University of Southampton

Faculty of Engineering

Department of Electrical and Computer Science

Southampton SO17 1BJ

"Blind Equalisation Schemes for Dispersive Stationary and Mobile Channels"

by

Spyros Vlahoyiannatos

Dipl Eng

A doctoral thesis submitted in partial fulfilment of the requirements for the award of Doctor of Philosophy of the University of Southampton

30 April 2001

Supervisor: Professor Lajos Hanzo

Dipl Eng, Msc, PhD, SMIEEE

Chair of Telecommunications

Department of Electronics and Computer Science

University of Southampton

Southampton SO17 1BJ

United Kingdom

University of Southampton

Abstract

Faculty of Engineering

Department of Electronics and Computer Science

Doctor of Philosophy

Blind Equalisation Schemes for Dispersive Stationary and Mobile Channels

by Spyros Vlahoyiannatos

In this thesis the problem of blind equalisation of stationary and fading mobile channels is investigated. Blind equalisers achieve a bandwidth economy by eliminating the training overhead, often invoked in trained channel equalisation. The family of blind equalisers found in the literature is critically appraised and the achievable performance is evaluated by computer simulations. We focus our attention on the set of per–survivor processing based techniques as well as on the Bussgang techniques. A specific application is also considered in the context of digital video broadcasting.

An extension of the well–known constant modulus algorithm, namely the DFE–CMA [1,2], is proposed for equalising channels having a long inverse impulse response. This algorithm is characterised in terms of its performance and convergence in comparison to the conventional constant modulus algorithm. It is found that the proposed algorithm outperforms the conventional constant modulus algorithm for transmissions over channels exhibiting impulse responses having equal–weight taps.

The extension of blind equalisation using channel decoding assisted feedback is also considered in the thesis. The corresponding modified per–survivor processing algorithm is iteratively invoked in order to provide improved joint data detection and channel estimation, by utilising the enhanced–reliability feedback of the channel decoder. The application of the M–algorithm is also considered in order to reduce the complexity of this blind turbo equalisation algorithm for transmissions over channels having a high number of CIR taps. The performance of this algorithm is evaluated by computer simulations and it is found that it improves the performance of non–iterative per–survivor processing, owing to the employment of the channel decoding feedback.

Finally, different channel coding techniques were invoked and benchmarked in conjunction with the above blind turbo–PSP equaliser. Specifically, a range of convolutional coding, turbo convolutional coding, trellis–coded modulation and turbo trellis–coded modulation assisted blind turbo–PSP schemes were comparatively studied.

Acknowledgments

I would like to take this opportunity to thank my supervisor, Professor Lajos Hanzo, for his help and guidance during the whole of my doctorate. Lajos played a very important part in every stage of this path and I am especially grateful for his support.

Also I would like to thank my colleagues, C. S. Lee and S. X. Ng for their co-operation and help throughout the course of my PhD as well as Peter Cherringham for managing our computer network and Denise Harvey for her good administrative work.

The financial support of the various financial supporters of the Southampton university communications group are also gratefully acknowledged.

Last but not least I would like to thank the people who did not take part in the technical side of my PhD but surrounded me and accompanied me with their own special way in this long and exciting trip.

List of Publications

1. C. S. Lee, S. Vlahoyiannatos and L. Hanzo, "Satellite Based Turbo–Coded, Blind–Equalised 4-QAM and 16-QAM Digital Video Broadcasting", *IEEE Transactions on Broadcasting*, Vol 46, pp 23–33, March 2000.

2. S. Vlahoyiannatos and L. Hanzo, "Blind Equalisers: A Comparative Study", submitted to the *Proceedings of the IEEE.*

3. C. S. Lee, S. Vlahoyiannatos and L. Hanzo, "Blind–Equalised and Turbo–Coded Satellite Based Digital Video Broadcasting", *ICT 2000*, Acapulco, 22–25, pp 562–566, May 2000.

4. L. Hanzo, W. Webb and T. Keller, "Single and Multi–carrier Quadrature Amplitude Modulation", 3^{rd} ed., Wiley, 2000, section: "Overview of Blind Equalisers" pp 170–226.

5. S. Vlahoyiannatos and L. Hanzo, "Soft Decision–Feedback equalization using the Constant Modulus Algorithm", submitted to the *European Transactions on Telecommunications.*

6. S. Vlahoyiannatos and L. Hanzo, "Blind PSP–based turbo equalization", to appear in the *Proceedings of the Vehicular Technology Conference 2001*, Spring, Rhodes, Greece, 6–9 May 2001.

7. S. Vlahoyiannatos, S. X. Ng and L. Hanzo, "Blind PSP–Based Turbo Equalization Using Coded Modulation", to appear in the *Proceedings of the European Conference on Communications 2001*, Bratislava, 5–7 July 2001.

8. S. Vlahoyiannatos and L. Hanzo, "Constant Modulus Algorithm Assisted Soft Decision Feedback Equalisation", to appear in the *Proceedings of the ICC 2001*, Helsinki, Finland, 11–15 June 2001.

List of Symbols

- t : Continuous time variable
- n : Discrete time variable
- $a()$: Transmitted symbols
- $x()$: Transmitted signal (continuous sense)
- $y()$: Received signal (discrete sense)
- $h()$: Channel impulse response
- $e()$: Additive noise signal
- $X()$: Transmitted signal spectrum
- $Y()$: Received signal spectrum
- $H()$: Channel transfer function in the frequency domain
- $E()$: Additive noise signal's spectrum
- $z()$: Equalised signal
- $\mathbf{a}(n)$: Transmitted signal vector at time instant n
- $\mathbf{y}(n)$: Received signal vector at time instant n
- $\mathbf{e}(n)$: Noise vector at time instant n
- $\mathbf{c}^{(n)}$: Equaliser vector at iteration n
- $\mathbf{w}^{(n)}$: Feedback equaliser vector at iteration n
- $\mathbf{H}(n)$: Channel impulse response matrix at time instant n
- $\mathbf{t}(n)$: Combined channel plus equaliser impulse response vector at time instant n

- $\mathbf{z}(n-1)$: Equalised symbol vector
- $\mathbf{C}$: Vector derivative of the equalised symbol vector with respect to the feedforward equaliser vector
- $\mathbf{W}$: Vector derivative of the equalised symbol vector with respect to the feedback equaliser vector
- λ : Step–size parameter of the equaliser coefficients update
- δ : Delay in terms of the number of symbols
- $\epsilon(n)$: Error term at time instant n
- $b(i,j)$: Autocorrelation of the received signal
- $\Psi(i)$: Correlation of the received signal with the transmitted signal sequence.
- $\mathbf{\Phi}$: Weighted autocorrelation matrix of the received signal sequence
- $\mathbf{P}$: Inverse of the received signal's autocorrelation matrix
- $\mathbf{g}(n)$: Weighted correlation between the transmitted and received signal sequences
- $\mathbf{k}(n)$: Kalman gain vector
- $\xi(n)$: Recursive Least Squares (RLS) error term
- $\beta(n,i)$: RLS weighting factor (also known as "exponential weighting factor" or "forgetting factor")
- w : RLS weighting factor parameter
- $g(z(n))$: Non–linear Bussgang function
- $J(n)$: Blind equalisation error cost–function
- γ : Sato's scaling coefficient
- R_p : Constant Modulus Algorithm (CMA) constant
- R_S : Signed–CMA constant
- $K(z)$: Kurtosis of signal $z(n)$
- $f_{n,R}$: Stop-and-Go variable for deciding about taking the real part of the update into account

- $f_{n,I}$: Stop-and-Go variable for deciding about taking the imaginary part of the update into account
- $R_{2,R}, R_{2,I}$: Stop–and–Go agorithm's constants
- k_1, k_2 : Benveniste–Goursat algorithm's constants
- M : Number of states in the M–algorithm
- $L()$: Cumulant function
- $d(n)$: QAM constellation points
- Δ : Oversampling factor
- s_n : Per–Survivor Processing (PSP) state at time n
- $c_x(m, n, l)$: Tricepstrum of signal x
- $\hat{x}$: estimate of x
- $\mathbf{A}^T$: Transpose of matrix $\mathbf{A}$
- $\mathbf{A}^*$: Conjugate of matrix $\mathbf{A}$
- $\mathbf{A}^H$: Hermitian of matrix $\mathbf{A}$
- $\mathbf{A}^{-1}$: Inverse of square matrix $\mathbf{A}$
- $\mathbf{A}^t$: Pseudo–inverse matrix of the non–square matrix of $\mathbf{A}$
- $[\mathbf{A}]_{i,j}$: Element of the matrix $\mathbf{A}$ at position i, j
- $||\mathbf{x}||^2$: L_2–norm of vector $\mathbf{x}$
- μ_n : n-th order moment of the transmitted symbols' magnitude distribution
- N : Equaliser order
- L : Channel order
- Q : Number of QAM constellation symbols
- K : Bits per symbol in the QAM constellation
- $\alpha_k, \beta_k, \gamma_k$: Previous, current and next state probabilities for the MAP algorithm's implementation

- E_a : Average signal power
- E_e : Average noise power

Contents

Chapter 1

Trained and Blind Equalisation

Prologue

In mobile communications the channel is hostile, containing dispersive multipath components, and its characteristics are perpetually changing due to the mobility of the user. The signal, which is transmitted over this time-variant channel suffers from fast as well as shadow fading and it is contaminated by the *Intersymbol Interference (ISI)* inflicted by multipath propagation. In order to be able to demodulate the signal, we have to remove the ISI. This can be achieved either by convolving the received signal with an appropriate sequence, namely with the channel equaliser's impulse response or by estimating the Channel's Impulse Response (CIR) and then restoring the original transmitted signal by 'reversing' the channel's effects with the aid of *channel estimation*) . Both of these methods will be highlighted in more depth during our further discourse in the context of blind equalisation.
The conventional method of channel equalisation is performed by means of estimating the CIR with the aid of a "known" sequence, which is also known at the receiver and is transmitted periodically. This sequence is referred to as the *training sequence*, which constitutes a transmission overhead, because it could have been used for transmitting useful information. For example, in GSM, 26 bits are used for equaliser training in each burst containing 116 information bits, which results in an overhead of 22.4%. Moreover, in mobile communications, due to the mobility of the user terminal, occasionally the connection can be lost completely. In this case, the need for equaliser adaptation "from scratch" arises, which means that there is no information at all about the CIR, until the next transmission burst arrives at the receiver. A blind equaliser may be more suitable for this situation, because it is designed to adapt to an unkown CIR without any information about it, which could be used for the initialisation of the channel equaliser. For example, in a multipoint network, every time a network connection between two network points is down, it is inefficient to send a training sequence to restore

this connection.

Substantial research efforts have been dedicated to blind equalisation in an effort to avoid the use of a training sequence, resulting in so-called *blind equalisation* techniques, which extract the appropriate information required for channel equalisation from the received signal. A range of different methods have been proposed by numerous authors for blind equalisation using various equaliser structures [3–7]. As the need for blind equalisers has grown in recent years, modifications of these basic algorithms have been proposed, which are referred to in the references.

The organisation of the thesis is as follows:

- In the first chapter we are introducing the problem of channel equalisation. The chapter is divided into two sections. In the first section trained equalisation techniques are introduced and discussed, while in the second section a rudimentary introduction to basic blind equalisation schemes is offered. A convergence study and a study on the equaliser parameter control are also given.

- In Chapter 2 comparative performance results are given for some of the blind equalisers discussed in Chapter 1 and performance comparison conclusions are drawn. An application of blind equalisers to digital video broadcasting is also considered, extending the specifications system to include turbo coding and higher–order modulation along with blind equalisation in order to increase the system's robustness while increasing the bandwidth and coping with more severe channel distortion.

- In Chapter 3 a decision feedback assisted extension of the constant modulus algorithm is proposed and studied. Performance results show that this algorithm provides better performance than the conventional constant modulus algorithm in equalising channels with long inverse impulse response.

- In Chapter 4 an iterative technique combining convolutional channel coding with per–survivor processing is proposed. This technique enhances the performance of per survivor processing by iteratively providing a–priori feedback from the channel decoder back to the equaliser's input.

- In Chapter 5 the turbo–PSP technique of Chapter 4 is extended by replacing the convolutional channel coding with coded modulation techniques. Performance results in this chapter indicate that the fading channels performance is improved.

- Finally, in Chapter 6 conclusions are drawn on the pre–existing as well as the novel research results of this thesis and further research proposals are presented.

Next, we outline the novel contributions of this thesis:

- An illustrative overview of previously proposed blind equalisation algorithms is presented together with a performance evaluation and comparison in terms of performance and complexity [8,9]. We evaluated the performance gain of switching to decision–directed equalisation after blind convergence. The parameter control issue of these algorithms is studied, focusing on the constant modulus algorithm and the relationship between the equaliser's parameters and its performance in terms of convergence speed and precision was evaluated experimentally. The convergence of Bussgang equalisers is also studied, confirming and providing an insight to previously published results. Finally an application of blind equalisers to a digital broadcasting application [10, 11] was proposed and studied, improving the overall system performance and proposing an extension of the system's specifications.

- The DFE–CMA [1, 2] is proposed, providing an extension to the convential constant modulus algorithm, which offers improved performance for channels with long inverse impulse response. A convergence analysis shows that under a certain assumption the convergence region of this algorithm does not include the length–dependent local minima of the Bussgang equalisers.

- The turbo–PSP algorithm [12] is proposed, combining per survivor procesing with turbo equalisation. Per survivor processing is extended to include a–priori input provided iteratively by a convolutional channel decoder. The performance of a per survivor processing equaliser is therefore improved by the channel decoder's output. Pilot phase estimation is proposed for removing the phase ambiguity associated with blind equalisers. It is shown that with only a minor proportion of symbols in the order of 2% we can achieve negligible performance degradation, compared to perfect phase estimation.

- The turbo–PSP algorithm is extended to employ coding modulation techniques [13] in substitution of convolutional coding. Different channel coding techniques were compared and the results show that by replacing convolutional coding with trellis coded modulation we can improve the fading channel performance of turbo–PSP.

1.1 Introduction to Equalisation

1.1.1 Introduction

Multipath propagation is a common problem arising in digital communications. Its main cause can be viewed as the signal arriving to the receiver from more than one paths as a result of multiple

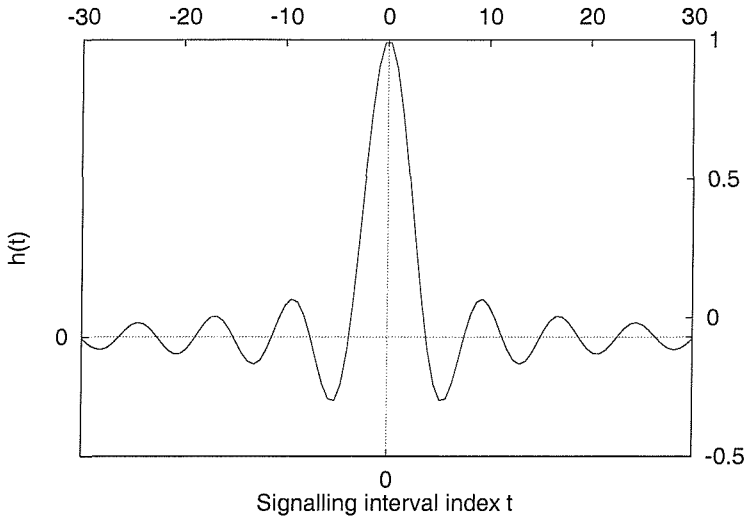


Figure 1.1: Stylised channel impulse response $h(t)$

reflections. Assuming that $X(f)$ is the spectrum of the transmitted signal $x(t)$, $H(f)$ is the spectral domain transfer function of the channel, $Y(f)$ is the received signal's spectrum and $E(f)$ is the spectrum of the Additive White Gaussian Noise (AWGN) we can form the equation:

$$Y(f) = H(f) \cdot X(f) + E(f). \tag{1.1}$$

Upon translating Equation (1.1) to the time domain we have:

$$y(t) = h(t) * x(t) + e(t), \tag{1.2}$$

where $*$ stands for convolution. The impulse response of a typical mobile channel may appear like the stylised function given in Figure 1.1. The convolution of $h(t)$ with the transmitted signal results in intersymbol interference (ISI) since not only the current transmitted bits, but also previous bits influence the received signal at any instant. This situation is undesirable because, besides the additive noise, the signal received during the current signalling interval is corrupted by the signals due to the bits transmitted during both previous and past signalling intervals. These neighbouring signals constitute a second noise–like term, referred to as *Convolutional Noise*.
Using sampled values at time instant $t = nT$ Equation (1.2) takes the form:

$$\{y(n)\} = \{h_n\} * \{x(n)\} + \{e(n)\}, \tag{1.3}$$

where "*" means convolution. In order to restore the original signal sequence $\{x(n)\}$ from the received sequence $\{y(n)\}$, we can convolve $\{y(n)\}$ with another sequence $\{c_n\}$ mimicking the 'inverse' of the

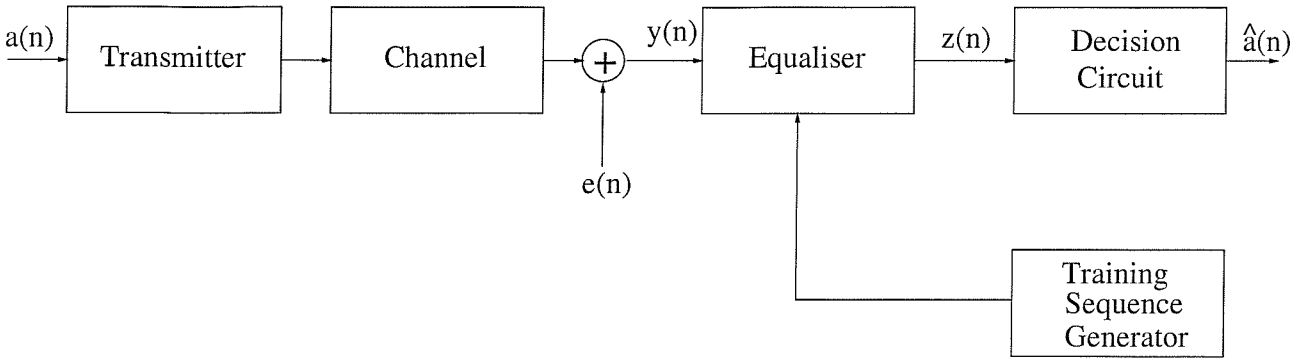


Figure 1.2: A simplified equalised system

channel's effects, yielding:

$$\{z(n)\} = \{c_n\} * \{h_n\} * \{x(n)\} + \{c_n\} * \{e(n)\}. \tag{1.4}$$

A number of methods have been suggested for determining the sequence $\{c_n\}$, when a–priori information in the form of an equaliser training sequence is available, which will be discussed later. Figure 1.2 shows a typical trained equaliser of this kind, where the knowledge of the training sequence is exploited by the equaliser at the receiver. In this scheme, the training sequence is transmitted periodically for the duration of certain time intervals. During these time intervals the training bits are used by the equaliser for estimating the optimum equaliser coefficients, which will then be used to equalise the channels's impact on the information bits before deciding upon the binary value of the most likely transmitted bit. In the following sections, we present typical trained equalisers.

1.2 Linear Trained Equalisers [14], [15]

The general form of a linear equaliser is shown in Figure 1.3. As we can see from Figure 1.3, the decision symbol $\hat{a}(n)$ at time n is based on the equalised symbol $z(n)$, which is formed by the suitably weighted successive signal values $y(n-N), \cdots, y(n+N)$, where $2N+1$ is the equaliser's length, followed by a decision circuit. Hence the equaliser is capable of cancelling the channel's effects over a duration of $2N+1$ consecutive symbol intervals. In mathematical terms, the equalised symbol at time n is given by:

$$z(n) = \sum_{i=-N}^{N} c_i \cdot y(n-i). \tag{1.5}$$

The equaliser weights are adjusted so as to minimise the multipath channel's distortions, based on a certain minimisation criterion. Depending on the specific criterion they satisfy, linear trained equalisers can be divided into a number of categories, which are presented in the next sections.

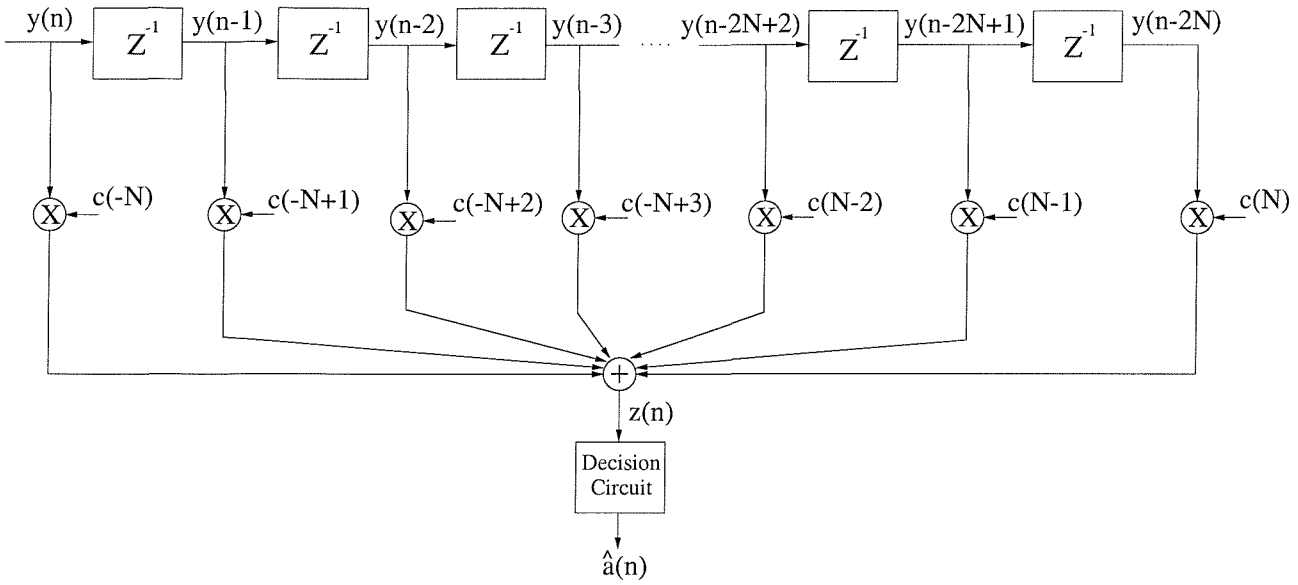


Figure 1.3: Typical linear equaliser

1.2.0.1 *Zero–Forcing* Equalisers

According to the so–called Zero–Forcing (ZF) approach [15], [9] we estimate the channel's impulse response (CIR) h_n and then attempt to cancel its effects, producing a new sequence c_n so that:

$$\{c_n\} * \{h_n\} = \{1, 0, 0, \ldots\}, \tag{1.6}$$

where $\{\}$ denotes a set and the equal sign here denotes equality between sets. If Equation (1.6) holds, then the system is perfectly equalised having no residual ISI. According to this approach, the channel is regularly sounded by transmitting a specific training sequence also known at the receiver and an estimate of the CIR coefficients $\{h_n\}$ is obtained. Considering $H(z)$ and $C(z)$ as the z–transform of the signals $\{h_n\}$ and $\{c_n\}$, respectively, and following the procedure described in [9] we obtain:

$$\begin{aligned} C(z) \cdot H(z) &= 1 \Rightarrow C(z) = \frac{1}{H(z)} \Rightarrow \\ \sum_{i=0}^{n} \hat{c}_m z^{-m} &= \frac{1}{\sum_{i=0}^{n} h_m z^{-m}} \Rightarrow \\ \hat{c}_0 &= \frac{1}{h_o} \\ \hat{c}_m &= -\frac{\sum_{i=0}^{m-1} \hat{c}_i h_{m-i}}{h_o}, \end{aligned} \tag{1.7}$$

where $\hat{c}$ denotes an estimate of c and Equation (1.7) is an approximation applicable to finite–duration CIRs. Estimates of the CIR $\{h_n\}$ can be obtained by sounding the channel at regular intervals. Zero–forcing equalisers can be used in low–noise environments, when the ISI is the main factor in signal

corruption. However, in their attempt to invert the channel's transfer function they neglect the effect of noise [9], which renders them unsuitable for high–noise environments. A more suitable equaliser for such environments is highlighted in the next section.

