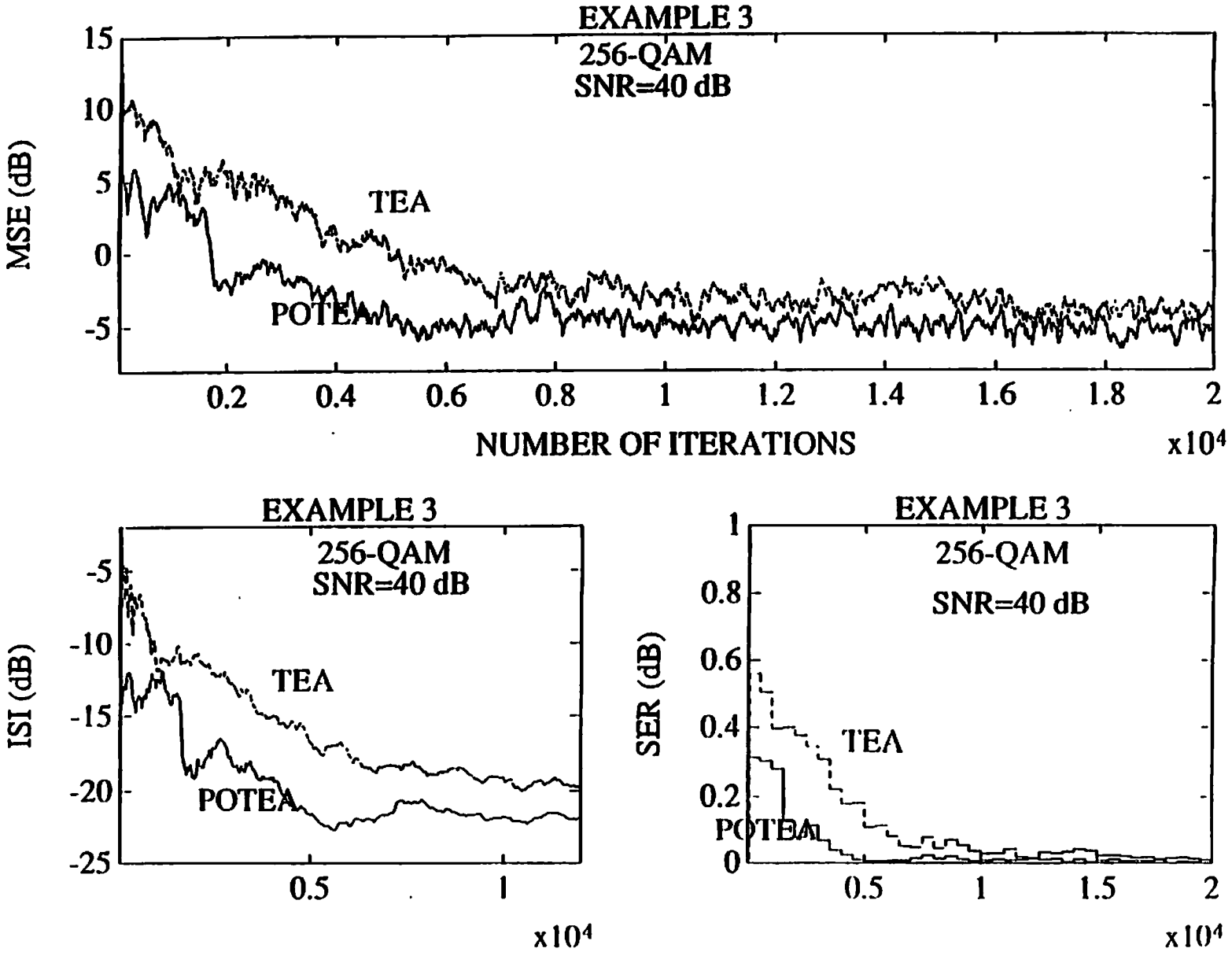


Figure 5.6 Performance comparison of the POTEA algorithm with the TEA algorithm.