1.2.0.2 *Least Mean Squares* Equalisers

The Least Mean Squares (LMS) equalisers [14], [15] use the equaliser training sequence for estimating the equaliser coefficients, so as to minimise the mean squared value of the error between the signal $z(n)$ at the input of the decision device and the transmitted symbol $a(n)$:

$$\epsilon(n) = z(n) - a(n) \tag{1.8}$$

yielding:

$$MSE^{(LMS)} = E[|\epsilon(n)|^2], \tag{1.9}$$

where $E[]$ is the expectation over all possible transmitted $\{a(n)\}$ and received $\{\hat{a}(n)\}$ data sequences. We can differentiate the Mean Squared Error (MSE) term in Equation (1.9) with respect to the tap–vector $\mathbf{c}$ of the equaliser, in order to obtain the tap–values of the equaliser tap vector $\mathbf{c}$ that minimise the MSE. Assuming an equaliser constituted by $2N+1$ taps, ranging from $-N$ to N, we obtain $\hat{\mathbf{c}}$ as a solution of the above linear system as [9]:

$$\sum_{i=-N}^{N} \hat{c}_i \cdot b(i,j) = h_{-j}, \quad j = -N, \ldots, N, \tag{1.10}$$

where

$$b(i,j) = \sum_{n=-\infty}^{+\infty} y(n-i) \cdot y(n-j), \quad i,j = -N, \ldots, N \tag{1.11}$$

is the autocorrelation function of the sequence $\{y(n)\}$. In matrix form, Equations (1.10) can be rewritten as:

$$\mathbf{R_{yy}} \cdot \hat{c} = \mathbf{h}, \tag{1.12}$$

where $\mathbf{R_{yy}}$ is a matrix containing the coefficients $b(i,j)$:

$$\mathbf{R_{yy}} = \begin{pmatrix} b(-N,-N) & b(-N,-N+1) & \ldots & b(-N,N) \\ b(-N+1,-N) & b(-N+1,-N+1) & \ldots & b(-N+1,N) \\ \vdots & \vdots & \ddots & \vdots \\ b(N-1,-N) & b(N-1,-N+1) & \ldots & b(N-1,N) \\ b(N,-N) & b(N,-N+1) & \ldots & b(N,N) \end{pmatrix}$$

$\hat{\mathbf{c}}$ is the $(2N+1)$x1 dimensional vector of the estimated equaliser coefficients:

$$\hat{\mathbf{c}} = [\hat{c}_{-N}, \hat{c}_{-N+1}, \ldots, \hat{c}_{N-1}, \hat{c}_N]^T, \tag{1.13}$$

and $\mathbf{h}$ is the $(2N+1)$x1 vector of the channel's impulse response at time instant n:

$$\mathbf{h} = [h_N, h_{N-1}, \ldots, h_{-N}]^T. \tag{1.14}$$

The matrix $\mathbf{B}$ has a so–called *Töplitz* structure if the signal $y(n)$ is stationary, that is if its statistical properties are independent of the actual time instant of the observation, i.e.:

$$b(i+n, j+n) = b(i,j), \quad \forall n. \tag{1.15}$$

The optimum equaliser coefficients are given as the solution of the linear system (1.12). We can solve this system of equations by inverting matrix $\mathbf{B}$ or by using a recursive solution. In order to avoid solving a $(2N+1)$x$(2N+1)$ system of equations we can use an iterative procedure, which estimates the equaliser tap coefficients at each stage using, for example, the classical steepest descent algorithm [9] as follows:

$$\hat{\mathbf{c}}^{(k+1)} = \hat{\mathbf{c}}^{(k)} - \lambda \nabla_{\hat{\mathbf{c}}^{(k)}} J(n), \tag{1.16}$$

where $\nabla_{\hat{\mathbf{c}}^{(k)}}$ is the gradient vector with respect to the vector $\hat{\mathbf{c}}^{(k)}$, which is defined in Appendix C while $J(n)$ is the MSE function, as defined in Equation (1.9) and the $(k+1)$–th iteration is carried out for all the vector components $\hat{c}_i$ at the n–th time instant. This relationship leads to the final form of the classic LMS equaliser coefficient update algorithm, which is expressed as [14], [15]:

$$\hat{\mathbf{c}}^{(k+1)} = \hat{\mathbf{c}}^{(k)} - \lambda \mathbf{y}^*(n)\epsilon(n), \tag{1.17}$$

where $\mathbf{y}(n) = [y(n-N), y(n-N+1), \ldots, y(n+N)]^T$ physically represents $2N+1$ successive received signal samples around the sampling instant n. From Equation (1.17) we can observe that for the optimum equaliser coefficients, i.e. when the vector $\hat{\mathbf{c}}$ does not change from one iteration to another, the remaining update term should have zero mean value, i.e.:

$$E[\mathbf{y}^*(n)\epsilon(n)] = 0. \tag{1.18}$$

This implies that for the optimum equaliser coefficients the error signal $\epsilon(n)$ is uncorrelated to the received signal $y(n)$. The choice of the iteration step–size parameter λ in Equation (1.17) is important, since small values of λ result in slow, but accurate convergence, while larger values result in

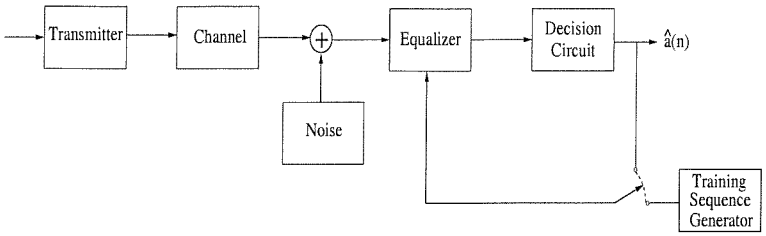


Figure 1.4: Decision–directed equalised system

faster but less accurate convergence of $\hat{\mathbf{c}}$ to its optimum value. The LMS approach is superior to the ZF technique because, besides the intersymbol interference, it can also combat the effect of noise, since the error term of Equation (1.8) takes into account both of these two factors of signal corruption.

1.2.0.3 *Decision–Directed* Equalisers

The general form of a Decision–Directed (DD) equalised receiver is shown in Figure 1.4. As it can be seen from this figure, the equaliser coefficients are constantly updated, even when the training sequence is not transmitted, by using the output of the equaliser after a decision has been made with respect to the most likely transmitted symbol. In other words, the decision device's noise–free output signal $\hat{a}(n)$ is used as a training sequence. This decision–directed approach can be very robust when the decisions are correct, that is if the signal to noise ratio is sufficiently high and the channel does not change rapidly, compared to the time interval between adjacent transmissions of the training sequence. When these two restrictions are satisfied, the use of the decision–directed scheme results in fast and robust convergence.

By defining the error as [9]:

$$\epsilon(n) = \sum_{n=-\infty}^{+\infty} \left(a(n) - \sum_{k=-N}^{N} \hat{c}_k y(n-k) \right)^2, \tag{1.19}$$

where $a(n)$ is the transmitted signal and $y(n)$ is the received signal at time instant n, we can find the equaliser coefficients, which minimise $\epsilon(n)$ as in [9]:

$$\sum_{k=-N}^{N} \hat{c}_k b(i,k) = \Psi(i), \qquad i = -N, \ldots, N, \tag{1.20}$$

where

$$\Psi(i) = \sum_{n=-\infty}^{\infty} a(n) y(n-i), \tag{1.21}$$

and $b(i,k)$ is the autocorrelation of $y(n)$, as defined in Equation (1.15). More explicitly, $b(i,k) = b|i-k|$ is considered to depend only on the difference $|i-k|$, implying that $y(n)$ is stationary, since we have assumed that the channel does not change rapidly. Clearly, $\Psi(i)$ is the correlation between the transmitted and received signals, namely $a(n)$ and $y(n)$. As with the other equalisers, we can either solve the linear system of equations (1.20) or use an iterrative procedure similar to (1.16) that of Equation (1.16).

Decision–directed equalisers are similar to blind equalisers, since in this type of equalisation no training sequence is available at the receiver, which therefore has to "guess" the original information symbols $a(n)$ in order to estimate the error and, accordingly, to adjust the equaliser coefficients.

1.2.0.4 *Recursive Least Squares* Equalisers

The so–called *Recursive Least Squares (RLS)* equalisers [14], [15] constitute a class of fast converging least squares equalisers, which use the incoming new data samples to update the old estimates of the equaliser coefficients. They use a computationally more demanding algorithm, than the classical steepest descent technique. We define the MSE term following Haykin [15] as:

$$MSE^{(RLS)} \equiv \sum_{i=1}^{n} \beta(n,i)|e(i)|^2, \tag{1.22}$$

where

$$e(i) = a(i) - z(i) \tag{1.23}$$

with $a(i)$ and $z(i)$ being the transmitted and the equalised received symbol at the input of the dicision device of Figure 1.2, respectively. More explicitly, the signal $z(i)$ is given by the convolution of the received signal and the equaliser coefficients as:

$$z(i) = \sum_{k=0}^{N-1} c_k^* y(i-k) = \left(\mathbf{c}^{(n)}\right)^H \cdot \mathbf{y}(i), \tag{1.24}$$

where

$$\mathbf{c}^{(n)} = [c_0^{(n)}, c_1^{(n)}, \ldots, c_{N-1}^{(n)}]^T \tag{1.25}$$

is the equaliser vector at time n, or equivalently at the nth iteration and

$$\mathbf{y}(i) = [y(i), y(i-1), \ldots, y(i-N+1)]^T \tag{1.26}$$

is the received signal vector. The weighting factor $\beta(n,i)$ is used to ensure that new samples are given more emphasis in the MSE function of Equation (1.22), than older ones. Therefore, a rational choice for $\beta(n,i)$ is a negative exponentially decaying weighting function [15]:

$$\beta(n,i) = w^{n-i}. \tag{1.27}$$

In order to find the optimum equaliser coefficients, which minimise the MSE term of Equation (1.22), we differentiate this error term with respect to the equaliser vector $\mathbf{c}$, following the procedure described in [9] or by Haykin in [15]. By doing so we arrive at:

$$\mathbf{\Phi}(n) \cdot \hat{\mathbf{c}}^{(n)} = \mathbf{g}(n), \tag{1.28}$$

where

$$\mathbf{\Phi}(n) = \sum_{i=1}^{n} w^{n-i} \cdot \mathbf{y}(i) \cdot \mathbf{y}^{H}(i), \tag{1.29}$$

or more explicitly:

$$\mathbf{\Phi}(n) = \sum_{i=1}^{n} w^{n-i} \cdot \begin{pmatrix} y(i) \cdot y^*(i) & \dots & y(i)^* \cdot y(i-N+1) \\ y(i-1) \cdot y^*(i) & \dots & y(i-1) \cdot y^*(i-N+1) \\ \vdots & \ddots & \vdots \\ y(i-N+2) \cdot y^*(i) & \dots & y(i-N+2) \cdot y^*(i-N+1) \\ y(i-N+1) \cdot y^*(i) & \dots & y(i-N+1) \cdot y^*(i-N+1) \end{pmatrix}, \tag{1.30}$$

$$\mathbf{g}(n) = \sum_{i=1}^{n} w^{n-i} \mathbf{y}(i) a^*(i) \Rightarrow \mathbf{g}(n) = \sum_{i=1}^{n} w^{n-i} \cdot a^*(i) \begin{pmatrix} y(i) \\ y(i-1) \\ \vdots \\ y(i-N+2) \\ y(i-N+1) \end{pmatrix} \tag{1.31}$$

and $\hat{\mathbf{c}}^{(n)}$ is the RLS estimate of the equaliser vector $\mathbf{c}^{(n)}$, as defined in Equation (1.25). In order to estimate the equaliser tap gains $\hat{\mathbf{c}}^{(n)}$ we can invert the matrix $\mathbf{\Phi}(n)$, yielding:

$$\hat{\mathbf{c}}^{(n)} = \mathbf{\Phi}^{-1}(n) \cdot \mathbf{g}(n). \tag{1.32}$$

Following the detailed considerations and calculations presented in [15] or [9], which are reproduced in Appendix B, we arrive at the RLS algorithm as:

$$\hat{\mathbf{c}}^{(n)} = \hat{\mathbf{c}}^{(n-1)} + \mathbf{k}(n) \cdot \xi^*(n), \tag{1.33}$$

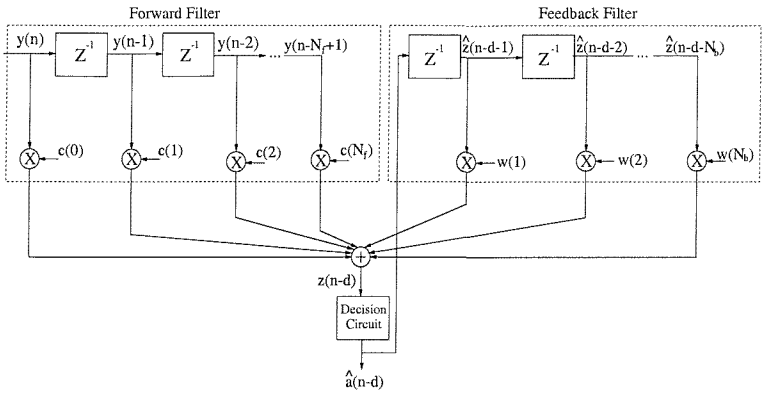


Figure 1.5: A typical Decision–Feedback Equaliser

where

$$\mathbf{k}(n) = \frac{w^{-1}\mathbf{\Phi^{-1}}(n-1)\mathbf{y}(n)}{1 + w^{-1}\mathbf{y}^H(n)\mathbf{\Phi^{-1}}(n-1)\mathbf{y}(n)} \tag{1.34}$$

$$\xi(n) = a(n) - \left(\hat{\mathbf{c}}^{(n-1)}\right)^H \mathbf{y}(n) \tag{1.35}$$

$$\mathbf{\Phi^{-1}}(n) = w^{-1}\mathbf{\Phi^{-1}}(n-1) - w^{-1}\mathbf{k}(n)\mathbf{y}^H(n)\mathbf{\Phi^{-1}}(n-1). \tag{1.36}$$

The matrix $\mathbf{\Phi^{-1}}(n)$ is the estimate of the inverse of $\mathbf{\Phi}(n)$ at the n–th time interval. Clearly, matrix additions and multiplications render this technique more complex, than the previous ones presented so far. In return, this equaliser converges to the optimum solution significantly faster, than the previously discussed ones. However, when applying the RLS algorithm in rapidly fading channel environments, its channel tracking capability is dictated by the duration of the time-domain window employed.

1.2.0.5 *Decision–Feedback* Equalisers

In automatic control systems it is desirable to use a negative feedback–based structure, since it improves their tolerance to noise. This happens because the negative feedback section generally decreases the system's gain–bandwidth product and consequently reduces the noise at the output of the system proportionately to the reduced bandwidth.

Decision–Feedback equalisers (*DFEs*) [14], [15] constitute a modification of their linear counterparts. These equalisers, besides the feedforward filter seen in Figure 1.3, also employ a feedback filter similar to the feedforward filter, but using as their input the noise–free output of the decision device. A general DFE structure is shown in Figure 1.5. Assuming that the feedforward equaliser coefficients $\hat{c}_i$ have already been computed, using for example one of the previously mentioned techniques, we can estimate the feedback coefficients b_i, which minimise the error. We express the error between the equalised signal $z(n)$ at the input of the decision device and the estimate of the transmitted signal

$a(n)$ as:

$$\epsilon(n-\delta) = E\left[|z(n-\delta) - a(n-\delta)|^2\right], \tag{1.37}$$

where δ is a delay associated with the DFE structure. This delay is physically interpreted as the time necessary for the feedforward filter to produce a correct decision to be fed to the input of the feedback filter. Using the minimisation method presented in [9], where the transmitted symbol estimate $\hat{a}(n-\delta)$ has been assumed correct, i.e. equal to the transmitted symbol $a(n-\delta)$, we arrive at the following estimate of the feedback coefficients:

$$b_i = \sum_{j=1}^{N_f-1} c_j h_{\delta+i+j}, \quad \text{i=1,...,}N_b, \tag{1.38}$$

where h_i is the channel impulse response. This CIR can be obtained by the receiver upon sounding the channel.

There are numerous other ways of estimating the DFE coefficients, each one resulting in a new DFE design. The DFE structure can be used instead of linear equaliser schemes generally improving the achievable performance. However, since the feedback section uses the output of the decision device, the false decisions must be rare for the DFE to perform well. In other words, at high SNR values the DFE is bound to work significantly better than linear equalisers. For this reason, as we will see in Chapters 4 and 5, various robust feedback techniques have been devised, which invoke more reliable symbol decisions based on channel coding.

1.2.0.6 Epilogue

The equalisers we have presented so far have the common characteristic of using a training sequence, which is transmitted periodically and is also known at the receiver. This implies that the channel is periodically sounded. Clearly, as has already been mentioned, this sequence creates a transmission overhead by absorbing a certain fraction of the transmitted bits, which could have been used as information bits.

Following the above brief introduction of the most basic trained equalisers, we can now proceed to the most important part of this treatise, which is the extension of these equalisers to the blind scenario. In order to equalise the received signal without the use of a training sequence, one has to estimate the CIR purely on the basis of the received signal. More explicitly, the parameters of the equaliser have to be estimated jointly with the signal, since they are unknown. This can be achieved by using statistical estimation techniques. The feasibility of using blind equalisation in mobile communications applications is still an open research issue. In [16] an interesting discussion was presented on this topic.

In the next section, we investigate blind equalisation techniques and discuss various issues related to their design.

1.3 Overview of Blind Equalisers

1.3.1 Introduction

Blind channel equalisation has been attracting scientific interest since 1952, when Bussgang published his original study of a scheme now known as the "Bussgang algorithm" [17]. Following Bussgang's study, the topic of blind equalisation has continued to attract both academic as well as practical attention. In recent years researchers have intensified their efforts in this field due to the definition of a range of important video broadcast standards. The Pan–European Satellite–based Digital Video Broadcast (DVB–S) [18], the terrestrial DVB–T [19] and the cable–based DVB–C [20] systems constitute a family of harmonised systems, where both the DVB–S and the DVB–C schemes recommend blind equalisation. In this section, we offer a comprehensive overview of these techniques, focusing on two particular families of methods, namely on the so–called Bussgang algorithms and on the joint channel impulse response (CIR) and data sequence estimation. This section is structured as follows. In Subsection 1.3.2 we provide a brief historical perspective. The basic principles of blind equalisation are discussed in Section 1.3.3, followed by an overview of the so–called Bussgang equalisers in Section 1.3.5. In Section 1.3.6 we discuss algorithms which have the common characteristic of normalisation of the equalised signal power. Section 1.3.7 is dedicated to a discussion on convergence. In Section 1.3.9 joint data and channel estimation techniques are discussed. The second–order statistics based algorithms' principles are highlighted in Section 1.3.10. The basic principles of the fourth–order statistics based channel estimation and equalisation algorithms using the so–called "polycepstra" are summarised in Section 1.3.11. Complexity comparisons are provided in Section 1.3.12. Finally, in the next chapter the performance of these equalisers is studied comparatively. Let us now commence our discourse with a historical perspective.

1.3.2 Historical Background

In Figure 1.6, we give a historical perspective on the most important blind equalisation techniques in communications, as they appeared in the literature. Blind equalisation was originally contrived for equalising benign wireline based links, such as telephone lines or for stationary point to point microwave links, where the channel's characteristics varied slowly and hence there was no need for employing frequent and explicit channel sounding with the aid of a training sequence. In 1975 Sato [3] proposed

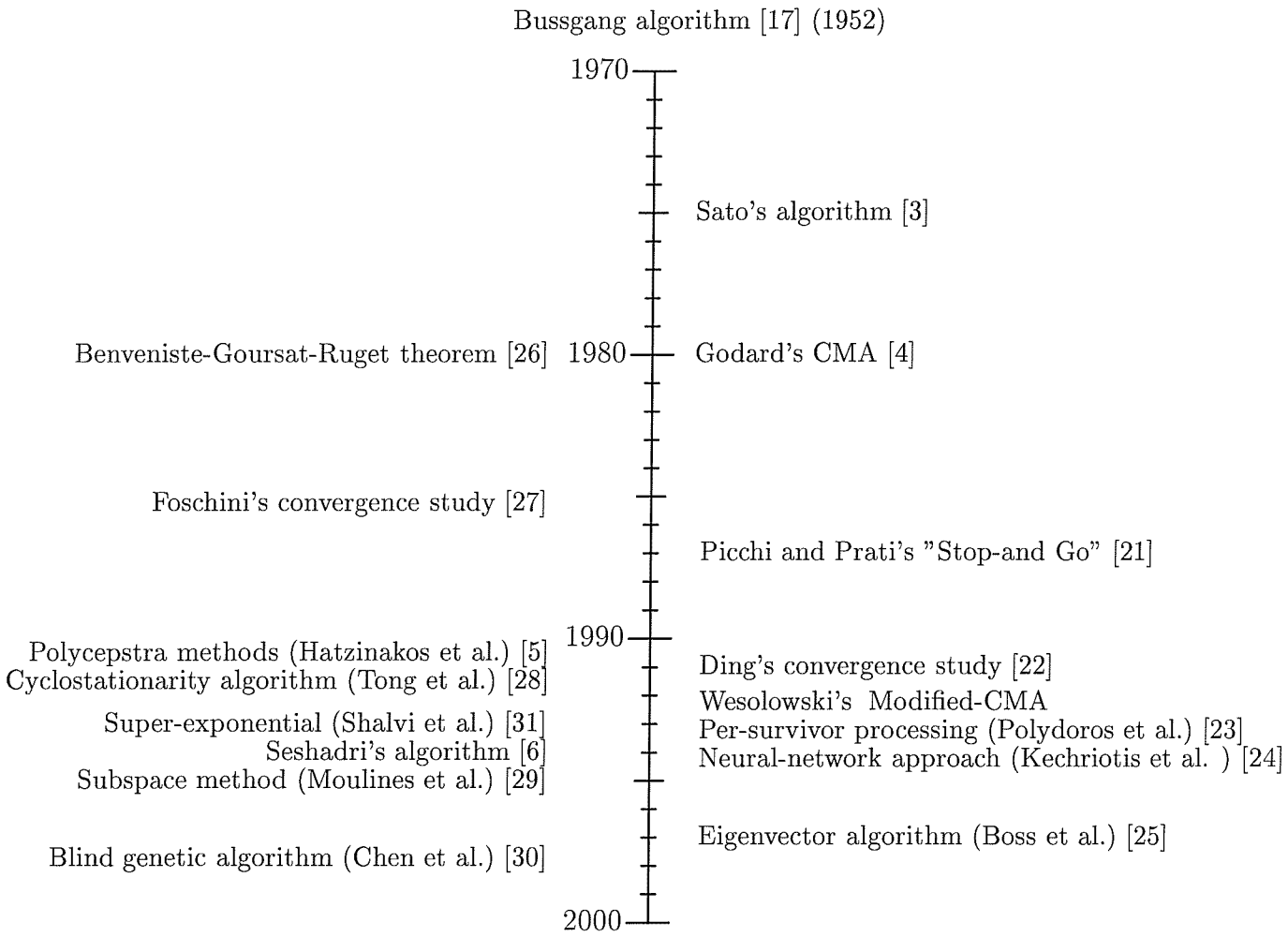


Figure 1.6: A brief history of blind equalisation

an algorithm for blind equalisation, which is now widely known as the "Sato's algorithm". Five years after Sato's publication, the blind equalisation problem was further studied for example by Benveniste, Goursat and Ruget in [26], where a sufficient condition for the convergence of a blind equaliser was established by the well–known Benveniste–Goursat–Ruget theorem as well as in [32], where a new algorithm was proposed. At the same time Godard [4] introduced a criterion, namely the so–called "constant modulus" (CM) criterion, leading to a new class of blind equalisers. Following Godard's contribution a range of studies were conducted employing the constant modulus criterion. Foschini [27] was the first researcher to study the convergence properties of Godard's equaliser assuming an infinite equaliser length. Later, Ding *et al.* continued this study [22] and provided an indepth analysis of the convergence issue in the context of a realistic equaliser. Although numerous researchers studied this issue, nevertheless, a general solution is yet to be found. A plethora of authors have studied Godard's equaliser, rendering it the most widely studied and applied blind equaliser. A well–known algorithm of the so–called Bussgang type [17, 33] was also proposed by Picchi and Prati [21]. Their "Stop–and–Go" algorithm constitutes a combination of the Decision–Directed algorithm [15] with Sato's algorithm [3]. After 1991, a range of different solutions to the blind equalisation problem were proposed. Seshadri [6] suggested the employment of the so–called M–algorithm, as a "substitute" for the Viterbi algorithm [34] for the blind scenario, combined with the so–called "least mean squares (LMS)" based CIR estimation. This CIR estimation was replaced by "recursive least squares (RLS)" estimation by Tzou, Raheli and Polydoros [23, 35], combining the associated channel decoding with the CIR estimation, leading to what was termed as "*Per–Survivor Processing*". Since then a number of papers have focused on this technique [23, 30, 36–47].

Prior to Seshadri's algorithm, Tong *et al.* [28] proposed a different approach to blind equalisation, which used oversampling in order to create a so–called "cyclostationary" received signal, and performed CIR estimation by measuring the autocorrelation function of this signal and by exploiting this signal's cyclostationarity. This technique was also applied to the case of 'sampling' the received signals of different antennas (instead of oversampling the signal of a single antenna) and further extended by Moulines *et al.* using a different method of CIR estimation, namely the so–called subspace method in [29]. Furthermore, Tsatsanis and Giannakis suggested that the cyclostationarity can be induced by the transmitter upon transmitting the signal more than once in each symbol interval [48]. A number of further contributions have also been published in the context of these techniques [49–64]. Finally, one of the first attempts of designing a blind equaliser, which would be more efficient in terms of both equalisation accuracy and convergence speed, than the family of Bussgang techniques at the cost of extra complexity was made by Hatzinakos and Nikias [5], who proposed a more sophisticated approach to blind equalisation by exploiting the so–called "tricepstrum" of the received signal. Until today, the blind equalisation problem is an open research topic, attracting significant amount of research. A

general answer to the fundamental question "*Under what circumstances is it preferable to use a blind equaliser to a trained–equaliser ?*" is yet to be provided.

Despite the scarcity of reviews on the topic, in the context of the Global System of Mobile Communications known as GSM an impressive effort was made by Boss, Kammeyer and Petermann [16], who also proposed two novel blind algorithms. We recommend furthermore the fractionally–spaced equalisation review of Endres *et al.* [65] and the Constant Modulus overview of Johnson *et al.* [66] based on a specific type of equalisers, namely on the so–called "fractionally–spaced" equalisers. A review of subspace–ML multichannel blind equalisers was provided by Tong and Perreau [67]. Further important references are the monograph by Haykin [15], the relevant section by Proakis [14] and the blind deconvolution book due to Nandi [68]. Comparative performance studies between various blind equalisers have also been performed. We recommend the second–order statistics–based comparative performance studies of Becchetti *et al.* [69], Kristensson *et al.* [70] and Altuna *et al.* [71], which is based on the mobile environment as well as the second–order statistics and PSP–based comparative study of Skowratanont and Chambers [72]. Furthermore, we recommend the fractionally–spaced Bussgang algorithm based comparative performance study by Shynk *et al.* [73], the CMA comparative performance study of Schirtzinger *et al.* [74] and the comparative convergence study by Endres *et al.* [75]. Let us now review the basic principles of blind equalisation in the next section.

1.3.3 Blind Equalisation Principles

1.3.4 Introduction

In this section we will introduce and discuss a range of blind equalisation principles, mainly focusing on the Bussgang techniques [17, 33]. Since Sato's original study in 1975 [3], blind equalisation has attracted significant scientific interest due to its potential in terms of:

- Overhead reduction. Training sequences sacrifice bandwidth in order to assist in determining the CIR, hence assisting equalisation. Blind equalisers do not need training sequences and therefore conserve bandwidth.
- Simplification of point to multipoint communications systems or broadcast. When a communications link is reset, equaliser adjustment "from scratch" is necessary. In this case, using a training sequence is inefficient, since the transmitter has to retransmit the training sequence specifically for each receiver, which is reset.

In Figure 1.7 the basic factors affecting the design of an equaliser are shown. In situations, where invoking trained equalisation is feasible, increased bandwidth efficiency is the main motivation behind

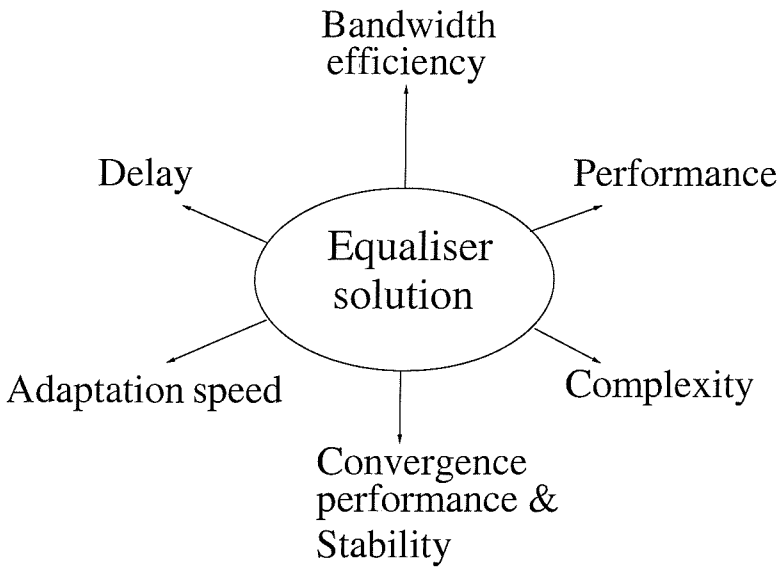


Figure 1.7: Contradictory design factors in the design of blind equalisers

using blind equalisation instead of trained equalisation, since the training sequence wastes a substantial fraction of the available bandwidth. The delay that an equaliser exhibits represents the amount of time that we can afford to wait, before the information is actually delivered from the equaliser to the other modules of the communication system. In some cases, for example when transmitting data files in a non–real–time fashion, delay is of no importance. However, the transmission integrity requirements of non–real–time data links are typically significantly higher than those of the delay–sensitive interactive speech or video links. More explicitly, in scenarios such as in a real–time interactive video telephone, the total system delay has to be confined to less than $100ms$ for maintaining 'lip–synchronisation'. The required performance of an equaliser is defined by the application for which it is designed. The average Bit Error Ratio (BER) as well as the burst Bit Error Ratio performance is a measure of the equaliser's ability to deliver correct hard–decision information. The Mean–Squared–Error (MSE) is another measure of performance, which quantifies the ability of an equaliser to deliver good soft–decision outputs. Both performance criteria are commonly measured and plotted against the Signal–to–Noise–Ratio (SNR). The implementational complexity of an equaliser is typically increased when the expected performance of the equaliser is higher. The task in this case is to exploit the equaliser's tolerable complexity in an efficient manner, so that the performance versus complexity ratio is maximised. The adaptation speed is the equaliser's ability to rapidly adapt the equaliser's coefficients to CIR changes. This characteristic is essential in violently fluctuating mobile communications channels but is of less significance for cable–based channels or any other channels, which can be modeled as quasi–stationary. Furthermore, blind equalisers typically require a substantial number of input symbols, before they can converge, so that they can circumvent the lack of training information. For this reason their adaptation speed is a crucial factor in the case of real–time links. Finally, the convergence of an

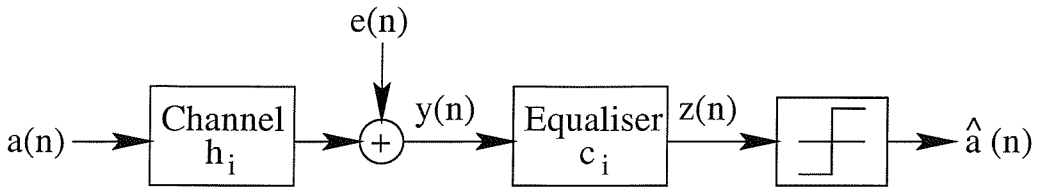


Figure 1.8: Equalised communications system

equaliser is not always guaranteed. Specifically, some blind equalisers have been shown to converge to spurious undesirable local minima, which can potentially lead them to deliver completely erroneous output symbols [22]. These situations have to be avoided when blind equalisers are used, provided that appropriate counter measures can be contrived at an acceptable complexity.

Here we curtail our discourse on the interpretation of Figure 1.7, although a range of further interesting design trade–offs can be gleaned from it. We note, however, that throughout this thesis we will refer to these design factors as measures of the equalisers' applicability to certain scenarios.

We will now proceed to describe the fundamental problem of blind equalisation in more depth. In our initial discussions the communication system parameters will be assumed to be time–invariant. This constraint is not necessary in general, although it is beneficial in terms of complexity reduction. Furthermore, the additive noise will be assumed to be Gaussian and white. Let us assume that the input bits, resulting from any encoder prior to modulation, are mapped into complex Quadrature Amplitude Modulation (QAM) [9] input symbols $a(n)$ transmitted at time instant n. These transmitted symbols are filtered by the CIR h_i and then the noise $e(n)$ is added to them, resulting in the received symbols $y(n)$ at time instant n in the form of:

$$y(n) = \sum_{i=-L_1}^{L_2} h_i \cdot a(n-i) + e(n), \tag{1.39}$$

where L_1 and L_2 are the length of the CIR's pre– and post–cursor sections surrounding the main tap (measured in terms of the number of transmitted symbols), respectively. An equaliser is typically placed after the channel in order to remove the channel–induced dispersion, as it is shown in Figure 1.8. Blind equalisation involves finding the 'best' equaliser filter, which regenerates the input symbols $a(n)$ at the receiver, without any knowledge of the CIR upon exploiting the knowledge of the distribution of the input QAM symbols. If the equaliser has N_1 feedback and N_2 feedforward taps $\{c_i\}$, then the equalised symbols will be of the form:

$$z(n) = \sum_{i=-N_1}^{N_2} c_i \cdot y(n-i) \tag{1.40}$$

and upon using Equation (1.39) we have:

$$z(n) = \sum_{i=-L_1-N_1}^{L_2+N_2} t_i \cdot a(n-i) + \sum_{i=-N_1}^{N_2} c_i \cdot e(n-i), \tag{1.41}$$

where $\{t_i\} = \{h_i\} * \{c_i\}$ is the convolution of the CIR with the equaliser filter, representing the total transfer function of the cascaded system constituted by the channel plus the equaliser. Assuming that the noise power is low compared to the power of the received signal, we can observe that the blind equalisation problem corresponds to estimating the suitable equaliser impulse response, which reduces the first term at the right–hand side of Equation (1.41) to only one of the summation terms. In this case, the cascaded system's impulse response tap–vector $\mathbf{t}$ takes the form of:

$$\mathbf{t} = (0, \cdots, 0, A, 0, \cdots, 0), \tag{1.42}$$

where A is a complex constant. This is the only case that corresponds to zero intersymbol interference (ISI). Assuming that the strongest signal path is located at time instant 0, Equation (1.41) can also be expressed as:

$$z(n) = t_0 \cdot a(n) + \sum_{i=-L_1-N_1,\ i\neq 0}^{L_2+N_2} t_i \cdot a(n-i) + \sum_{i=-N_1}^{N_2} c_i \cdot e(n-i). \tag{1.43}$$

The first term of Equation (1.43) is the useful one, including the one and only path at time 0. The second term is the ISI term, which is also referred to as *convolutional noise* [9]. This is because this term is a noise term, as far as the receiver is concerned and since it is the result of the convolution of the CIR h_i and the equaliser's impulse response c_i with the input signal $a(n)$. This is usually the main noise contributor at the initialisation of the equalisation process and it is reduced further during the stages of the equalisation process, leaving only the real noise term as the sole signal impairment, when the equalisation is perfect. Finally, the third term of Equation (1.43) is the noise term $e(n)$, convolved with the equaliser's impulse response c_i, since the noise has been filtered by the equaliser.

A problem similar to blind equalisation is the problem of blind deconvolution [76]. In blind deconvolution the aim is to perform joint channel and data estimation in a non–real time fashion, which is a fundamental difference with respect to the blind equalisation problem. The received signal is stored and then deconvolved in order to produce the CIR together with the input signal. This is the case, for example, when seismic signals are considered [77]. We have no knowledge of the signal emerging from the crust of the earth or of the channel that exists between the source of this signal and our receiver, hence we attempt to record the signal in order to deconvolve it later. When a channel estimation is available to us, we can estimate the input signal in two ways. A feasible approach is to use a sequence

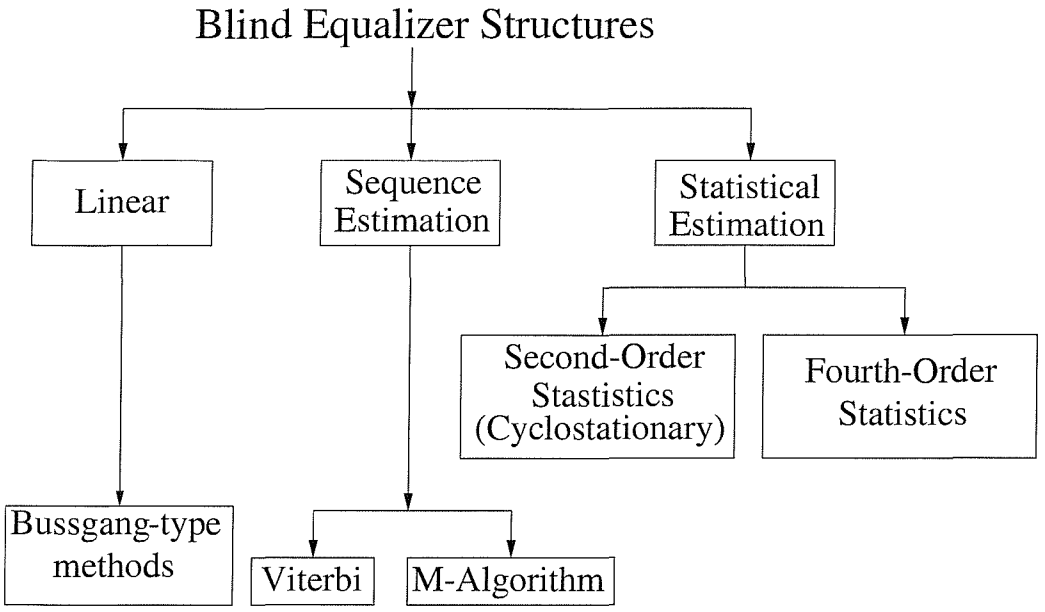


Figure 1.9: Classification of blind equalisers

estimation technique, such as the Viterbi algorithm [34]. Another approach is to produce an inverse filter of the channel and to perform filtering with the aid of this inverse filter. We will discuss this technique in more detail in Section 1.3.9.

Let us now point out an important aspect of channel equalisation, noting that if the distribution of the equaliser's input signal is Gaussian, then the equalised signal of Equation (1.41) is also Gaussian distributed [76]. This is true even if there is more than one term in the sum which are non–zero, i.e. even if ISI exists. Conversely, provided that the distribution of the equaliser's input signal is Gaussian, the equalised signal constituted by a sum of "independent identically distributed (i.i.d.)" Gaussian variables is also a Gaussian variable irrespective of whether there is residual ISI at its output. In this case, equalisation is impossible, since the equaliser cannot distinguish the zero–ISI equalised signal sequence from the other possible candidate sequences, which contain ISI. In other words, it is impossible to equalise the received signal, since the zero–ISI candidate received signal sequence is indistinguishable from the other ISI–contaminated candidate sequences. Hence blind equalisation cannot be performed, if the distribution of the equaliser's input signal is Gaussian. Fortunately, the uniform distribution of typical QAM data sources is far from the Gaussian, which renders the received signal amenable to blind equalisation.

Blind channel identifiability issues have been discussed by Benveniste, Goursat and Ruget in [26] and by Huang and Gustaffson in [78] among others.

The equalisers alluded to in this section are classified in Figure 1.9. Following the above general discussions on blind equalisibility, let us now proceed with the overview of the family of Bussgang

techniques in the next section.

1.3.5 Bussgang Blind Equalisers

In this section we set out to characterise a range of basic equaliser schemes involving the class of "*Bussgang*" techniques dating back to 1952 [17]. In our discussion we will consider the following signal vectors, which obey the notations of Figure 1.8:

$$\mathbf{a}(n) = [a(n+N_1+L_1),\ldots,a(n-N_2-L_2)]^T \tag{1.44}$$

$$\mathbf{y}(n) = [y(n+N_1),\ldots,y(n-N_2)]^T \tag{1.45}$$

$$\mathbf{c} = [c_{-N_1},\ldots,c_{N_2}]^T \tag{1.46}$$

$$\mathbf{t} = [t_{-N_1-L_1},\ldots,t_{N_2+L_2}]^T \tag{1.47}$$

$$\mathbf{e}(n) = [e(n+N_1),\ldots,e(n-N_2)]^T \tag{1.48}$$

$$\mathbf{H} = \begin{pmatrix} h_{-L_1} & \ldots & h_{L_2} & 0 & \ldots & 0 & 0 \\ 0 & h_{-L_1} & \ldots & h_{L_2} & 0 & \ldots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & h_{-L_1} & \ldots & h_{L_2} & 0 \\ 0 & 0 & \ldots & 0 & h_{-L_1} & \ldots & h_{L_2} \end{pmatrix} \tag{1.49}$$

where ^{T} denotes transpose, * denotes conjugate and ^{H} denotes the Hermitian matrix. From these definitions the following matrix relationships hold:

$$\mathbf{y}(n) = \mathbf{H}\cdot\mathbf{a}(n) \tag{1.50}$$

$$\mathbf{t} = \mathbf{H}^T\cdot\mathbf{c} \tag{1.51}$$

$$z(n) = \mathbf{c}^T\cdot\mathbf{y}(n) \tag{1.52}$$

$$z(n) = \mathbf{t}^T\cdot\mathbf{a}(n) + \mathbf{c}^T\cdot\mathbf{e}(n). \tag{1.53}$$

Explicitly, Equation (1.50) describes the received signal $y(n)$ as the convolution of the transmitted signal $a(n)$ and the CIR h_i. Equation (1.51) reflects the convolution of the CIR with the equaliser's impulse response and Equation (1.52) characterises the equaliser's output signal $z(n)$ as the convolution of the received signal $y(n)$ with the equaliser's impulse response c_i. Following these definitions, a general form of the Bussgang equaliser update Equations [17] can be expressed as:

$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} - \lambda\cdot\mathbf{y}^*(n)\cdot[z(n) - g\{z(n)\}], \tag{1.54}$$

where $g\{z(n)\}$ is a non–linear zero–memory function of the equalised output $z(n)$ and λ is the so–called "step–size" parameter, controlling the speed and the accuracy of the equaliser's convergence. The condition for attaining convergence in the mean value for these algorithms is [15]:

$$E[z(n) \cdot \mathbf{y}^*(n)] = E[g\{z(n)\} \cdot \mathbf{y}^*(n)] \tag{1.55}$$

or

$$E[z(n) \cdot y^*(n-i)] = E[g\{z(n)\} \cdot y^*(n-i)], \quad i = -N_1, \cdots, N_2. \tag{1.56}$$

Upon multiplying each side of this equation by the relevant equaliser tap coefficient c_i^* and summing the results for $i = -N_1, \cdots, N_2$ as in [15], we obtain:

$$E[z(n) \cdot z^*(n)] = E[g\{z(n)\} \cdot z^*(n)]. \tag{1.57}$$

If instead of multiplying by c_i^* we had multiplied by c_{i-k}^*, assuming that the equaliser has an infinite number of taps, then Equation (1.57) would take the form of:

$$E[z(n) \cdot z^*(n-k)] \simeq E[g\{z(n)\} \cdot z^*(n-k)], \tag{1.58}$$

which reflects the so–called *Bussgang property* that is satisfied by the Bussgang algorithms, when the equaliser length is doubly infinite. When both the feedforward and feedback equaliser lengths are sufficiently high, then the Bussgang property is approximately satisfied. In the strict sense, however, only Equation (1.57) is satisfied. When the equaliser has converged, then the equalised symbols $z(n)$ approximate the transmitted symbols $a(n)$ and Equation (1.57) becomes:

$$E[z(n) \cdot g\{z(n)\}] = E[|a(n)|^2] = 1, \tag{1.59}$$

provided that the input power is normalised.

The problem of finding the optimum Bussgang equaliser corresponds to finding the function $g\{z(n)\}$, which provides the best estimate of the corresponding transmitted symbol $a(n)$ for each equalised symbol $z(n)$, whilst satisfying Equation (1.59). According to the Maximum Likelihood (ML) criterion, this can be achieved by setting $g\{z(n)\} = E[a(n)|z(n)]$, i.e. setting $g\{z(n)\}$ equal to the expected (or most probable) value of the transmitted symbol, given the equalised symbol at time n, $z(n)$. Note here that we are investigating zero–memory solutions, that is only the value of the current equalised symbol is taken into account in the process and no previous values. In order to find this expected value, we have to estimate the distribution $f_z(z)$ of the equalised symbols $z(n)$. During the equaliser's initialisation, in general the equalised signal contains ISI. If during this initialisation phase we ignore

the channel noise by assuming that the ISI is the main signal impairment at this stage, then this means that the equalised signal consists of a number of transmitted signal replicas, each having a different delay and weight. When the number of these replicas is sufficiently high, according to the central limit theorem we can approximate the distribution of the equalised symbols $z(n)$ with a Gaussian distribution. In our analysis in this section we shall assume that the central limit theorem condition can be invoked since there is a sufficiently high number of ISI terms. In practical situations the presence of six to eight ISI terms is sufficient for the central limit theorem to become applicable. Assuming also M–level QAM transmissions, the expected value giving the estimate of $g\{z(n)\}$ obeys the following form, which is similar to the one given in [76] for pulse–amplitude modulation (PAM):

$$g_{ML}\{z(n)\} = E[a(n)|z(n)] = \frac{\sum_{i=1}^{M} A_i e^{-|z(n)-A_i|^2/2\sigma^2}}{\sum_{i=1}^{M} e^{-|z(n)-A_i|^2/2\sigma^2}}, \tag{1.60}$$

where the coefficients A_i constitute the signal amplitudes associated with the QAM constellation and σ^2 is the variance of the noise, consisting of two components, namely the convolutional noise and the Additive White Gaussian Noise (AWGN) induced by the channel. An estimate of the noise variance σ^2 must be available for the evaluation of Equation (1.60). However, the function $g\{z(n)\}$ is only the optimum one under the assumption of a Gaussian distribution for the composite noise, which is produced by the ISI plus the channel's additive noise. Depending on this distribution, different Bussgang algorithms exist. The well–known Godard (or CMA) [4], Sato [3], Benveniste–Goursat [26] or Stop–and–Go [21] algorithms constitute a few such algorithms. We will describe each of them in the forthcoming paragraphs and discuss their characteristics.

As an illustration, in Figure 1.10 we have plotted the real part of the function $g\{(z)\}$, evaluated from Equation (1.60), for the algorithm of $g_{ML}\{z(n)\} = E[a(n)|z(n)]$ under the assumption of $-20dB$ additive Gaussian noise power. The real part of a 16–QAM signal can take four discrete values, symmetrically distributed around the origin. The estimated signal, which approximates the most likely transmitted signal, should be close to these legitimate constellation points. We observe that this is the case, when the above ML algorithm is used, under the assumption of low noise ($20dB$ below the signal level). In this low–noise scenario, the ML algorithm estimates the transmitted signal as the constellation symbol, which is closest to the equalised symbol $z(n)$ at time instant n. This is illustrated in Figure 1.10 by the four levels, corresponding to the four discrete values that the real (or the imaginary) part of a 16–QAM signal can assume. If the noise variance is not sufficiently low, however, then the surface of Figure 1.10 loses its resemblance to the 16–QAM constellation. This is shown in Figure 1.11, where the noise variance was assumed to be $-10dB$, i.e. $10dB$ higher than in Figure 1.10. When no CIR information is available, it might be extremely optimistic to assume that, even without channel noise, the power of the ISI–induced noise would be as low as $-20dB$. In fact

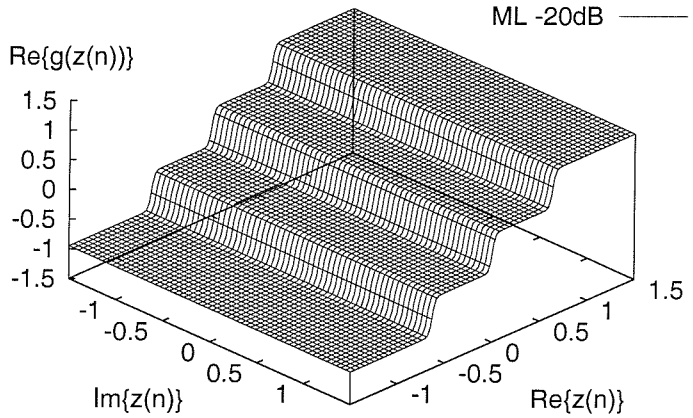


Figure 1.10: The real part of the function $g\{z(n)\}$ in Equation (1.60) plotted against the complex $z(n)$–plane for ML detection assuming Gaussian noise having $-20dB$ variance for 16–QAM.

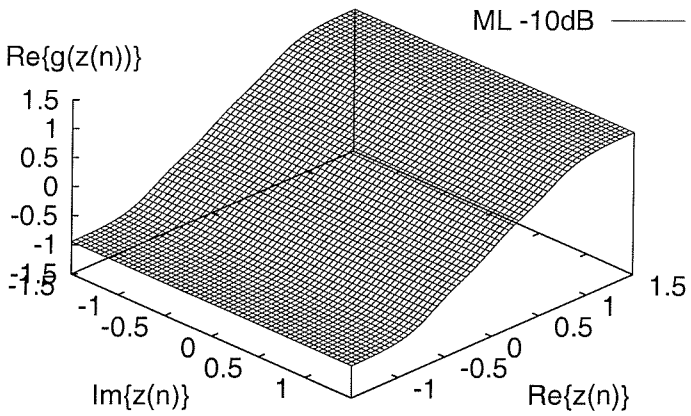


Figure 1.11: The real part of the function $g\{z(n)\}$ in Equation (1.60) plotted against the complex $z(n)$–plane for ML detection assuming Gaussian noise having $-10dB$ variance for 16–QAM.

it might be well over $0dB$, thus rendering the 16–QAM pattern unrecognisable in the received signal. The approximation of the estimation function $g\{z(n)\}$ for each Bussgang algorithm is given in a 2D plot in [76] for 8–level PAM, where the signals are real–valued. Similar figures can be generated by extending the approach of [76] to the case of 16–QAM, where the signals assume complex values, as can be seen in the 3D plots of Figures 1.10 and 1.11.