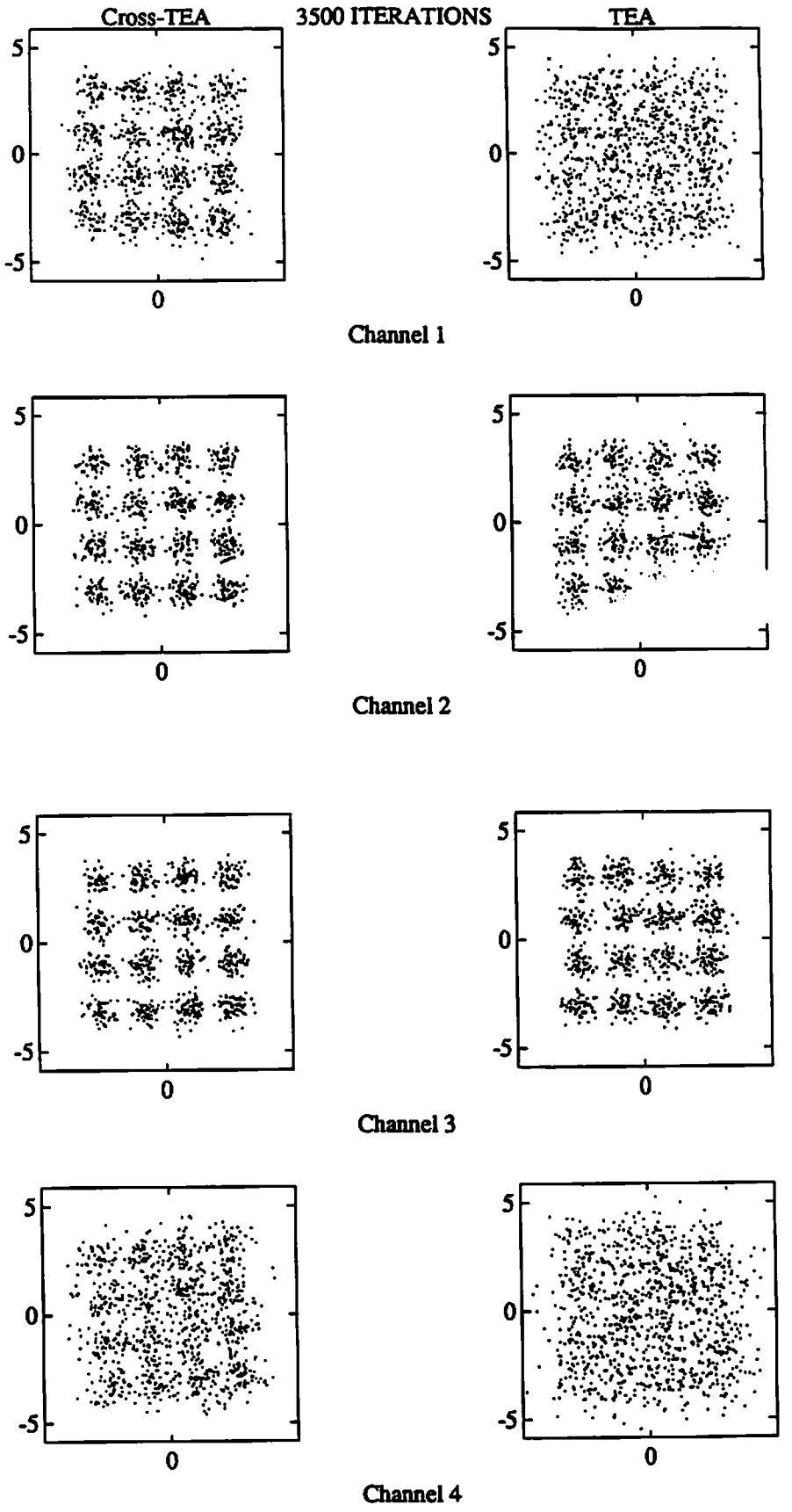


Figure 5.7 Eye Diagrams for CTEA and TEA at 3500 iterations..