An alternative interpretation of the function $g\{z(n)\}$ can be observed by considering a "cost–function" $J\{z(n)\}$. The minimisation of this cost–function leads to the desired equaliser tap values, according to the wide–spread steepest descent algorithm [15]:

$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} - \lambda \cdot \frac{\partial J\{z(n)\}}{\partial \mathbf{c}}, \tag{1.61}$$

which physically implies that the taps $\mathbf{c}^{(n)}$ at instant n are modified by the derivative of the cost–function – after weighting by the step–size λ – in the direction of minimising the cost–function. According to Equations (1.61) and (1.54) the following relationship holds between $g\{z(n)\}$ and $J\{(z(n)\}$:

$$\frac{\partial J\{z(n)\}}{\partial \mathbf{c}} = \mathbf{y}^*(n) \cdot (z(n) - g\{z(n)\}. \tag{1.62}$$

Again, in simple, but conceptually feasible terms Equations (1.61) and (1.62) can be interpreted as updating each tap of the equaliser on the basis of the gradient of the error term

$$\epsilon(n) = z(n) - g\{z(n)\} \tag{1.63}$$

with respect to (wrt) a specific tap. Depending on the polarity of the cost–function's derivative wrt a specific tap, this tap is updated according to the step–size λ, such that in the next step it reduces $\epsilon(n)$ – hence the negative sign in Equation (1.61). Upon using the differentiation rules with respect to a vector given in Appendix C, we can readily arrive at:

$$g\{z(n)\} = z(n) - J'\{z^*(n)\}, \tag{1.64}$$

where $'$ denotes the derivative and the associated difference quantifies the discrepancy of the equalised output $z(n)$ and the derivative of the cost–function given by Equation (1.62). Equation (1.64) will be useful, when we consider the Bussgang cost–functions individually and derive the corresponding equaliser tap update algorithms.

Again, in all of our discussions, we employ QAM [9] and the general structure of the equaliser is shown in Figure 1.12. As we can see from this figure, the blind equaliser coefficients are updated using the knowledge of the received signal vector $\mathbf{y}(n)$, the equalised signal $z(n)$, the phase–corrected equalised

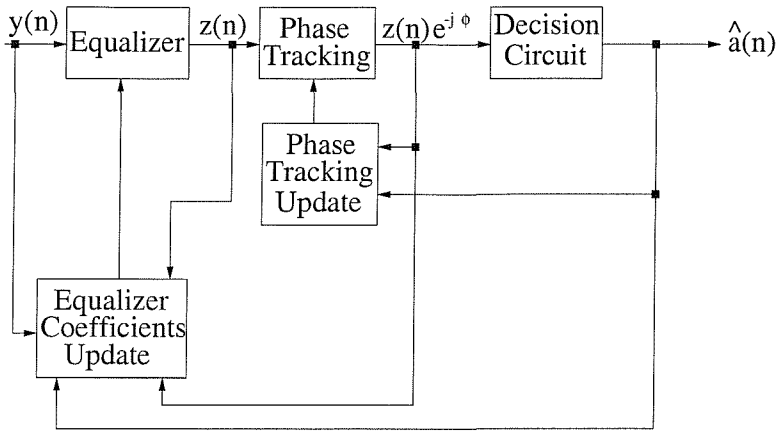


Figure 1.12: Equaliser structure used in Bussgang techniques

signal $z(n)e^{-j\phi}$ and the estimated signal $\hat{a}(n)$. We note, however, that a specific Bussgang equaliser may not make use of all of these signals in order to update its equaliser tap coefficients. It depends on the algorithmic implementation, which of these signals are invoked in the tap update process. What is common, however, to all Bussgang algorithms is that the error estimate in Equation (1.63) will be a function of the equalised symbol $z(n)$ at time n only, i.e. they are based on a zero–memory estimation.

In the equalisers presented in this section the sampling rate is identical to the signalling–rate – or Baud–rate, that is we use only one sample per symbol period. These equalisers are referred to as *symbol-spaced* schemes, as opposed to *fractionally–spaced equalisers*, which use more than one sample per symbol in order to equalise the channel. A typical example of fractionally–spaced equalisers is constituted by the family of second–order cyclostationary statistics based blind channel estimation algorithms [7]. In the context of Bussgang schemes, the extension of these equalisers to fractionally spaced arrangements is relatively straightforward. Such algorithms have been reported in the literature for example by Pei and Shih in [79] or by Dogancay and Kennedy in [80]. They have been further studied for example by Endres, Johnson and Green in [81], by LeBlanc, Fijalkow and Johnson in [82], by Endres, Halford, Johnson and Giannakis in [65], by Magarini *et al.* in [83] and by Papadias and Slock in [84].

Before proceeding to the discussion of Sato's algorithm, we note that the Bussgang zero–memory function $g\{z(n)\}$ of Equation (1.54) can be extended to the non–zero–memory case, if we take into consideration more than one equalised symbols in generating the error function of Equation (1.63). This was proposed by Yang for the CMA in [85] but more on this will be discussed in Chapter 3. Let us now consider a range of Bussgang algorithms in a little more depth in the forthcoming subsections.

1.3.5.1 Sato's Algorithm [3]

Sato's pioneering contribution in 1975 [3] described the first blind equalisation algorithm proposed, which was designed for real–valued signals and PAM. However, its extension to complex–valued signals and QAM is straightforward, especially in the spirit of Godard's publication of the well–known CMA [4], which was derived for complex–valued QAM signals. Sato's algorithm dedicated to real valued signals $z(n)$ uses the following cost–function [3]:

$$J^S(n) = E\left[(|z(n)| - \gamma)^2\right], \tag{1.65}$$

where γ is Sato's scaling coefficient and $E[]$, again, represents the expectation over all possible transmitted data sequences. It is clear that this cost–function is forcing the absolute value of the equalised signal to a fixed value γ. This is a plausible policy to pursue, when Binary Phase Shift Keying (BPSK) is used, but not for any other multilevel PAM scheme. For these multilevel constellations the minimisation of the Sato cost–function of Equation (1.65) may not seem to lead to the correct update of the equaliser taps at each iteration. Nonetheless, experimental experience shows that the minisation of Equation (1.65) may still lead to convergence to the desired zero–ISI equilibrium, although not in all cases, as we shall see in Section 1.3.7. The associated complex Sato cost–function can be defined as in [32]:

$$J^S(n) = E\left[(|Re\{z(n)\}| - \gamma)^2\right] + E\left[(|Im\{z(n)\}| - \gamma)^2\right]. \tag{1.66}$$

The steepest descent algorithm – which results from the cost–function of Equation (1.66) – can be found by determining the gradient of the cost–function with respect to the equaliser tap vector $\mathbf{c}$. Alternatively, using Equation (1.64) we obtain:

$$g_{Sato}\{z(n)\} = csgn(z(n)), \tag{1.67}$$

where the complex "signum" function is given by:

$$csgn(z(n)) = sgn(Re\{z(n)\}) + j \cdot sgn(Im\{z(n)\}). \tag{1.68}$$

In Figure 1.13 a plot similar to these in Figures 1.10 and 1.11 is given for Sato's algorithm. Clearly, this surface does not follow the four legitimate values of the 16–QAM constellation pattern of Figure 1.10, it is constituted into two planes. All the other Bussgang algorithms follow a pattern similar to that of Sato's algorithm and none of them follows the specific 16–QAM constellation pattern of the

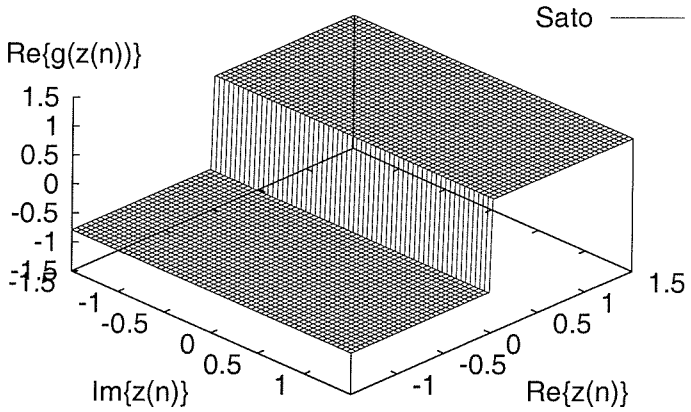


Figure 1.13: The real part of the function $g\{z(n)\}$ in Equation (1.67) plotted against the real part of the equalised symbol $z(n)$ for different Sato algorithm. The modulation used is 16–QAM.

ML algorithm seen in Figure 1.10. Sato's algorithm [3] is thus given, according to Equation (1.54) by:

$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} - \lambda \cdot \mathbf{y}^*(n) \cdot \epsilon^{Sato}(n), \tag{1.69}$$

where $\epsilon^{Sato}(n)$ is the Sato–error defined as:

$$\epsilon^{Sato}(n) = z(n) - \gamma \cdot csgn(z(n)). \tag{1.70}$$

Explicitly, the tap vector $\mathbf{c}$ in Equation (1.69) is adjusted according to the correction term $\lambda \cdot \mathbf{y}^*(n) \cdot \epsilon^{Sato}(n)$, where Sato's error term $\epsilon^{Sato}(n)$ depends on the cost–function of Equation (1.66). As it can be seen from Equation (1.70), this algorithm uses only the sign of the equalised output values $z(n)$ in order to update the equaliser coefficients. This implies that the exact value of $z(n)$ is ignored. Clearly, an error in the polarity of $z(n)$ is less probable, than an error in its exact value, when compared to the actual transmitted value $a(n)$. Therefore, Sato's algorithm has the advantage that it avoids using the generally error–prone exact value of $z(n)$, in favour of invoking the less spurious polarity of it. In the case of symmetric multilevel PAM transmissions, for which the algorithm was originally proposed, $Re\{z(n)\}$ assumes equi–probable positive and negative values. Using suitable coding and taking into account only the sign of $z(n)$ implies ignoring the fine–resolution channel effects. The same idea can be adopted for QAM transmissions, where $Re\{z(n)\}$ and $Im\{z(n)\}$ can be treated as two independent PAM constellations.

Setting the value of the scaling coefficient γ in Equation (1.69) is very important, since it actually directs the signal $z(n)$ to the point of its convergence, i.e. to the original constellation points. A way

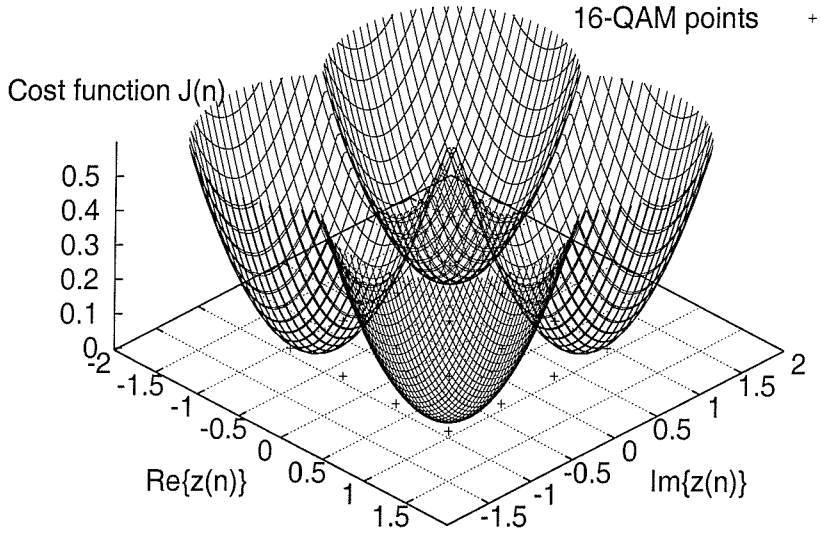


Figure 1.14: The cost–function of Sato's algorithm in Equation (1.66) in the context of square 16–QAM against the complex plane of the equalised signal $z(n)$

of achieving this is by constraining the mean value of the error update term of Equation (1.69) to zero. Therefore, the optimum value is set for γ in the minimum mean squared error sense. This optimum value was set by Sato [3] to:

$$\gamma = \frac{E[a(n)^2]}{E[|a(n)|]} \tag{1.71}$$

for real valued signals. This is the only value of γ, which sets the mean value of the error term in Equation (1.70) to zero. For complex–valued transmitted signals γ is given by a similar relationship:

$$\gamma = \frac{E[Re\{a(n)^2\}]}{E[Re\{|a(n)|\}]} = \frac{E[Im\{a(n)^2\}]}{E[Im\{|a(n)|\}]}. \tag{1.72}$$

Sato's cost–function Having presented Sato's algorithm, we will now take a closer look at the cost–function of this algorithm in order to explain some of the

algorithm's properties. In Figure 1.14 the cost–function is plotted against the complex plane of the equalised symbol $z(n)$ seen at the input of the decision device in Figure 1.2. When the equaliser reaches convergence, this equalised signal should cluster around the constellation points of the modulation scheme used, in this example 16–QAM. However, despite these expectations, we can clearly observe that the cost–function of Sato's algorithm exhibits minima only at the four points $(\pm\gamma, \pm\gamma j)$. More explicitly, the observed minima represent actual constellation points only for QPSK and BPSK. By contrast, for 16–QAM, the minima do not represent constellation points. This, of course, does not

mean that the algorithm does not converge for 16–QAM. What it implies is that in blind equalisation the "desired" state of equilibrium does not satisfy the condition that the equalised symbol is precisely equal to the transmitted symbol. As it has already been mentioned, this is because the receiver does not "know" what the transmitted symbol was, since no training sequence is used. We will see later that different Bussgang–type equalisers exhibit different types of minima in terms of their related cost–functions and this is reflected in their convergence properties. From Figure 1.14 we also observe that at the point $z(n) = 0$ the cost–function exhibits a local maximum. Local maxima are unstable points for these equalisers, since by using the negative gradient of their cost–function, we ensure that only local minima can be stable extrema. This is augmented in more depth in Appendix C.2. The above–mentioned characteristics do not prove the convergence of Sato's algorithm, but provide a deeper physical interpretation of these properties.

Having described Sato's algorithm, which was historically the first blind equaliser proposed, we will now present a modification of this algorithm.

Signed–Sato algorithm [86] A modified version of Sato's algorithm, referred to as the "*Signed–Sato*" technique was proposed by Weerackody, Kassam and Laker in [86]. According to this approach, the error signal is altered by taking only its sign into account, modifying Equation (1.69) as follows [86] :

$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} - \lambda \cdot \mathbf{y}^*(n) \cdot csgn(\epsilon^S(n)). \tag{1.73}$$

This algorithm performs the $csgn()$ function on the error term of Equation (1.70) rendering the update procedure insensitive to the error's actual value. One of the consequences of this is that when the equaliser is approaching convergence, the "almost correct" values of the equalised symbols are not exploited by the equaliser adaptation procedure. This implies that the convergence accuracy of this algorithm must be poorer than that of Sato's. On the other hand, faster convergence is achieved.

We observe the following properties of this algorithm, compared to Sato's algorithm:

- Its complexity is somewhat lower since the $csgn()$ function eliminates some multiplications, as we will see in Section 1.3.12.
- Its convergence has an increased dependence on the step–size parameter in the sense that a smaller λ is required for convergence, as compared to Sato's λ.
- Its convergence speed is higher.
- Its robustness appears to be better. This is because it only uses the sign of the error, which is less likely to be wrong than the noise–contaminated error itself.

1.3.5.1.1 Switching to Decision–Directed Equalisation After reaching the converged steady–state, maintaining the same step–size as during the convergence phase would result in poor equalisation performance. One should decrease the step–size, when convergence is accomplished, so that a better convergence accuracy is achieved. Ideally, one would want to switch to decision–directed equalisation, which could be combined with any of the equaliser coefficient computation techniques of Chapter 1.1, i.e. the LMS or RLS algorithms, for example. In order to accomplish this, we have to ensure that there are no decision errors at the receiver, otherwise a catastrophic performance degradation could be inflicted by the decision errors. If we could have this 'ideal side–information', then decision–directed adaptation would be applicable, which would act effectively as explicit equaliser training. However, the decision–directed equalisation works even in conjunction with a non–zero, but sufficiently low bit error rate. An early study on this issue has been carried out by Mazo in [87], where he identified the local minima of the decision–directed algorithm in the case of Pulse Amplitude Modulation (PAM) based transmission. Quite clearly, these minima depend on the modulation scheme used and on the number of equaliser taps, which in turn depend on the channel characteristics.

A more appropriate application of decision–directed equalisation is found in scenarios when explicit symbol–reliability information is available at the receiver, which can be generated for example with the aid of the channel decoder. In fact channel coding assisted schemes constitute the most promising improvements of blind equalisers designed for mobile channels. Simulations using Sato's algorithm during the initial phases of communication and switching to decison–directed mode after convergence are given in Section 2.5, using the conventional Sato algorithm as benchmarker. It is shown in Section 2.5 that employing the DD-enhanced algorithm provides a BER improvement, which is higher for higher–order QAM and for high SNR values, since for high SNRs the decision–directed errors are less frequent. For a deeper discussion on error–detection the reader is referred to [88–93]. Having described Sato's algorithm, let us now consider the so–called "*Constant Modulus Algorithm*", which was proposed by Godard in [4].

1.3.5.2 Constant Modulus Algorithm [4]

A more general algorithm was proposed by Godard for blind equalisation in [4] and also by Treichler in [94], which was further generalised later by Shalvi and Weinstein in [95]. Its related cost–function is defined as:

$$J^{(p,q)}(n) = \frac{1}{pq} E[||z(n)|^p - R_p|^q], \tag{1.74}$$

which will be elaborated on below. Godard's original algorithm and all other related algorithms use only $q = 2$ and this is the case that we will focus on from now on. One can observe that Sato's algorithm in Equation (1.66) is similar to the CMA of Equation (1.74), when we have $p = 1$ and $q = 2$. The reason for considering the difference betweeen the amplitude of $z(n)$ and a constant R_p instead of the actually transmitted symbol $a(n)$, is that in blind equalisation we attempt to match the equalised signal not to the actually transmitted sequence, which is never available, but to its statistics, which of course is known at the receiver. Given this cost–function, the coefficient adjustment algorithm using the steepest descent technique of Equation (1.61) can then be invoked. The value of the parameter R_p has to be matched to the constellation in a way similar to the setting of γ for Sato's algorithm in Equation (1.66). Clearly, this algorithm forces only the amplitude of the received signal to match a desired mean value, but ignores the phase. Consequently, the phase of the received signal might have arbitrary variations. Godard suggested that equalising only the magnitude of the signal should be adequate. The associated phase ambiguity can be removed by using differential phase encoding, which is congenial to the nature of blind equalisation. Indeed, a sign ambiguity is related to any blind equaliser, stemming from the fact that the QAM constellations are symmetric with respect to the x and y–axes. As it is clear from the definition of the CMA's cost–function in Equation (1.74), in the case of pure phase modulation, the equaliser's output $z(n)$ will be constrained to a constant value and the algorithm will readily converge [4]. A variation of this algorithm, which solves the problem of the arbitrary phase rotation is the so–called Modified–CMA, presented in the next section. Using Equations (1.64) and (1.74) we obtain:

$$g_{CMA}(z(n)) = z(n) \cdot \left[1 - |z(n)|^2 + R_2^2\right]. \tag{1.75}$$

Substituting this Equation into Equation (1.54) we readily arrive at the equaliser tap update equation of the CMA [4]:

$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} - \lambda \cdot \mathbf{y}^*(n) \cdot z(n) \cdot \left[|z(n)|^2 - R_2\right]. \tag{1.76}$$

The value of R_2 can be found by constraining the mean value of the update term of Equation (1.76) equal to zero, assuming that the equalised signal $z(n)$ is equal to the transmitted signal $a(n)$ with its phase rotated by a random value, i.e. assuming that the state of perfect equalisation has been reached. The procedure of determining R_2 is exactly the same as the one, which was used to compute Sato's scaling coefficient γ, yielding [4]:

$$R_2 = \frac{E[|a(n)|^4]}{E[|a(n)|^2]}. \tag{1.77}$$

In Figure 1.15, the CMA's cost–function is plotted against the complex plane of the equalised symbol

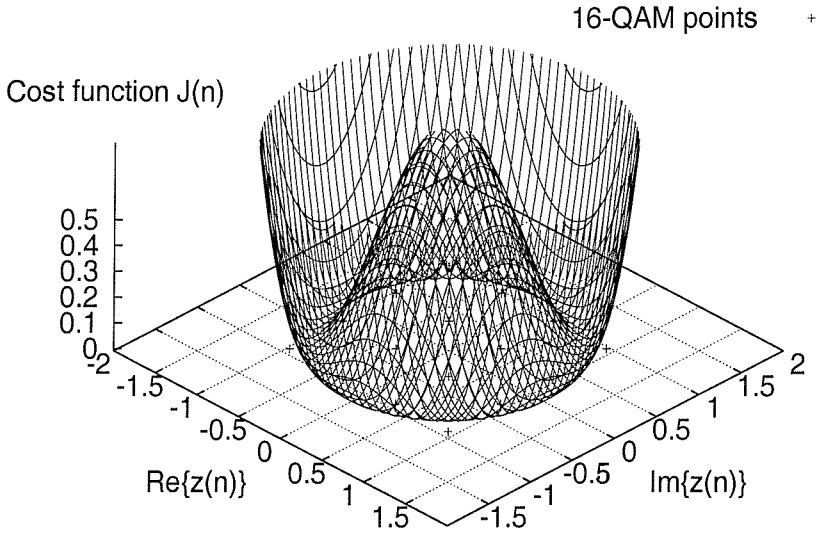


Figure 1.15: The cost–function of Equation (1.74) for the CMA in the context of square 16–QAM against the complex plane of the equalised signal $z(n)$

$z(n)$ at the decision device's input seen in Figure 1.2, in the same way as it was done for Sato's algorithm in Figure 1.14. It is plausible that the minima of this cost–function are on the circle of radius $\sqrt{R_p}$. As in the case of Sato's algorithm, the algorithm's adaptation procedure does not attempt to force the equalised symbol to be equal to the transmitted one, except in the case of pure phase modulation, such as for example for M–PSK. The difference between Sato's and Godard's algorithms is exactly this point. Specifically, while Sato's algorithm favours QPSK or BPSK, the CMA favours any pure phase modulated constellation, i.e. PSK, in which case, convergence becomes accurate. By using star QAM as in [9], we can force the equaliser to converge to a circle, or even to a pair of circles, as in the case of the twin–ring Star 16–QAM constellation. However, it is shown in [9] that this constellation is not optimum in the sense of noise resilience, since the Euclidean distance amongst its constellation points is lower than that of square 16–QAM [9]. One can readily visualise that by appropriately allocating points on the $z(n)$ plane, we can generate cost–functions, matching certain geometric patterns, each one giving rise to a different adaptation algorithm, which has its own properties. In this sense, an "optimum" equaliser would originate from a cost–function exhibiting local minima as close to the constellation points as possible. In the extreme case, when the local minima fall exactly on the constellation points, the cost–function has to be a polynomial of degree equal to at least twice the number of the constellation points, since only such a polynomial can have this number of local minima. This would imply using a high–order cost–function, and hence the resulting adaptation algorithm of which would be prone to instability. Small variations caused by additive noise can drive

the equaliser to instability. However, the same will happen when the equaliser's initialisation does not set it to its region of convergence, which is usually the case.

Having introduced Godard's classic CMA, let us now consider an important modification of it in the next section.

Signed–CMA A modification of this algorithm, similar to that of Sato's described in Section 1.3.5.1, is the so–called "*Signed–CMA*" proposed by Weerackody, Kassam and Laker in [96]. This algorithm uses the cost–function:

$$J(n) = E\left[|Re\{z(n)\}| + |Im\{z(n)\} - R_S|\right], \tag{1.78}$$

where R_S is a constant. The minimisation of this cost–function with respect to the equaliser's tap vector $\mathbf{c}$ is described in Appendix D.3, which results in the following update equation [96]:

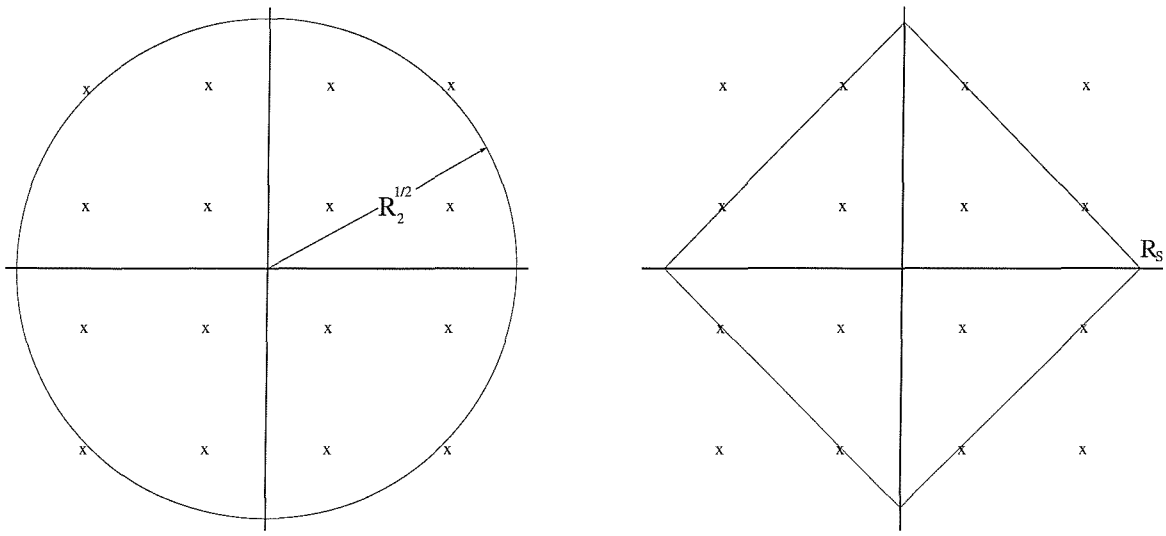
$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} - \lambda \cdot \mathbf{y}^*(n) \cdot sgn(|Re\{z(n)\}| + |Im\{z(n)\}| - R_S) \cdot csgn(z(n)) \tag{1.79}$$

where, again, $csgn()$ is defined as in Equation (1.68). It is clear from Equation (1.79) that the error signal expressed as $sgn(|Re\{z(n)\}| + |Im\{z(n)\}| - R_S) \cdot csgn(z(n))$ which multiplies $\mathbf{y}^*(n)$ in the update formula of Equation (1.79) results in quantised values of the form $\{\pm 1, \pm j\}$. This, in turn, results in forcing the equaliser to converge to the circumference of a 45^o rotated square in the signal space, instead of a circle, as Godard's algorithm would. As an illustration, in Figure 1.16 the corresponding convergence trajectories are drawn for the CMA and for the Signed–CMA algorithms in the case of square 16–QAM. In Figure 1.17, the error–surface of the Signed–CMA algorithm is drawn against the complex plane of the equalised signal, for square 16–QAM. As in the case of the Signed–Sato algorithm [86] of Section 1.3.5.1, this algorithm uses a decision based only upon the sign of the error and should be considered only in conjunction with a variable step–size parameter, in order to switch to a low step–size after convergence was attained.

A generalisation of Godard's algorithm was proposed by Shalvi and Weinstein in [95]. For this algorithm the cost–function to be minimised is the so–called Kurtosis of the equalised signal $z(n)$, defined as [5]:

$$K(z) = E[|z(n)|^4] - 2 \cdot E^2[|z(n)|^2] - |E[z^2(n)]|^2 \tag{1.80}$$

for complex–valued equalised symbols $z(n)$. The algorithm resulting from this cost–function is detailed in [95], where it is shown that this generalised algorithm results in Godard's algorithm as a special case. Following this algorithm, Shalvi and Weinstein proposed their so–called super–exponential algorithm [31], which uses 4–th order cumulants and converges at a nearly super–exponential speed. Cumulant–based blind equalisation algorithms have been proposed for example in [25, 97–101]. As



(a) (b)

Figure 1.16: Godard (a) versus Sign–Godard (b) convergence trajectories for 16–QAM according to Equations (1.74) and (1.78) respectively [96]

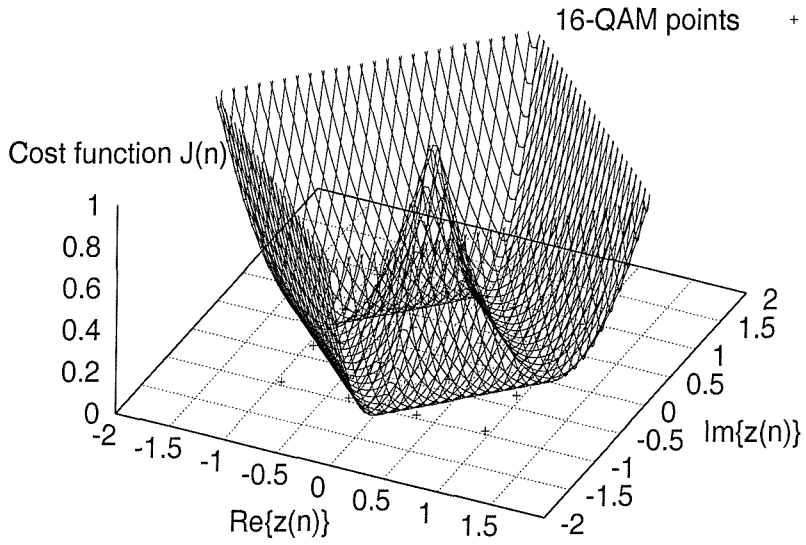


Figure 1.17: The cost–function of Equation (1.78) for the Signed–CMA algorithm in the context of square 16–QAM against the complex plane of the equalised signal $z(n)$

an important extension of Godard's algorithm, Wesolowsky's modified CMA is presented in the next section.

1.3.5.3 Modified Constant Modulus Algorithm [102]

Another modified version of Godard's CMA [4] was proposed by Wesolowsky in [102], employing a cost–function, which relies on both the real and imaginary parts of the equalised signal $z(n)$ [4]:

$$\begin{aligned} J(n) &= E[\Big(|Re\{z(n)\}|^2 - R_{2,R}\Big)^2 + \\ &\quad \Big(|Im\{z(n)\}|^2 - R_{2,I}\Big)^2]. \end{aligned} \tag{1.81}$$

The idea behind this cost–function, as compared to the CMA cost–function of Equation (1.74) is that both the real and imaginary parts of the signal are forced to a constant value and, therefore, the random phase ambiguity of the CMA now becomes only 90^o. This is meaningful in pure phase modulation, in which case the CMA may converge to an arbitrarily phase–shifted solution. For QAM though, the 90^o symmetry of the constellation makes it possible for both algorithms to converge to a 90^o phase–shifted solution. Following the same procedure as in the context of the other algorithms, based on Equations (1.61) and (1.62) we obtain:

$$\begin{aligned} \mathbf{c}^{(n+1)} &= \mathbf{c}^{(n)} - \\ &\quad -\lambda \cdot \mathbf{y}^*(n) \cdot [Re[z(n)] \cdot \Big((Re[z(n)])^2 - R_{2,R}\Big) \\ &\quad +j \cdot Im\{z(n)\} \cdot \Big((Im\{z(n)\})^2 - R_{2,I}\Big)] \end{aligned} \tag{1.82}$$

$$\begin{aligned} g_{MCMA}(z) &= z - (Re\{z\} \cdot \Big((Re\{z\})^2 - R_{2,R}\Big) + \\ &\quad j\Big(Im\{z\} \cdot \Big((Im\{z\})^2 - R_{2,I}\Big)\Big)). \end{aligned} \tag{1.83}$$

The values of $R_{2,R}$ and $R_{2,I}$ can be found using the same method as for the other algorithms, namely by constraining the mean value of the update terms in Equation (1.82), yielding [4]:

$$R_{2,R} = \frac{E\left[\left(Re\left\{a(n)\right\}\right)^4\right]}{E\left[\left(Re\left\{a(n)\right\}\right)^2\right]} \tag{1.84}$$

$$R_{2,I} = \frac{E\left[\left(Im\left\{a(n)\right\}\right)^4\right]}{E\left[\left(Im\left\{a(n)\right\}\right)^2\right]}. \tag{1.85}$$

As for all other Bussgang algorithms, in Figure 1.18 the error surface of Equation (1.81) is plotted against the complex plane of the equalised signal for 16–QAM. It is shown that while the CMA cost–function of Equation (1.74) has minima on a circle, the MCMA has its own minima at four points,

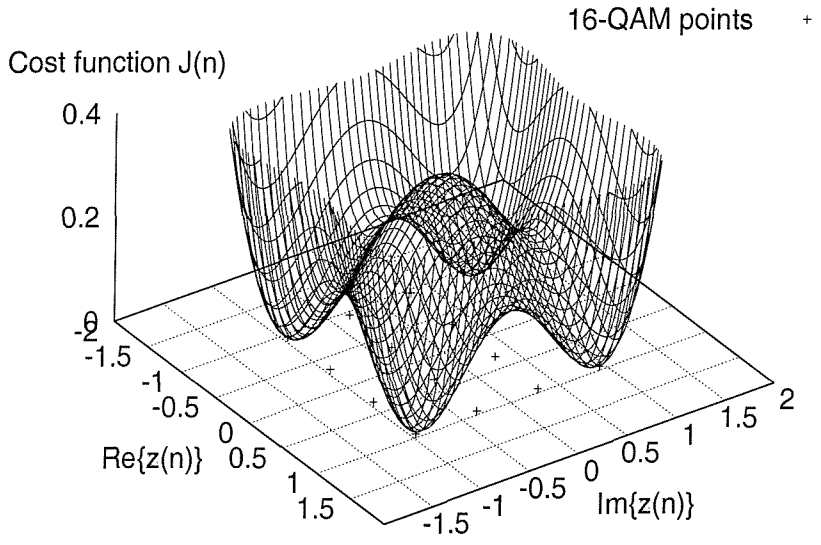


Figure 1.18: The cost–function of Equation (1.81) for the Modified–CMA algorithm in the context of square 16–QAM against the complex plane of the equalised signal $z(n)$

just like Sato's algorithm, while using fourth–order statistics instead of second–order statistics, as is the case for Sato's algorithm. It was shown by Wesolowsky [102] that the MCMA exhibits slightly faster convergence than the classical CMA, in particular for medium–distortion channels. It also offers slightly better steady–state performance, as will be shown in Chapter 2. Let us turn our attention to considering the *Benveniste–Goursat* algorithm [32].

1.3.5.4 Benveniste–Goursat Algorithm [32]

In Sato's algorithm [3], the error signal was expressed in Equation (1.70), which is repeated here for convenience:

$$\epsilon^{Sato}(n) = z(n) - \gamma \cdot csgn(z(n)).$$

The error term is used to update the equaliser coefficients, according to Equation (1.69). This error signal is however non–zero, when the signal is perfectly equalised, except in the case of QPSK, since γ is a constant value, reflecting the statistics of $a(n)$, while $z(n)$ generally takes its values from a multilevel constellation, when the SNR is sufficiently high and the equaliser has converged. This results in inaccurate steady–state behaviour along with small error fluctuations around the point of equilibrium associated with the minimum error. In other words, even near the optimum equaliser setting, not every tap update drives the equaliser towards the desired equilibrium. In order to remedy

these deficiencies, Benveniste and Goursat [32] considered the decision–directed error signal expressed as:

$$\epsilon^{DD}(n) = z(n) - \hat{a}(n), \tag{1.86}$$

which becomes zero, when equalisation has been accomplished, giving a good–steady state performance. On the other hand, this error signal cannot be employed during the equaliser's initial convergence phase, since at the beginning of the equalisation process the decisions concerning $z(n)$ are often erroneous and this would drive the coefficient update equation to an ill–conditioned state. Combining the error signals in Equation (1.70) and (1.86), each one scaled by a certain weight, Benveniste and Goursat [32] formulated a new error signal as:

$$\epsilon^{BG}(n) = k_1 \cdot \epsilon^{DD}(n) + k_2 \cdot |\epsilon^{DD}(n)| \cdot \epsilon^{Sato}(n), \tag{1.87}$$

where k_1 and k_2 are the corresponding weighting factors. This error signal is zero, when equalisation is perfect and, at the same time, it is not as error–prone as a purely blind decision–directed (DD) approach would be at start up, since then the influence of Sato's error term $\epsilon^{Sato}(n)$ in Equation (1.70) offers better error estimation. Using this combined error signal we readily arrive at the Benveniste–Goursat (B–G) algorithm [32], adjusting the equaliser taps according to:

$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} - \lambda \cdot \mathbf{y}^*(n) \cdot \epsilon^{BG}(n). \tag{1.88}$$

A good choice for k_1 and k_2 in Equation (1.87) would be to initialise the algorithm with a large k_2/k_1 ratio and decrease the ratio, when the equaliser is close to convergence in order to render the steady–state equalisation more accurate. This philosophy is similar to the idea of switching to decision–directed mode, when the equaliser has converged.

The related $g\{z(n)\}$ function in this case can be found by comparing Equations (1.54) and (1.88), yielding [76]:

$$g(z) = z(n) - k_1 \cdot \epsilon^{DD}(n) + k_2 \cdot |\epsilon^{DD}(n)| \cdot \epsilon^{Sato}(n). \tag{1.89}$$

Having discussed the Benveniste–Goursat algorithm, we will now consider another DD–like algorithm in the next section, namely the *stop–and–go* algorithm by Picchi and Prati [21].

1.3.5.5 Stop–and–Go Algorithm [21]

In the previous algorithms the equaliser coefficient update is inevitably occasionally wrong due to the statistical nature of the algorithms. This leads to a reduced convergence speed and also to a degradation of the steady–state performance of the equaliser. In order to avoid this impediment to

some degree, Picchi and Prati [21] suggested an algorithm, which decides whether a specific received symbol should contribute to the update process and updates the equaliser coefficients, only when it has decided that this would bring their values closer to their steady state ones. The algorithm used for updating the coefficients is the classic error feedback algorithm with the decision–directed error expressed as in Equation (1.86), which is repeated here for convenience:

$$\epsilon^{DD}(n) = z(n) - \hat{a}(n).$$

Two variables, namely $f_{n,R}$ and $f_{n,I}$ are introduced in [21], each of which defines a measure of the probability that the update of the real or imaginary part of the equaliser coefficients is correct. Naturally, the actual probability is unknown at the receiver, but it can be estimated using the philosophy of the Sato–type error of Equation (1.70), setting $f_{n,R}$ and $f_{n,I}$ to 1 and 0, depending on our confidence in the success of the update, as [21]:

$$f_{n,R} = \begin{cases} 1 & \text{if } sgn(Re[\epsilon^{DD}(n)]) = sgn(Re[\epsilon^{Sato}(n)]) \\ 0 & \text{if } sgn(Re[\epsilon^{DD}(n)] \neq sgn(Re[\epsilon^{Sato}(n)]) \end{cases} \tag{1.90}$$

and

$$f_{n,I} = \begin{cases} 1 & \text{if } sgn(Im\{\epsilon^{DD}(n)\}) = sgn(Im\{\epsilon^{Sato}(n)\}) \\ 0 & \text{if } sgn(Im\{\epsilon^{DD}(n)\}) \neq sgn(Im\{\epsilon^{Sato}(n)\}). \end{cases} \tag{1.91}$$

Practically this implies that the update of the real part of the equaliser coefficients only takes place when:

$$\left.\begin{array}{llll} & Re[z(n)] > \gamma & \text{and} & Re[\hat{a}(n)] < Re[z(n)] \\ \text{or} & 0 < Re[z(n)] < \gamma & \text{and} & Re[\hat{a}(n)] > Re[z(n)] \\ \text{or} & -\gamma < Re[z(n)] < 0 & \text{and} & Re[\hat{a}(n)] < Re[z(n)] \\ \text{or} & Re[z(n)] < -\gamma & \text{and} & Re[\hat{a}(n)] > Re[z(n)] \end{array}\right\}. \tag{1.92}$$

The graphical interpretation of this is given in Figure 1.19 where the "go" regions – i.e. the regions for which the equaliser decides to perform an update – are shown in the equalised complex symbol plane $z(n)$. From this we can infer that the "correct" values for $z(n)$ are those ones that are expected to bring $\hat{a}(n)$ closer to γ. This issue was richly illustrated in [21] in geometrical terms. This is a plausible, but certainly imperfect criterion. The probabilities of making a false update decision are also calculated in [21].

With the aid of these definitions we can form the algorithm using the classical error feedback algorithm of Equation (1.54), but involving both the real and imaginary parts of the error, enabled or disabled

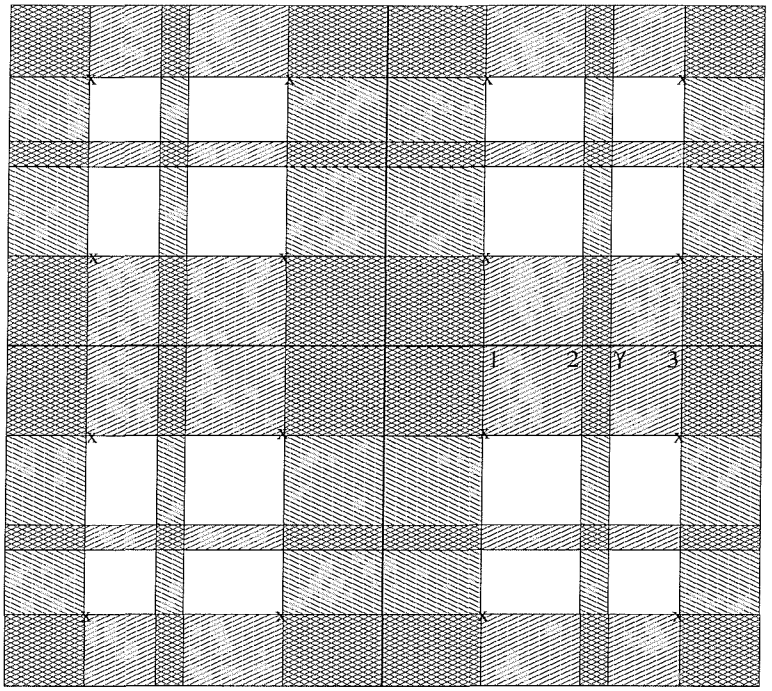


Figure 1.19: A graphical interpretation of the decision areas of the Stop–and–Go algorithm in the context of Equations (1.90) and (1.91) for 16–QAM. The 135^o–rotated lines correspond to the regions where the real part of the decision–directed error can be used for updating while the 45^o–rotated lines correspond to the regions, where the imaginary part of the error can be used for updating

by $f_{n,R}$ and $f_{n,I}$, as follows:

$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} - \lambda \cdot \mathbf{y}^*(n) \cdot \left[f_{n,R} Re\{\epsilon^{DD}(n)\} + j f_{n,I} Im\{\epsilon^{DD}(n)\} \right]. \tag{1.93}$$

A simple modification of this algorithm, which was suggested by Choi, Hwang and Song in [103], uses a CMA–type error term, instead of the Sato error term of Equation (1.70), in order to form the decisions concerning the validity of the equaliser update at each symbol. This error term is of the form:

$$e^{CMA} = z \cdot (|z(n)|^2 - R_p)^2, \tag{1.94}$$

where the symbols $z(n)$ and R_P are defined in Section 1.3.5.2. This error term corresponds to circular–type regions in the complex equalised symbol domain $z(n)$. The decision regions, in a similar fashion to the Stop–and–Go algorithm using the Sato error update of Figure 1.19, are portrayed in Figure 1.20. It is not intuitive, why an equalised symbol in these regions is more likely to update the coefficients correctly, than in the non–shadowed regions of Figures 1.19 and 1.20. However, we can say that when the equalised signal falls into these areas, then the decision–directed error term of Equation (1.86) becomes similar to the blindly detected Sato–type (or CMA–type) error term of Equation (1.94) and hence this signal can be used more confidently for updating the equaliser coefficients. This is because

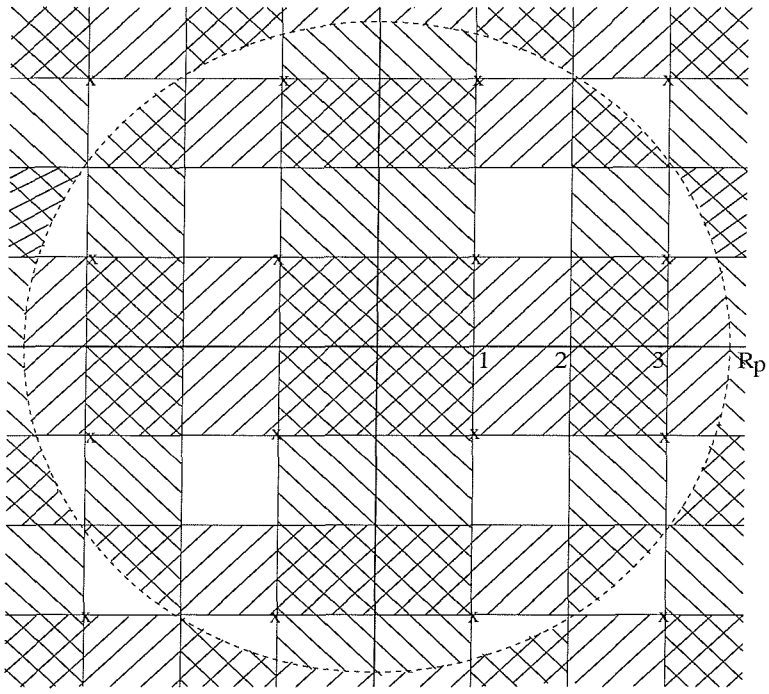


Figure 1.20: A graphical interpretation of the decision areas of the Modified–Stop–and–Go algorithm in the context of Equations (1.93) and (1.94) for 16–QAM. The 45^o–rotated lines correspond to the regions where the real part of the decision–directed error can be used for updating while the 135^o–rotated lines correspond to the regions, where the imaginary part of the error can be used for updating

this error term comes from a blind estimator and its mean value is always zero. In the opposite scenario, when the equalised signal $z(n)$ falls outside these areas, then the equaliser is simply not updated. Another observation is that any algorithm which is used to describe the blindly–estimated error, like Sato's algorithm or the CMA, would characterise the shape of the shadowed regions. For example, Sato's algorithm in Equation (1.69), which is based on the sign function, gives rectangular areas while the CMA of Equation (1.76), which is based on constant signal magnitude, gives circular regions. Close observation of Figures 1.19 and 1.20 reveals that the basic areas of equaliser updating in the two figures are similar; the difference is only in the shape of these regions.

This algorithm is expected to have an advantage over the previous algorithms of Sections 1.3.5.1–1.3.5.4, since it uses Equations (1.91)–(1.92) for rejecting unreliable coefficient updates and to render the convergence more steady and accurate. Nevertheless, the algorithm's convergence is hampered to a certain degree, since it does not use all the incoming symbols for updating the equaliser coefficients.

Finally, again, we present the symbol estimation function $g\{z(n)\}$ corresponding to this algorithm, which is readily found by observing Equations (1.54) and (1.93), yielding:

$$g\{z(n)\} = z(n) - \left[f_{n,R} \cdot Re\{\epsilon^{DD}(n)\} + jf_{n,I} \cdot Im\{\epsilon^{DD}(n)\}\right]. \tag{1.95}$$

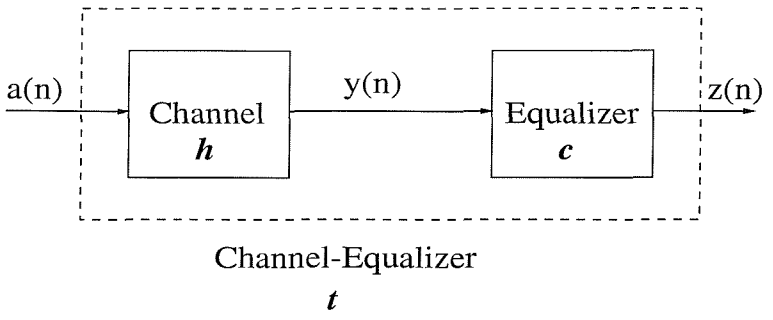


Figure 1.21: Combined channel plus equaliser system model considered by the super–exponential algorithm

Having reviewed the most important Bussgang algorithms, we now discuss the so-called "normalised algorithms" in the next section.

1.3.6 Normalised Algorithms

The philosophy of the normalised algorithms differs from that of the Bussgang techniques of Section 1.3.5 in that they use an equaliser update term, which is divided by the square of the L_2–norm of the received signal vector. This appears as a normalisation of $\mathbf{y}$, which explains the term "normalised".