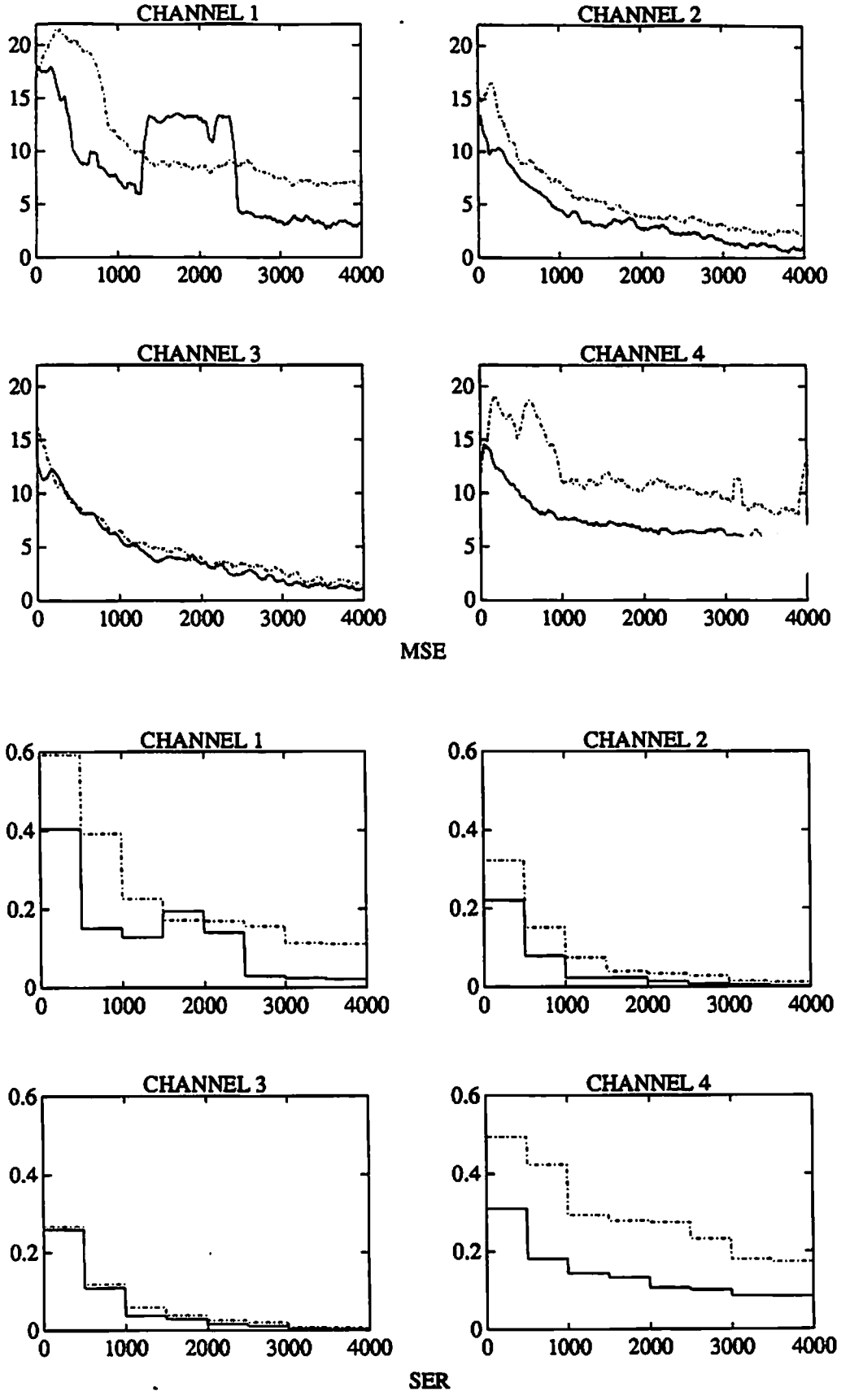


Figure 5.8 MSE and SER plots for all four channels, CTEA vs. TEA.