1.3.6.1 The Super–Exponential Algorithm [31]

In this section a fast–converging blind equaliser is presented, which was proposed by Shalvi and Weinstein in [31]. The combined channel plus equaliser system–model is depicted in Figure 1.21. In the absence of ISI the combined channel plus equaliser scheme is expected to have a Kronecker–δ impulse response, which is ensured by this algorithm. Its design was the result of the authors' efforts to find a simple equaliser, which would converge only to the desired response and at a rapid rate. The algorithm is formulated by denoting the i–th entry of the combined channel plus equaliser scheme's impulse response vector, by t_i and by updating the i–th coefficient according to the following basic relationships:

$$t_i^{'} = (t_i)^p \cdot ((t_i)^*)^q \qquad p,q \geq 0,\ p+q \geq 2; \quad i = 0, \cdots, 2N+1 \tag{1.96}$$

$$t_i^{''} = \frac{t_i^{'}}{||\mathbf{t}^{'}||}, \tag{1.97}$$

where $\mathbf{t}^{'}$ is the combined vector of the channel plus equaliser scheme's impulse response and $||\mathbf{t}^{'}||$ is its L_2 norm given by:

$$||\mathbf{t}^{'}|| = \sqrt{\sum_{i=-N_1-L_1}^{N_2+L_2} |t_i'|^2}, \tag{1.98}$$

while $\mathbf{t}^*$ denotes the complex conjugate of $\mathbf{t}$. It is clear that Equations (1.96) force all but one of the coefficients to converge to zero, and the remaining one to a magnitude of unity. Alternatively, Equations (1.96) encourage the equaliser's impulse response taps to diverge towards instability, due to the powers to which the taps t_i are raised. Moreover, it forces the coefficients to converge rapidly, since they are raised to high powers. This boosts the speed of convergence for this algorithm, ultimately obeying a nearly exponential rate, hence its name. Furthermore, Equations (1.97) normalise the combined channel plus equaliser scheme's tap vector, ensuring that the vector $\mathbf{t}^{''}$ always has a unity norm, preventing it from diverging. Since a coefficient t_i may potentially converge to either a unity or a zero magnitude and since the vector $\mathbf{t}^{''}$ has a norm of unity according to (1.98), the only stable points of convergence for this system are the points for which one coefficient has a unity magnitude and the rest are zero. This solution is the one which corresponds to the condition of zero ISI and, consequently, it is the desired response. The only ambiguity that exists for these points of convergence is concerning which of the coefficients has the unity magnitude, and the associated phase of this coefficient. These ambiguities are present in all blind equalisers and they are inherent in their nature. Having described the algorithm in terms of the combined channel plus equaliser impulse response, Shalvi and Weinstein proposed an equaliser adaptation procedure corresponding to the above equations [31]. This adaptation procedure is based on higher–order statistics (cumulants) and the order of the statistics required depends on the values of p and q in Equations (1.96). The algorithm is described by the following equations [31]:

$$\mathbf{c}^{'} = \mathbf{R}^{-1} \cdot \mathbf{d}, \tag{1.99}$$

$$\mathbf{c}^{''} = \frac{\mathbf{c}^{'}}{\sqrt{\mathbf{c}^{'} + \mathbf{R} \cdot \mathbf{c}^{'}}}, \tag{1.100}$$

where

$$[\mathbf{R}]_{ij} = \frac{E\left[y(n-j); y^*(n-i)\right]}{E\left[|a(n)|^2\right]}, \tag{1.101}$$

$$[\mathbf{d}]_i = \frac{L\left(z(n) : p; z^*(n) : q; y^*(n-i)\right)}{L\left(a(n) : p; a^*(n) : q+1\right)}, \tag{1.102}$$

and $y(n)$ is the received signal, $a(n)$ is the transmitted signal, $z(n)$ is the equalised signal at time instant n and $\mathbf{c}^{'}$, $\mathbf{c}^{''}$ are the equaliser vector and the normalised equaliser vector respectively. Furthermore, the definition and properties of the cumulants (functions of the form $L()$) are given in Appendix F. It is clear that this algorithm has an inceased complexity because of the matrix inversion

involved in Equation (1.99) and due to the cumulants that have to be evaluated in Equation (1.102) for performing the updates. A method for estimating the cumulants is given in [31] for $p = 2$ and $q = 1$. For large p or q values, the complexity is significantly increased. On the other hand, global convergence is guaranteed and at a nearly super–exponential speed. A first–order approximation of the super–exponential algorithm was also provided in [31], which led to the CMA. Additionally, it has been shown by Gu and Tong [104] that under certain circumstances the convergence characteristics of the super–exponential algorithm are similar to those of the CMA, since both algorithms exhibit only length–dependent undesirable local minima. Several further studies have been conducted in the context of this algorithm since it was proposed [105]. A range of additional modifications have also been proposed in [106], while in [107] its extension to the RLS is proposed. In [108] the algorithm's extension to the fractionally–spaced case is considered and finally in [109] a reduced–complexity modification is suggested.

1.3.6.2 Normalised CMA [110]

This algorithm, proposed by Papadias and Slock in [110], uses a different approach from the classical CMA. In mathematical terms the cost–function to be minimised is formulated as:

$$min_{\mathbf{c}^{(n+1)}}||\mathbf{c}^{(n+1)} - \mathbf{c}^{(n)}||_2^2 \tag{1.103}$$

where $||\mathbf{x}||_2^2$ is the L_2–norm of the vector $\mathbf{x}$, which is defined as:

$$||\mathbf{x}||_2^2 = \sum_{i=1}^{N} |x_i|^2 \tag{1.104}$$

under the constraint that the estimate $\check{z}(n)$ of the equalised symbol $z(n)$ is constrained to take a value from the set of 'legitimate' symbols of the constellation $d(n)$. More explicitly, the constraint is that

$$\check{z}(n) = d(n) \tag{1.105}$$

where

$$\check{z}(n) = \mathbf{y}^T(n) \cdot \mathbf{c}^{(n+1)} \tag{1.106}$$

is the dot–product of the received vector $\mathbf{y}(n)$ and the equaliser tap vector $\mathbf{c}^{(n+1)}$, physically giving an estimate $\check{z}(n)$ of the equalised symbol of $z(n)$, by filtering the received signal $y(n)$ using the updated equaliser taps $\mathbf{c}^{(n+1)}$, rather than the old ones, $\mathbf{c}^{(n)}$. The cost–function of Equation (1.103) can be physically interpreted as an equalisation technique which converges when the equaliser taps do not

significantly change from the iteration n to $n+1$. To elaborate a little further, the quantity $d(n)$ in Equation (1.105) can be either the actually estimated symbol of the QAM constellation in the context of the decision–directed approach yielding $d(n) = \hat{a}(n)$, or a point on the circle having a radius of $\sqrt{R_p}$ and centred at zero in Figure 1.16(a) in the context of the CMA approach. Alternatively $d(n)$ may assume one of the points $\{\pm\gamma, \pm j\gamma\}$ in Figure 1.16 for Sato's approach in Section 1.3.5.1, at time n. The constraint of Equation (1.105) is often referred to as a 'hard constraint', which is difficult to satisfy. A modification of this constraint is the so–called 'soft constraint', which is easier to implement [110]:

$$min_{\mathbf{c}^{(n+1)}} ||\mathbf{y}(n)||_2^2 \cdot ||\mathbf{c}^{(n+1)} - \mathbf{c}^{(n)}||_2^2 + \mu \cdot |\check{z}(n) - d(n)|^2. \tag{1.107}$$

At this stage we assume that $d(n)$ is known, however, during our further discourse in this section we will give the best choices for it [110]. The original minimisation problem of Equation (1.103) can be viewed as a special case of the generalised constraint of Equation (1.107). In fact, this soft constraint represents a weighted combination of the cost–function minimisation in Equation (1.103) and the previous hard constraint. The minimisation of Equation (1.107) leads to the following update equations [110]:

$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} - \lambda \cdot \frac{\mathbf{y}^*(n)}{||\mathbf{y}(n)||_2^2} \cdot (z(n) - d(n)) \tag{1.108}$$

where $\lambda = \mu/(1+\mu)$ and

$$d(n) = \begin{cases} R_p \cdot sign(z(n)) & \text{for CMA} \\ \hat{a}(n) & \text{for DD,} \end{cases} \tag{1.109}$$

and where $sign(re^{j\phi}) = e^{j\phi}$ removes the magnitude r and retains the phase information $e^{j\phi}$.
We can observe the basic difference in comparison to the Bussgang techniques of Section 1.3.5, which is the normalisation of the received vector $\mathbf{y}(n)$ by its L_2 norm in Equation (1.108). The philosophy of this equaliser is that it reaches convergence, when its coefficients change only slightly at each iteration, that is if $\mathbf{c}^{(n+1)} \approx \mathbf{c}^{(n)}$ and also Equation (1.105) is satisfied, which means that the estimated equalised symbols belong to the desired set. When DD equalisation is employed, Equation (1.105) implies that the estimation $\check{z}(n)$ of $z(n)$ is perfect, provided that the associated decision was error–free. In the case of the CMA, the equalisation is imperfect, if the modulation is not purely phase modulation, i.e. the constellation points are not on a circle, as in Figure 1.16(a). This implies that the ISI does not become zero, it rather oscillates around zero. This point was made earlier, in the context of our discussion on the CMA. The Normalised CM algorithm (*NCMA*) considered here can also be readily extended to the RLS technique, which guarantees fast adaptation in exchange for its higher complexity. In order to achieve this, one only has to replace the DD error of the constraint in Equation (1.105) with the weighted sum of errors, as in Equation (1.22) and proceed as in Section 1.2.0.4, based on the

calculations of Appendix B. This algorithm, is referred to as the *affine projection algorithm*, which was proposed by Ozeki and Umeda [111].

1.3.6.3 Soft Constraint Algorithm [112]

The Soft Constraint Satisfaction (SCS) algorithm, which was contrived by Constantinides *et al.* in [112], relies on an approach similar to that of the NCMA. The criterion that has to be satisified is the same as in Equation (1.103), but the constraint of Equation (1.105) is different. In mathematical terms the constraint requires the magnitude of the equalised samples to be constant, which is formulated as:

$$|\check{z}(n)|^2 = R^2, \tag{1.110}$$

where again, the equalised samples are given by:

$$\check{z}(n) = \mathbf{y}^T(n) \cdot \mathbf{c}^{(n+1)}. \tag{1.111}$$

In order to find a practical solution, again, the constraint of Equation (1.110) is relaxed so that it becomes incorporated in the minimisation process and we attempt to find an algorithm, which minimises the following cost–function:

$$min_{\mathbf{c}^{(n+1)}} ||\mathbf{c}^{(n+1)} - \mathbf{c}^{(n)}||_2^2 + \mu \cdot (|\check{z}(n)|^2 - R^2). \tag{1.112}$$

The resulting algorithm is [112]:

$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} - \frac{\lambda}{||\mathbf{y}(n)||_2^2} \cdot \mathbf{y}^*(n) \cdot (z(n) - \check{z}(n)), \tag{1.113}$$

where a step–size parameter λ was introduced and $\check{z}(n)$ is estimated as

$$\check{z}(n) = \frac{z(n)}{1 - \lambda \cdot (1 - |z(n)|/R)}, \tag{1.114}$$

while R is a constant to be determined on the basis of [112] by setting the mean value of the error term in Equation (1.113) to zero, when the system is close to convergence. In this case $\lambda \approx 1$, $\check{z}(n) \approx R \cdot sign(z(n))$ and R is determined as [112]:

$$R = \frac{E\left[|a(n)|^3\right]}{E\left[|a(n)|^2\right]}. \tag{1.115}$$

Using $\lambda \neq 1$ the algorithm does not satisfy Equation (1.112) exactly, only approximately. By comparing Equations (1.108) and (1.113) as well as setting $\lambda = 1$, the algorithm becomes the same as the

NCMA.

It is shown in [112] that the Soft Constraint algorithm has better convergence properties, than the CMA, as well as the Sato and the Normalised CMA algorithms in terms of convergence probability. A generalised version of the algorithm is given in [112] using not one, but multiple constraints. This algorithm involves matrix inversions and it is rather demanding in terms of computational complexity. Having reviewed a range of Bussgang algorithms, let us now consider some of the associated convergence issues in the next section.

1.3.7 Convergence Issues

1.3.7.1 State of the Art

In this section, the convergence properties of the Bussgang equalisers are discussed and the problems associated with them are explored. These issues attracted the attention of researchers as early as 1980, when Benveniste, Goursat and Ruget proved the convergence of a class of blind equalisers, under an assumption for the distribution of the transmitted signal and the assumption of an infinitely parametrised equaliser, Godard also studied the convergence of the algorithm he proposed [4]. Later, in 1985, Foschini [27] provided a proof of the convergence of the CMA, when the length of the equaliser is doubly infinite, i.e. when both the number of the feedforward and the feedback taps of the equaliser was infinite. It was not until a few years later that Ding *et al.* [22] proved that when the equaliser is not of infinite length, then there can be undesirable stable local minima, depending on the CIR. Ding *et al.* arrived at this conclusion by considering a special class of channels, namely the so–called "autoregressive" channels, and by finding the local minima of a CMA equaliser for these channels. Their theory presented in [113] also revealed that the local minima of the CMA based Baud–rate spaced Bussgang equaliser correspond to local minima of all other Baud–rate spaced Bussgang equalisers for the same channel, arising from the fact that the equalisers do not have an infinite length. These minima are thus referred to as length–dependent minima [114]. Algorithm–dependent minima do not exist in the family of CMAs, but do exist in Sato's algorithm [115], in the context of the Benveniste–Goursat algorithm [32] and in conjunction with the Stop–and–Go algorithm [114]. The above two CMAs and also the Shalvi–Weinstein algorithms [31] exhibit only length–dependent local minima. It has to be mentioned that a general solution for the convergence of the Bussgang equalisers is still an open research issue.

The regions around undesirable local minima have also been studied by Ding *et al.* [116, 117] and by Johnson *et al* [118], while initialisation strategies have been proposed by Li and Ding in [119]. Moreover, it has been indicated that the convergence performances of the CMA and the Shalvi–

Weinstein algorithms are similar to each other and also similar to the performance of the LMS (or Wiener) receiver [104, 120, 121]. Finally, dynamic convergence issues have been treated, for example, in [122–124]. Similar studies have recently been conducted also for fractionally–spaced equalisers [59, 81, 125, 126].

1.3.7.2 Convergence in the Absence of Channel Noise

In this overview we give a basic analysis model for the convergence of Bussgang equalisers and interpret some well–established results. We commence this analysis with a convergence analysis of the CMA. We consider a noiseless environment, which simplifies our discussion. We recall the error term of the CMA's equaliser tap update formula from Equation (1.76):

$$\epsilon^{\mathbf{CMA}}(n) = \mathbf{y}^*(n) \cdot z(n) \cdot (|z(n)|^2 - R_2). \tag{1.116}$$

This error term is basically the derivative of the CMA's cost–function in Equation (1.74) with respect to the equaliser tap vector. The points at which the mean value of this error term becomes zero define the local minima, maxima and saddle points of this algorithm. Therefore, in order to find the possible local minima, we have to evaluate the local minima of the following equation:

$$\mathbf{y}^*(n) \cdot z(n) \cdot (|z(n)|^2 - R_2) = \mathbf{0}. \tag{1.117}$$

By substituting the vectors from Equation (1.44) to (1.49) into Equation (1.117) and taking the expectation we have:

$$\begin{aligned} & E\left[\mathbf{y}^*(n) \cdot z(n) \cdot (|z(n)|^2 - R_2)\right] \\ &= E\left[\mathbf{H}^* \cdot \mathbf{a}^*(n) \cdot (\mathbf{a}^T(n) \cdot \mathbf{t}) \cdot (z^*(n)z(n) - R_2)\right] \\ &= E\left[\mathbf{H}^* \cdot (\mathbf{a}^*(n) \cdot \mathbf{a}^T(n)) \cdot \mathbf{t} \cdot \left((\mathbf{t}^H\mathbf{a}^*(n)) \cdot (\mathbf{a}^T(n)\mathbf{t}) - R_2\right)\right] \\ &= E\left[\mathbf{H}^* \cdot \left((\mathbf{a}^* \cdot \mathbf{a}^T)(\mathbf{t} \cdot \mathbf{t}^H)(\mathbf{a}^* \cdot \mathbf{a}^T) - R_2 \cdot (\mathbf{a}^* \cdot \mathbf{a}^T)\right) \cdot \mathbf{t}\right] \\ &= \mathbf{H}^* \cdot \mathbf{T} \cdot \mathbf{t}, \end{aligned} \tag{1.118}$$

where

$$\begin{aligned} \mathbf{T} \;=\; & \mu_4 diag(|t_i|^2 - 1) + \\ & \mu_2^2 \cdot (\mathbf{t}\mathbf{t}^H + \mathbf{t}^T\mathbf{t}^*\mathbf{I} - \\ & -2 \cdot diag(|t_{-K_1}|^2, \cdots, |t_{K_2}|^2)), \end{aligned} \tag{1.119}$$

or

$$[\mathbf{T}]_{i,j} = \begin{cases} \mu_4 \cdot (|t_i|^2 - 1) + \mu_2^2 \sum_{k=-K_1, k\neq i}^{K_2} |t_k|^2 & i = j \\ \mu_2^2 t_i \cdot t_j^* & i \neq j \end{cases} \quad (1.120)$$

and $\mu_j = E[|a(n)|^j]$, while $K_1 = N_1 + L_1$, $K_2 = N_2 + L_2$. The candidate stationary points will satisfy the set of equations:

$$\mathbf{H}^* \cdot \mathbf{T} \cdot \mathbf{t} = \mathbf{0}. \quad (1.121)$$

The resulting equations may have two types of solutions. We can assume that the null–space of matrix $\mathbf{H}^*$ is trivial, i.e. we assume that:

$$\mathbf{H}^* \cdot \mathbf{x} = \mathbf{0} \Rightarrow \mathbf{x} = 0. \quad (1.122)$$

As long as the number of the taps is finite, the channel matrix $\mathbf{H}^*$ is an $(N_1+N_2+1)\times(N_1+N_2+L_1+L_2)$ dimensional matrix, which has less rows than columns. A system characterised by $\mathbf{H}^* \cdot \mathbf{x} = 0$ in conjunction with such a matrix $\mathbf{H}^*$ always has an infinite number of non–trivial solutions. If the equaliser has an infinite number of feedforward and feedback taps though, then the situation changes. We can see that the channel matrix $\mathbf{H}$ of Equation (1.49) now has an infinite number of rows and columns, thus being a square matrix having linearly independent rows, which implies that the system characterised by Equation (1.122) has only the trivial solution. This was shown differently by Ding *et al.* in [113]. For the moment, we will assume that the matrix has a trivial nullspace. In this case we can find the stationary equilibrium points of the algorithm by finding the solution of the following set of equations:

$$\mathbf{T} \cdot \mathbf{t} = \mathbf{0}. \quad (1.123)$$

The solution of these equations can be found to be any vector $\mathbf{t}$, which has some zero entries and some non–zero values, all exhibiting the same magnitude. Since the vector $\mathbf{t}$ is constituted by the convolution of the CIR with the equaliser's impulse response, ideally $\mathbf{t}$ would be a Dirac delta function. The vectors $\mathbf{t}$ having more than one non–zero components represent saddle points, yielding unstable equilibria, which do not affect the equaliser's convergence performance. This can be shown by examining the second derivative of the tap update formulae in Equation (1.76) with respect to the equaliser tap vector $\mathbf{c}$. The second derivatives can be found in the same way as the first derivatives, with the exception that now the derivative is with respect to the conjugate of the equaliser tap vector $\mathbf{c}$, yielding:

$$\frac{\partial^2 J_{CMA}(n)}{\partial \mathbf{c} \partial \mathbf{c}^H} = \mathbf{H}^* \cdot \mathbf{T}' \cdot \mathbf{H}^T, \quad (1.124)$$

where

$$\mathbf{T}' = \mu_4 diag(2|t_i|^2 - 1) +$$

$$2\mu_2^2 \cdot (\mathbf{t}\mathbf{t}^H + \mathbf{t}^T\mathbf{t}^*\mathbf{I} - \\ -2 \cdot diag(|t_{-K_1}|^2, \cdots, |t_{K_2}|^2)) \quad (1.125)$$

or

$$[\mathbf{T}']_{i,j} = \begin{cases} \mu_4 \cdot (2|t_i|^2 - 1) + 2\mu_2^2 \sum_{k=-K_1, k\neq i}^{K_2} |t_k|^2 & i = j \\ 2\mu_2^2 \cdot t_i \cdot t_j^* & i \neq j. \end{cases} \quad (1.126)$$

The positive definiteness of a matrix $\mathbf{A}$ can be verified by considering the term $\mathbf{x}^T \cdot \mathbf{A} \cdot \mathbf{x}^*$. In our case, this term becomes:

$$\mathbf{x}^T \cdot \mathbf{H}^* \cdot \mathbf{T}' \cdot \mathbf{H}^T \cdot \mathbf{x}^* \quad = \\ (\mathbf{H}^H \cdot \mathbf{x})^T \cdot \mathbf{T}' \cdot (\mathbf{H}^H \cdot \mathbf{x})^*. \quad (1.127)$$

From this relationship we observe that if the matrix $\mathbf{T}'$ is positive (negative) definite, then the matrix $\mathbf{H}^* \cdot \mathbf{T}' \cdot \mathbf{H}^T$ is also positive (negative) definite. Therefore, it is sufficient to estimate the positive definiteness of $\mathbf{T}'$. By examining the sub–determinants of $\mathbf{T}'$, we can easily see that if only one entry of the vector $\mathbf{t}$ is non–zero, then all of the sub–determinants have the form μ_4^i and are positive, which implies that the matrix is positive definite and the associated error surface point is a local minimum. Finally, all the other solutions, for which more than one component of the vector $\mathbf{t}$ is non–zero, have positive and negative subdeterminants for the matrix $\mathbf{T}'$, thus constituting saddle points.

The stationary points of the Modified CMA [102] of Section 1.3.5.3 have a similar form to the stationary points of the CMA [4] of Section 1.3.5.2, as explored by Wesolowsky in [102], since the cost–functions of these two algorithms are similar.

Above we have found the stationary and saddle points of the CMA under the assumption that the rows of the channel matrix are linearly dependent. In general, this assumption is not true, unless the number of feedforward and feedback taps of the equaliser is infinite. In practice, however, for a number of taps, that is substantially higher than the channel order, this is approximately true. In practical cases the number of taps is sufficiently high, when it is more than three times the channel order. Ding *et al.* showed in [22, 113] that if the number of equaliser taps is not infinite, depending on the form of the CIR, undesirable equilibria may be present. The authors considered a specific channel, namely the autoregressive channel and found the undesirable equilibria associated with this channel. These are length–dependent equilibria, which are likely to exist in a similar form for every other Bussgang equaliser. This can be seen, if we recall Equation (1.118). The multiplicative matrix $\mathbf{H}^*$ in Equation (1.118), which is responsible for the undesirable equilibria explored by Ding *et al.* [22], stems from the $\mathbf{y}^*$ factor multiplying the error term in Equation (1.54). This equation is common

to all Bussgang equalisers, which implies that the term $\mathbf{H}^*$ will exist for any other algorithm of this type and therefore all these algorithms will exhibit these unstable equilibria. Interestingly, the same term is encountered in the context of the LMS equaliser of Section 1.2.0.2, which indicates that there should be a correspondence between the local minima of the Bussgang algorithms exhibiting only length–dependent local minima and the LMS equaliser. In fact, it has been shown by Zeng in [120] as well as by Li and Tong in [119] that there is a one–to–one correspondence between the local minima of the CMA and those of the LMS algorithms under certain assumptions, including amongst others, that the modulation scheme used is BPSK and that the noise variance is low. Therefore, despite the lack of a proof for the general correspondence of the CMA's minima with the LMS minima, at least under certain specific circumstances this has been shown to be true. In Appendix E the above result is confirmed using a similar procedure.

1.3.7.3 Convergence in the Presence of Channel Noise

Let us now focus on the situation, where channel noise is inflicted and we repeat the same analysis under this assumption. In this case in Equation (1.117) we make the following substitutions:

$$\mathbf{y}(n) = \mathbf{H} \cdot \mathbf{a}(n) + \mathbf{e}(n) \tag{1.128}$$

$$z(n) = \mathbf{t}^T \cdot \mathbf{a}(n) + \mathbf{c}^T \cdot \mathbf{e}(n), \tag{1.129}$$

where all the signals and vectors have been defined in Equations (1.44)–(1.49). We consider the mean values of these signals over the following two random variables:

- the QAM symbols $a(n)$, which are assumed to be *independent identically distributed (i.i.d.)* and
- the noise samples $e(n)$, which are assumed to be uncorrrelated, having a zero mean value and a variance of σ^2, but no specific distribution is assumed for them.

Under these assumptions Equation (1.117) becomes (Appendix G):

$$\mathbf{H}^* \cdot \mathbf{T} \cdot \mathbf{t} + \sigma^2 \cdot \left(2\mu_2 \mathbf{H}^* ||\mathbf{c}||^2 \mathbf{t} + 2\mu_2 ||\mathbf{t}||^2 \mathbf{c} - R_2 \mathbf{c}\right) + \sigma_4^4 \left[|c_i|^2 c_i\right]_i + 2\sigma^4 \left[c_i \sum_{k \neq i} |c_k|^2\right]_i = \mathbf{0}, \tag{1.130}$$

where $[A_i]_i$ symbolises a column of matrix $\mathbf{A}$, the i–th element of which is A_i and σ_4^4 is the 4–th order moment of the noise samples' distribution. Comparing Equations (1.121) and (1.130) we observe the existence of three extra additive terms, all of which exist only because of the additive noise. Due to its complexity, to date no general solution of Equation (1.130) has been found. The stationary

points of the error surface in Sato's algorithm were explored in [115], while the stationary points of the modified CMA of Section 1.3.5.3, which were explored in [102], are similar to those of the CMA. A different type of analysis, suggested by Shynk and Chan in [127], assumed that the convolutional noise is normally distributed and provides similar results. Fractionally–spaced equalisers of this type have also been studied. It has been shown by Ding *et al.* [59, 126] that a sufficient condition for the existence of only one desirable global minimum is that the equaliser's length is at least equal to the channel's delay spread and, at the same time, there are no common zeros for the subchannels created by considering the z–transform of the oversampled channel. This condition implies that for the channel to be correctly equalised, the subchannels should be uncorrelated, so that invoking sampling more than once per symbol interval provides us with independent information. More explicitly, the subchannels are defined as the CIRs, which are delayed, with respect to the symbol–spaced CIR, by a fraction of the symbol interval. In mathematical terms, this means that the subchannel i would be defined as:

$$\mathbf{h}_i = [h(-L_1 \cdot T + i \cdot T_s), \cdots, h(L_2 \cdot T + i \cdot T_s)]^T \tag{1.131}$$

where T is the symbol duration and $T_s = T/\Delta$, while Δ is the oversampling factor. In other words this means that the associated subchannels are reasonably diverse. Endres *et al.* [81] have also explored the scenario, where the first of these conditions was not met. In closing, we note that the extension of Bussgang equalisers to fast RLS estimation based schemes has been proposed by Douglas *et al.* in [128] as well as by Papadias and Slock in [110].

1.3.7.4 Avoiding the Undesirable Equilibria of Bussgang Equalisers

1.3.7.4.1 Detecting Ill–Convergence In this section we will propose a way of overcoming the problem of undesirable equilibria of the Bussgang blind equalisers of Section 1.3.7.2. The idea goes back to 1994, when Dogancay and Kennedy [129] studied the feasibility of testing for the convergence of a decision–directed equaliser provided that only the probability distribution of the input sequence, which is BPSK, is known. In this study we follow a different procedure. In order to decide whether the equaliser has converged or not, we will statistically test the following hypothesis:

- H1: The equaliser has converged. In this case $|t_i| = 1$ for some i and $t_j = 0$ for $j \neq i$.

- H2: The equaliser has not converged. In this case there are more than one j indices for which $t_j \neq 0$.

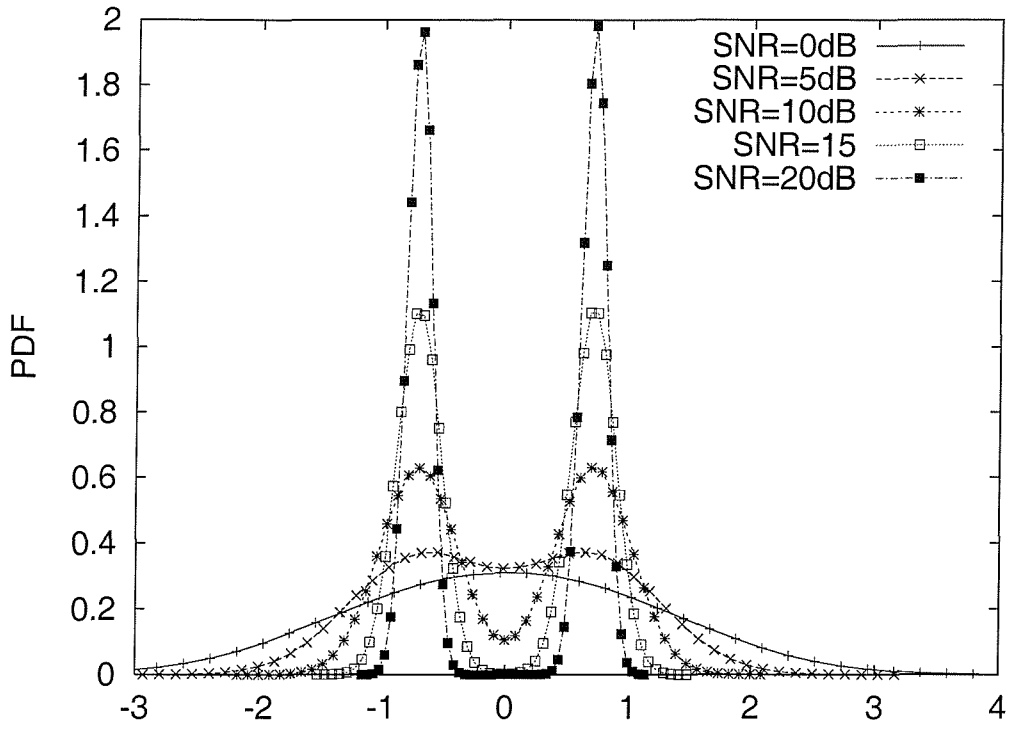


Figure 1.22: The PDF of the real or imaginary component of the equalised signal under H1 for QPSK for various channel SNRs

If H1 is valid, then the equalised signal will be of the following form:

$$z(n) = a(n-\delta) + \sum_i c_i \cdot e(n-i), \tag{1.132}$$

where δ is the equaliser decision's delay (which can be zero) and $e(n)$ represents the noise samples, which we assume to be uncorrelated white Gaussian noise. An obvious way of testing the hypothesis H1 is to test, whether the distribution of the equalised symbols matches the type of distribution that we would expect under H1. In Figure 1.22 we have plotted the Probability Density Function (PDF) of the real or imaginary part of the equalised noise–contaminated symbols for QPSK under the hypothesis H1 for various channel SNRs. The same plot is given in Figures 1.23 and 1.24 for 16–QAM and 64–QAM respectively. We observe that the PDF approaches the Gaussian distribution when the SNR is low and the discrete uniform distribution of the QAM symbols when the SNR is high. Based on the measured pdf of the equalised symbols we can run a statistical inference test – such as for example the Kolmogorov–Smirnov or the Chi–square test [130] – to decide whether the H1 hypothesis is valid. If in reality the hypothesis H2 is valid instead of H1, then the distribution of the equalised symbols is that of numerous weighted uniformly distributed variables, stemming from the ISI, plus numerous weighted Gaussian distributed variables having a zero mean value, which stem from the noise filtered by the equaliser. Hence, the latter component becomes a non–white Gaussian variable.

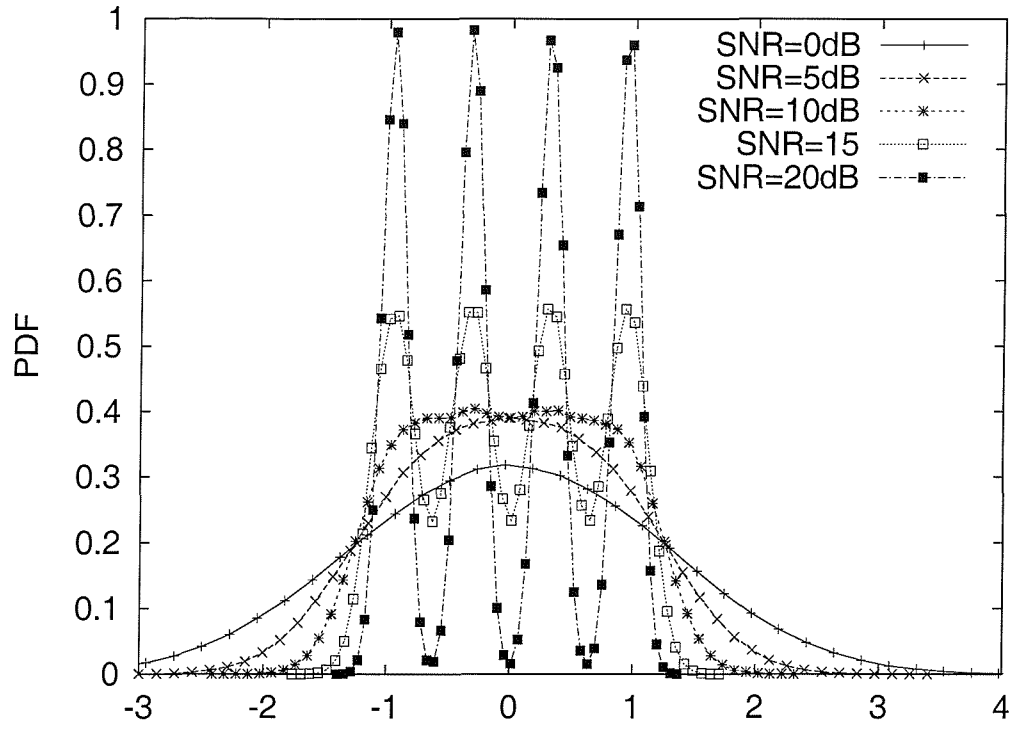


Figure 1.23: The PDF of the real or imaginary part of the equalised signal under H1 for 16–QAM for various channel SNRs

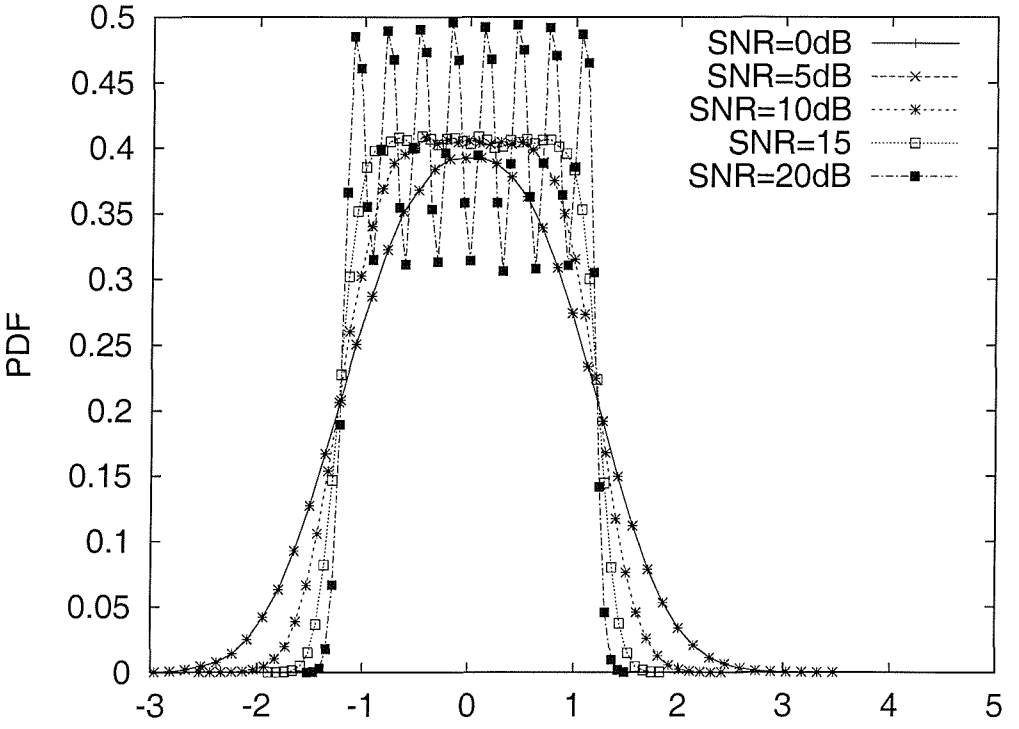


Figure 1.24: The PDF of the real or imaginary part of the equalised signal under H1 for 64–QAM for various channel SNRs

The former component is no longer the uniform distribution, which it was under H1. Ill–convergence can be detected, if the hypothesis H2 is detectable on the basis of recognising the above–mentioned bi–modal PDFs.

1.3.7.4.2 Avoiding Ill–Convergence Situations Once an ill–convergence situation has been detected, the task of avoiding this situation arises. This can be achieved by re–initialising the equaliser far from the current point of convergence. The question in this case is how the equaliser taps should be re–initialised. Various initialisation strategies have been suggested for blind equalisers [131, 132], but the generic solution to this problem is difficult to find. A conceptually simple way of achieving this would be to retain all the unstable equaliser tap sets in memory and to re–initialise the equaliser far from these tap sets. The equaliser vector's trajectory could also be used for providing information about the region of the unstable equilibrium so that the reinitialisation point can be adjusted to be sufficiently far from this region. If the equaliser converges again to another undesirable equilibrium, then the unstable tap sets stored in memory are updated. This procedure assumes, however, that the channel is time–invariant. The equaliser is never "allowed" to return to the tap setttings that have already been classified as unstable.

1.3.8 Controlling the Equaliser Parameters

When designing a linear blind equaliser, an important issue to be considered is how to set the equaliser parameters. The basic parameters to be set are the equaliser order and the step–size. Naturally, in specific equalisers there may be a range of other extra parameters to be set. For example, a Benveniste–Goursat equaliser of Section 1.3.5.4 also requires the adjustment of the k_1 and k_2 parameters. In a study concerning this issue, which was conducted by Wesolowsky in [133], a procedure of smoothly switching between two different sets of step–sizes and filter orders is proposed for the Stop–and–Go algorithm.

Selecting the equaliser order appropriately is crucial, which has to be based on an estimation of the channel's impulse response. Both the length and the shape of this CIR is of importance. If the CIR duration is underestimated, then the equaliser order may not be sufficiently high and the resultant residual ISI will corrupt the signal. On the other hand, increasing the equaliser order indiscriminately is detrimental, since:

- The equaliser complexity will be increased. It has to be noted, however, that the complexity of linear equalisers increases only linearly with the number of equaliser taps, as will be seen in Section 1.3.12 and therefore the complexity increase is not significant.

- The speed of convergence will be decreased as a result of the parallel adaptation of an increased number of equaliser taps.

- The convolutional noise will be increased. Usually, when the equaliser has converged to a desirable point, the equaliser taps around the main tap have a large value, while the taps which are substantially delayed with respect to the main tap have relatively small values. When the number of equaliser taps is increased, the small–valued blind equaliser taps, which are far from the main tap, are often not updated correctly, since their optimum values are too small.

A way of overcoming the problems of excessive equaliser order is to retain a high number of equaliser taps and to estimate which of the equaliser taps are similar in terms of their magnitude to the value of the step–size. These taps are close to zero and their contribution to the equalisation process is random and therefore unnecessary. By forcing them to zero we can eliminate some taps, once the system has decided that they are insignificant. This reduces both the associated complexity and the convolutional noise. In order to determine, which taps can be eliminated during the equalisation process, we should identify the specific taps having relatively low values with respect to the main taps, which also exhibit a large variance with respect to their mean value, implying that these values are changing randomly, rather than converging. The equaliser's step–size, on the other hand, is responsible for the speed of convergence and also for the accuracy of convergence of the blind equalisers concerned. When the CIR is time–variant, the equaliser has to able to adapt to the CIR changes. The Bussgang equalisers do not exhibit fast convergence characteristics that would allow them to adapt to the typically rapid CIR variations occuring in mobile environments, except when the mobile speed is low and hence the CIR is slowly varying.

If the CIR does not change significantly during a transmission frame, invoking the blind equaliser in a forward–inverse manner, as seen in Figure 1.25 following Letaief *et al.* [134], can equalise the channel, provided that a sufficiently high number of QAM symbols can be processed. More explicitly, in Figure 1.25 this involves feeding the equaliser with the received input frame $a(n), \cdots, a(n-N)$, then continuing by feeding the time–reversed frame $a(n-N), \cdots a(n)$ to the equaliser and so forth, until the equalisation becomes perfect. Again, this procedure is illustrated in Figure 1.25. One might wonder, why we actually need to reverse the signal input to the equaliser during the odd–indexed iterative steps. The answer lies in an assumption that all these equalisers have to satisfy in order to work, namely that the input signal is i.i.d.. Independence is the assumption, which is violated when we feed the same input frame several times to the equaliser. By occasionally reversing the input frame, we create pseudo–randomness, as in turbo–equalisation for example [135], and hence the equaliser converges more rapidly. When randomness does not exist in the transmitted data, then the Bussgang equelisers fail to converge to the ideal point of equilibrium. Instead, they converge to points, which

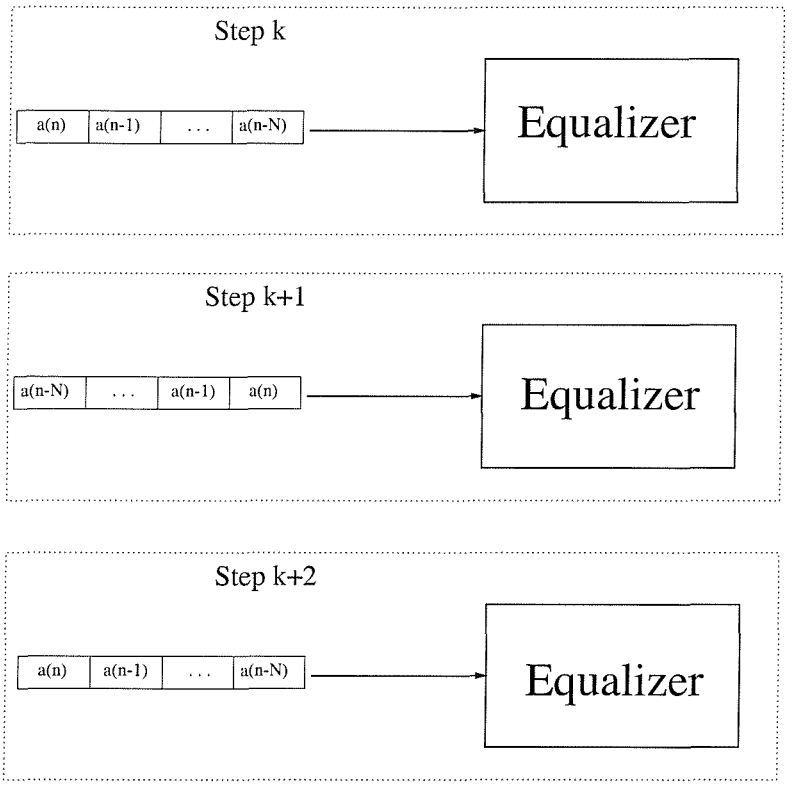


Figure 1.25: Forward–inverse equalisation scheme [134]

are shifted with respect to the ideal point in the vector space and this shift is a function of the exact correlation between the transmitted data symbols. However, we use the word "randomness" in the sense that the correlations between the data, from the first to the last, are all zero. In practical cases some correlation exists amongst the data symbols. This correlation implies a low–pass filtered power spectral density and the bandwidth of the corresponding hypothetical low–pass filter has a substantial influence on the Bussgang equaliser's convergence. If this bandwidth is sufficiently high with respect to the reciprocal of the equaliser's adaptation time, then 'sufficient randomness' is present for the equaliser's convergence point to be associated with a low BER performance degradation.

As has been mentioned already, a small step–size guarantees good convergence accuracy, while a large step–size results in poor acccuracy and it also might drive the equaliser to instability. Each equaliser has its own convenient range of step–size values to be used, which also depends on the modulation scheme used. In one of the original contributions on Bussgang blind equalisers, Godard [4] suggested using a variable step–size for his constant–modulus equalisers. Commencing with a large step–size ensures fast adaptation towards the point of convergence and then reducing the step–size upon approaching convergence offers the required accuracy.

Another point that we will emphasize here is the inter–dependence of the step–size and the equaliser order. Specifically, concerning the interactions between the step–size parameter and the equaliser

order, it should be noted that increasing the number of equaliser taps generally requires a smaller step–size value for avoiding instability, while maintaining the same accuracy. This becomes plausible, if we recall a point we made concerning the equaliser order, namely that the higher the equaliser order, the higher the convolutional noise, provided the equaliser order is already sufficiently high for removing the ISI. By decreasing the step–size, the equaliser converges with better accuracy and this reduces the convolutional noise, which was engendered by the increase in the number of taps. The above statements are true not only for linear equalisers. They are more general and all the blind equalisers of iterative nature obey these properties. For example, the sequence estimation techniques, which will be presented in Section 1.3.9 also use LMS (or RLS) type channel estimation that have the same characteristics as the iterative Bussgang techniques. Various studies of the equaliser parameter–control issues have been conducted [136–139]. Below we give a more indepth analysis of the equaliser parameters' influence on the achievable performance.

1.3.8.1 Convergence Accuracy

In order to find the optimum equaliser order for a given channel, it is beneficial to observe the Mean Squared Error (MSE) of the equalised signal measured with respect to the input signal, which is defined as:

$$MSE = E[|z(n) - a(n)|^2]. \tag{1.133}$$

The right equaliser order is the one which gives the minimum MSE. Therefore, the MSE has to be evaluated as a function of the equaliser order, which is typically an arduous task. This is a very complicated function to estimate. However, we will invoke some useful approximations, which will simplify this function. We commence by noting that if $\mathbf{e}(n)$, $\mathbf{t}$, $\mathbf{c}$ and $\mathbf{a}(n)$ are as defined in Equations (1.44)–(1.48), then with the aid of Equation (1.53) we can express the MSE term of Equation (1.133) as:

$$\begin{aligned} MSE &= E\left[\left|\mathbf{t}^T\mathbf{a}(n) + \mathbf{c}^T\mathbf{e}(n) - a(n)\right|^2\right] \\ &= E\left[(\mathbf{t}^T\mathbf{a}(n) + \mathbf{c}^T\mathbf{e}(n) - a(n))(\mathbf{a}^H(n)\mathbf{t}^* + \mathbf{e}^H(n)\mathbf{c}^* - a^*(n))\right] \\ &= \mu_2\left(||\mathbf{t}||^2 - 2Re\{t_\delta\} + 1\right) + \sigma^2||\mathbf{c}||^2, \end{aligned} \tag{1.134}$$

where $||\mathbf{x}||^2$ is the L_2–norm of vector $\mathbf{x}$, σ^2 is the noise variance, μ_2 is the mean value of the square of the transmitted QAM symbols $a(n)$ and δ is the delay associated with the equaliser's convergence to a delayed solution. Furthermore, we have assumed that the noise is uncorrelated with the input signal $a(n)$, implying that no co–channel interference exists. It can be seen from Equation (1.134) that the non–zero MSE is the result of two corrupting factors, namely that of:

- the additive noise (term $\sigma^2 \cdot ||\mathbf{c}||^2$)
- the intersymbol interference (term $\mu_2 \left(||\mathbf{t}||^2 - 2Re\{t_\delta\} + 1\right)$).

Let us now study these two factors separately in the next paragraphs.
Firstly, the effect of additive channel noise incorporates the filtering of the noise term by the equaliser. As we can see in Equation (1.134), this filtering results in the noise power being multiplied by the L_2 norm $||\mathbf{c}||^2$ of the equaliser vector $\mathbf{c}$. Therefore, we expect a degradation of the MSE due to noise, which is not exactly equal to the genuine channel noise power, but rather to the sum of the noise power and the L_2 norm of the equaliser vector, when expressed in the logarithmic domain, in terms of dB. This will be used to investigate the excess ISI which is produced by the noise's influence on the equaliser's tap estimation, as we will see in the next section. Secondly, the ISI exists for three basic reasons:

- the number of taps of the equaliser is insufficient
- despite the sufficiently high equaliser order there exists a non–zero steady–state MSE for the equaliser associated with the specific modulation scheme and the equalisation algorithm used
- the additive noise power is sufficiently high for limiting the blind equaliser's capability of estimating the right tap values.
- imperfect transmitter and receiver filtering, which results in ISI.

1.3.8.1.1 Channel Order Mismatch The first of these reasons simply implies that the number of equaliser taps is lower than that necessary for near–perfect equalisation, i.e. for attaining a sufficiently low ISI in the absence of additive noise. The behaviour of the Bussgang equalisers, when their number of taps is low and the channel noise power is also low, is similar to that of the LMS equaliser. This is because in this case the generally superior LMS equaliser exhibits an MSE performance, which is dictated by the limited number of equaliser taps. In this case, by designing the LMS equaliser required for the channel encountered we can estimate its transfer function and the corresponding ISI for each equaliser order. This procedure is shown by means of a simulation example in Figure 1.26.

1.3.8.1.2 Steady–State MSE The second one of the above–mentioned reasons is true generally for the family of blind equalisers, since they usually exhibit a steady–state error. For example the classic CMA has a zero steady–state error, when used in conjunction with QPSK but a non–zero MSE when used for 16– or 64–QAM. This can be readily interpreted by observing the tap–update procedure

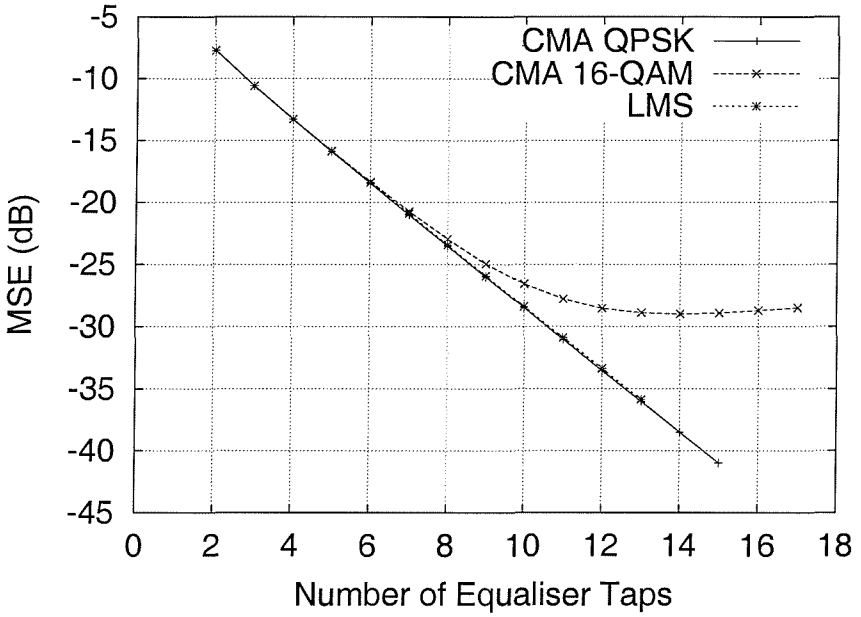


Figure 1.26: MSE as a function of the equaliser order for QPSK and 16–QAM using the CMA. The channel model used is the one–symbol delay CIR as given in Figure 1.27. The LMS MSE is again given as a benchmark performance curve. We can observe that while the MSE curve of QPSK coincides with that of the LMS because of the zero mean squared value of the update term of Equation (1.76), the 16–QAM exhibits a non–zero steady–state MSE.

of this algorithm from Equation (1.76), which is repeated here for convenience:

$$\mathbf{c}^{(n+1)} = \mathbf{c}^{(n)} + \lambda \mathbf{y}^*(n) z(n) (|z(n)|^2 - R_p). \tag{1.76}$$

Explicitly, the update term $\lambda \mathbf{y}^*(n) z(n) (|z(n)|^2 - R_p)$ will give zero variance for all PSK symbols, since in this case it is identically zero for all the phasors of the modulation constellation, but it will give a non–zero variance for 16– or 64–QAM, since in this case there exist points in the constellation for which the term is non–zero. In fact this is the case for all the constellation points. In Figure 1.26 this is demonstrated for the CMA algorithm in the context of both QPSK and 16–QAM. Because of the non–zero mean squared value of the update term of Equation (1.76), a modulation–dependent ISI term will set a lower limit for the MSE which cannot be removed. It can be seen in Equation (1.76) that this MSE limit depends quadratically on the step–size parameter λ. Explicitly, this is because of the linear dependence of the blind estimation error of Equation (1.76) on λ, which results in a quadratic dependence on λ in terms of the MSE. Indeed, plotting the converged–state MSE for the CMA based equaliser in the context of 16–QAM as a function of the equaliser order for different λ values corroborates this observation, as it is seen in Figure 1.28. The channel model used in this example is given in Figure 1.27.

We will now give a mathematical justification of these observations based on the Bussgang blind

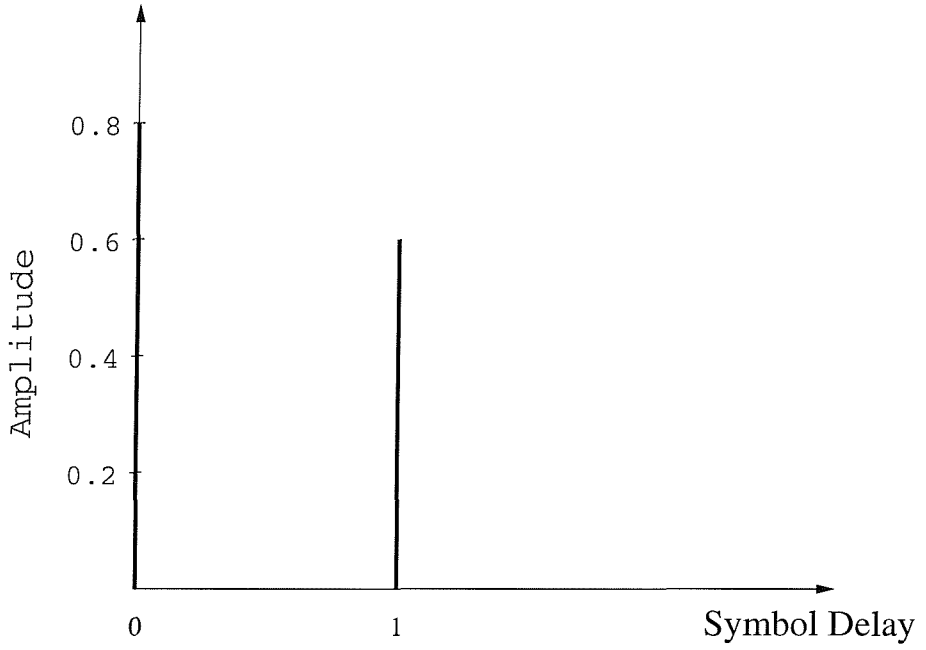


Figure 1.27: The two–path CIR model used in the simulations in this section

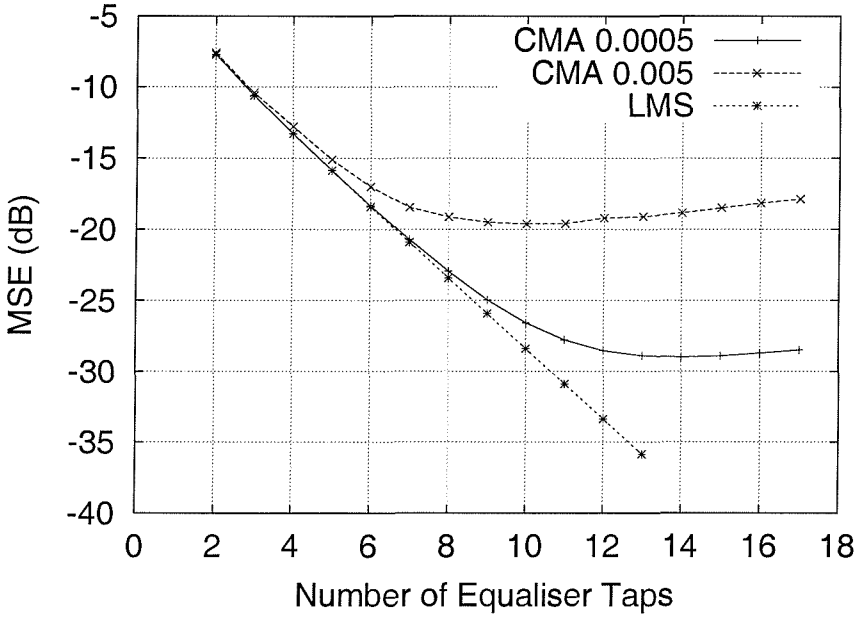


Figure 1.28: MSE for the CMA step–size values of $\lambda = 0.005$ and 0.0005 as a function of the equaliser order for 16–QAM using the CMA. The channel model used is the one–symbol delay CIR given in Figure 1.27. The LMS MSE is given as a lower limit, since this is a trained equaliser, which gives a benchmark performance. The curves corresponding to the two λ values are separated by $10dB$, which corresponds to the square of the difference between the two λ values in terms of dB.

equalisers of Section 1.3.5. In general, a Bussgang blind equaliser updates the equaliser taps according to the update formula of Equation (1.54), which is repeated here for convenience:

$$\mathbf{c}^{n+1} = \mathbf{c}^{(n)} - \lambda \cdot \mathbf{y}^*(n) \cdot \epsilon(n), \tag{1.135}$$

where $\epsilon(n)$ is the algorithm–specific estimation error. For example, by observing the CMA's tap update formula of equation (1.76), we can easily see that the error becomes:

$$e_{CMA}(n) = -z(n) \cdot (|z(n)|^2 - R_p). \tag{1.136}$$

Using Equation (1.135), we can arrive at the MSE term for a Bussgang equaliser pursuing the following approach. The equalised symbol at the input of the decision device in Figure 1.2 at time n is given by:

$$\begin{aligned} z(n) &= \mathbf{c}_{(n)}^T \cdot \mathbf{y}(n) \\ &= \mathbf{c}_{(n)}^T \cdot (\mathbf{H}\mathbf{a}(n)) = \mathbf{t}_{(n)}^T \mathbf{a}(n) \\ &\overset{(1.135)}{=} \left(\mathbf{c}_{(n-1)}^T - \lambda \cdot \mathbf{y}^H(n) \cdot \epsilon(n-1)\right) \cdot (\mathbf{H}\mathbf{a}(n)) \\ &= \mathbf{c}_{(n-1)}^T \mathbf{H}\mathbf{a}(n) - \lambda \mathbf{y}^H(n-1)\mathbf{H}\mathbf{a}(n) \\ &= \mathbf{t}_{(n-1)}^T \mathbf{a}(n) - \lambda \mathbf{a}^H(n-1)\mathbf{H}^H\mathbf{H}\mathbf{a}(n)\epsilon(n-1). \end{aligned} \tag{1.137}$$

Note that the index $_{(n)}$ in the subscript indicates the n–th iteration. The MSE can be expressed with the aid of this equation as:

$$MSE = E\left[\left|\mathbf{t}_{(n-1)}^T \mathbf{a}(n) - \lambda \mathbf{a}^H(n-1)\mathbf{H}^H\mathbf{H}\mathbf{a}(n)\epsilon(n-1) - a(n)\right|^2\right], \tag{1.138}$$

which can be simplified to:

$$MSE = \mu_2 \left(||t||^2 + 1 - 2Re\{t_\delta\}\right) + \lambda f_1 + \lambda^2 f_2, \tag{1.139}$$

where

$$f_1 = 2Re\{E\left[\mathbf{t}^H\mathbf{a}^*(n)\mathbf{a}^H(n-1)\mathbf{H}^H\mathbf{H}\mathbf{a}(n)\epsilon(n-1) - a^*(n)\mathbf{a}^H(n-1)\mathbf{H}^H\mathbf{H}\mathbf{a}(n)\epsilon(n-1)\right] \tag{1.140}$$

$$f_2 = E\left[\mathbf{a}^H(n-1)\mathbf{H}^H\mathbf{H}\mathbf{a}(n)\mathbf{a}^T(n-1)\mathbf{H}^T\mathbf{H}^*\mathbf{a}^*(n)|\epsilon(n-1)|^2\right]. \tag{1.141}$$

Equation (1.139) consists of three terms. The first term represents the error produced by the residual ISI, which becomes zero, when the combined channel plus equaliser impulse response vector $\mathbf{t}$ takes the ideal form of $(1, 0, \ldots, 0)$, i.e. when the equaliser manages to supress all the ISI. The second term of Equation (1.139) is a λ–dependent term, which has a mean value determined by higher order moments according to Equation (1.140). Finally, the third term is a λ^2–dependent term which also exhibits a mean value related to higher order moments as seen in Equation (1.141). The reason we have not simplified Equation (1.139) is that it actually depends on the algorithm–specific error estimate $\epsilon(n-1)$. This error is generally not entirely uncorrelated with the transmitted symbol vector $\mathbf{a}(n)$, which renders the calculation of Equation (1.139) more complicated. Moreover, the channel impulse response must also be known for evaluating the terms of Equation (1.139). What is important here is that we observe the λ^2–related dependence of the MSE in Equation (1.139) that we also observed in the simulation example of Figure 1.28. Experimental evidence suggests that the term f_1 is insignificant compared to the term f_2, which is responsible for the λ^2–dependence of the MSE in Equation (1.139) in the case studied.

By contrast, the dependence of the MSE on the equaliser order is implicit in all three terms of Equation (1.139). The step–size dependence of the first term is indirect, because the more taps the equaliser has, the more taps will contribute to the L_2–norm of the vector $\mathbf{t}$. In excess of a certain number of taps, increasing the equaliser order will not improve the MSE, but it will create more non–zero terms in the L_2–norm of $\mathbf{t}$. The dependence of f_1 and f_2 on the number of equaliser taps is not explicit because of the complexity of Equations (1.139), but the equaliser order determines the dimension of the arrays and matrices in these equations, which in turn determine the effects of channel noise on the equaliser taps.

1.3.8.1.3 The Effect of Noise on the Equaliser Tap Adaptation Having studied the infiuence of the equalisation algorithm, the equaliser order and the step–size on the residual ISI, we will now focus our attention on the influence of channel noise on the equaliser tap estimation. It is expected that due to channel noise the equaliser will have a degraded performance, which will exceed the excess MSE directly introduced into the receiver by the additive channel noise. In other words, if we have a channel SNR of $+15dB$, then we expect an MSE value in the range of $[-10, \cdots, -15]dB$, since the noise will affect the equaliser's tap estimation process, resulting in imperfect equalisation, even if the equaliser order is sufficiently high. The gravity of this performance degradation depends on the specific type of equaliser, the equaliser order, the CIR and the modulation scheme used. In Figure 1.29 the MSE performance of a CMA–assisted 16–QAM modem is shown for different SNR values.
As in all previous sections, we will now provide a mathematical characterisation of the effects of

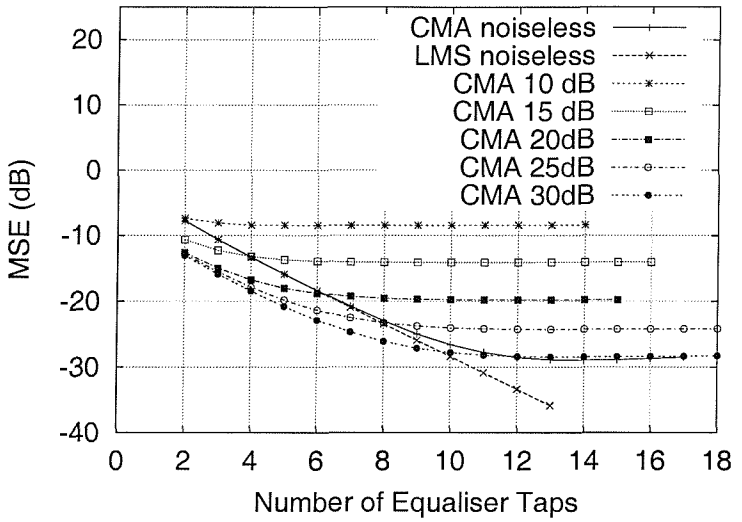


Figure 1.29: MSE as a function of the equaliser order and the SNR for a CMA–assisted 16–QAM modem. The channel model used is the one–symbol delay CIR given in Figure 1.27. In these curves, the MSE due to the channel's AWGN has been subtracted, in order to quantify the direct influence of the channel noise on the equaliser tap estimation process, which enhances the ISI. We observe that for a sufficiently large number of taps the equaliser's performance is only slightly affected by this effect, except when the SNR is low. For SNRs as low as $10dB$ the noise's influence is approximately $2dB$ while for all other SNR values it is less than $1dB$. Note that these MSE curves exhibit a positive slope for high SNRs, because the MSE in Equation (1.139) is bound to depend linearly on the number of taps.

channel noise. We commence by referring to Equation (1.137), which is rewritten as:

$$\mathbf{t}_{(n)} = \mathbf{t}_{(n-1)} - \lambda \mathbf{H}^T \cdot \mathbf{y}^*(n-1)\epsilon(n-1). \tag{1.142}$$

In this form, noting that

$$\mathbf{y}(n-1) = \mathbf{H}\mathbf{a}(n-1) + \mathbf{e}(n-1), \tag{1.143}$$

we can rewrite Equation (1.142) as:

$$\mathbf{t}_{(n)} = \mathbf{t}_{(n-1)} - \lambda \mathbf{H}^T \cdot (\mathbf{H}^*\mathbf{a}^*(n-1) + \mathbf{e}(n-1))\epsilon(n-1). \tag{1.144}$$

The mean value of the update term of this equation, which is constituted by the second term, can be formulated as:

$$E[\Delta\mathbf{t}] = E[\mathbf{t}_{(n)} - \mathbf{t}_{(n-1)}] = \lambda E\left[\left(\mathbf{H}^T\mathbf{H}^*\mathbf{a}^*(n-1)\epsilon(n-1) + \mathbf{H}^T\mathbf{e}(n-1))\epsilon(n-1)\right)\right]. \tag{1.145}$$

Since the error term $\epsilon(n-1)$ of Equation (1.145) may contain terms such as the equalised symbols $z(n)$, which are corrupted by the channel noise, the mean value of $\Delta\mathbf{t}$ in Equation (1.145) depends

on the algorithm–specific error term. For example, for the CMA, this error term can be found after lengthy calculations, yielding:

$$E[\Delta\mathbf{t}] = \lambda\mathbf{H}^T\left(\mathbf{H}^* \cdot \mathbf{T} \cdot \mathbf{t} + \sigma^2 \cdot \left(2\mu_2\mathbf{H}^*||\mathbf{c}||^2\mathbf{t} + 2\mu_2||\mathbf{t}||^2\mathbf{c} - R_2\mathbf{c}\right) + \sigma_4^4\left[|c_i|^2c_i\right]_i + 2\sigma^4\left[c_i\sum_{k\neq i}|c_k|^2\right]_i\right), \tag{1.146}$$

where σ_4^4 and $\mathbf{T}$ were defined in Section 1.3.7. Equation (1.146) indicates that the mean value of the update term is generally non–zero. This explains the observed fact that the noise actually affects the equaliser's ability to converge to the optimum point of equilibrium. From Equation (1.146) we can also observe that $\Delta\mathbf{t}$ depends both on the power and on the 4–th order moment of the channel noise, as well as on the step–size parameter λ. For a low noise power, the 4–th order moment diminishes in the sum, but for a high noise power both terms of Equation (1.146) become important. Finally, according to Equation (1.145) the step–size λ and $\Delta\mathbf{t}$ are linearly related to each other, hence the MSE corresponding to this type of ISI is proportional to λ^2. Note however that this ISI term is not the predominant ISI contribution in Equation (1.146), except when the noise power is sufficiently high in order to render $\Delta\mathbf{t}$ of Equation (1.146) sufficiently high for driving the vector $\mathbf{t}$ to a state of convergence, which is associated with a higher ISI than the ISI produced by the other factors, discussed in the previous paragraphs.

1.3.8.2 Convergence Speed

1.3.8.2.1 Noiseless Environment In this section we study the convergence behaviour of the blind equalisers, as before, focusing on the CMA equaliser, and provide simulation examples for justifying our conclusions. We commence with the zero–noise assumption, which will assist us in understanding the equaliser's behaviour without corrupting effects or impairments. We will firstly make a remark which stems directly from the CMA tap–update formula of Equation (1.76). Since the coefficient update is "limited" by the step–size λ at each symbol interval, this parameter controls the speed of convergence. In conjunction with a large step–size the equaliser is expected to converge faster, provided that the step–size is not so high as to affect the equaliser's convergence capability. This limitation will be discussed later in this section. In fact, when the step–size is sufficiently small, it is the only factor which limits the speed of convergence. This is evidenced by Figure 1.30, where the MSE learning curves are portrayed for 16–QAM using the CMA and for different equaliser orders. It is clear that all learning curves have the same slope at their initial stage of convergence. When λ is small, then it determines the slope of the curve during the learning stage. In other words, this slope is determined by λ. We can also observe in this figure that the variation in the convergence speed for different number of equaliser taps is simply due to the fact that each MSE curve reaches its

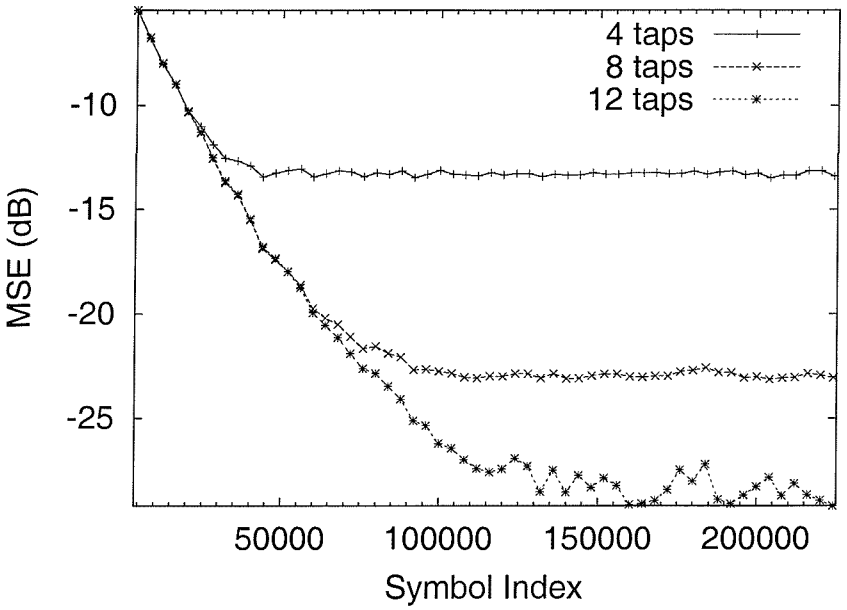


Figure 1.30: MSE learning curves for different equaliser orders in the context of 16–QAM using the CMA. The channel model used is the one–symbol delay CIR given in Figure 1.27. We can observe that in all cases the slope of the curve at the initial stage is the same. However, their difference manifests itself in the steady–state MSE, which depends on the order of the equaliser. Suitable averaging of the MSE has been performed by using intervals of 1000 symbols and taking the average in these intervals so that the random fluctuations are smoothened

steady–state after a different convergence time. For example, the 4–tap curve reaches its steady–state earlier, since this steady–state MSE is the highest, while the 12–taps curve reaches its steady–state later, since its steady–state MSE is the lowest and hence it requires more "learning" time. In Figure 1.30 suitable averaging of the MSE has been performed by using intervals of 1000 symbols and taking the average in these intervals so that the random fluctuations are smoothened. In Figure 1.31 learning curves are provided for different step–size values, exhibiting a different slope for each curve. We can also observe the difference in terms of their steady–state MSE, which was studied in the previous section. We can see that there is an MSE floor level, which depends on the step–size, since a lower step–size value is associated with a more stable steady–state performance. Based on the discussions of the previous paragraph on the achievable convergence accuracy we can also give a more explicit reason for the λ–dependence of the convergence slope by recalling Equation (1.137), which describes the MSE of a Bussgang equaliser. We can rewrite this equation as:

$$\mathbf{t}_{(n)} = \mathbf{t}_{(n-1)} - \lambda \mathbf{H}^T \cdot \mathbf{y}^*(n-1)\epsilon(n-1). \tag{1.147}$$

Since the update of the complex combined channel plus equaliser impulse response vector $\mathbf{t}$ is proportional to λ, the slope of the learning curve is also expected to be proportional to λ, provided that the multiplier of λ is constant. In fact, it is not, because the estimation error $\epsilon(n)$ is approaching

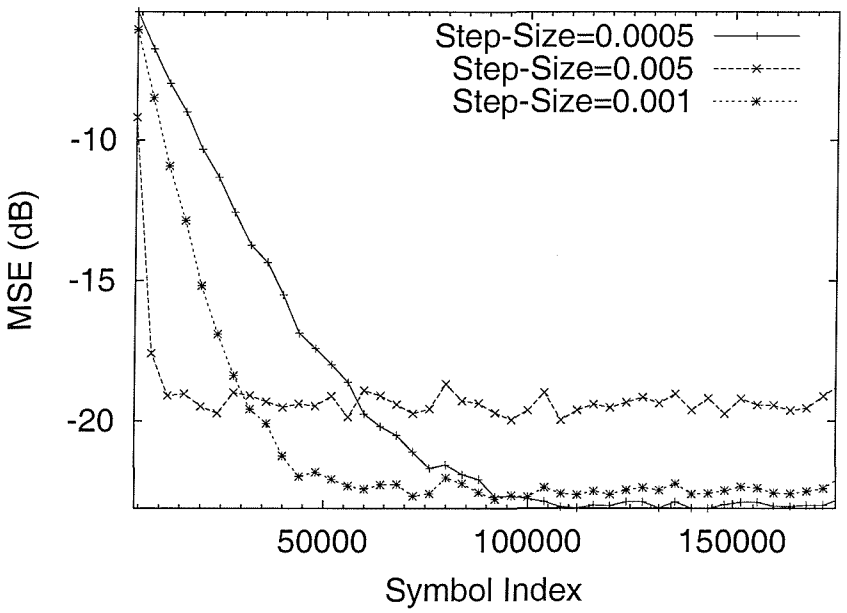


Figure 1.31: MSE learning curves for different step–size values λ in the context of 16–QAM using the CMA. The channel model used is the one–symbol delay CIR given in Figure 1.27. We can observe that the slope of the curve at the initial convergence stage is proportional to λ. In fact, the 0.0005 step–size gives a $-2.5\text{x}10^{-4} dB$/symbol slope, the 0.001 step–size gives a $-5\text{x}10^{-4} dB$/symbol slope, while the 0.005 step–size gives a $-2.5\text{x}10^{-3} dB$/symbol slope. These values correspond to the step–size differences in dB. Suitable averaging has been performed in the MSE as in Figure 1.30.

zero, as the equaliser approaches convergence. Nonetheless, the convergence speed of a blind equaliser of this type is low, which renders the multiplier of λ slowly varying, especially at the initial stage of convergence. We can observe this in Figure 1.31, where the learning curves have an initial slope depending on λ, which is gradually reduced and becomes zero, when convergence is reached.

1.3.8.2.2 Effect of Noise on the Convergence Speed In this paragraph we study the effect of channel noise on the convergence speed of blind equalisers. As we have already mentioned in the previous paragraph dedicated to the issues of convergence accuracy, when the noise power is low, the only effect of the channel noise is a steady–state MSE floor, which cannot be removed. According to Equation (1.134) this error floor corresponds to the sum of the noise power and the L_2 norm of the equaliser vector, expressed in terms of dB, as it was argued before, in Section 1.3.8.1. Nevertheless, when the noise power is high, the equaliser's convergence rate becomes poor. This phenomenon also contributes to the MSE floor, which becomes higher than what would be expected by the mere superposition of channel noise on the received signal. As a result, the convergence speed is increased under noisy conditions. This is because the noisy MSE floor level is higher, than the noiseless MSE floor, and hence it is reached within a reduced time in comparison to the lower noiseless MSE floor. The above statements are justified with an example in Figure 1.32. In this figure we have plotted the

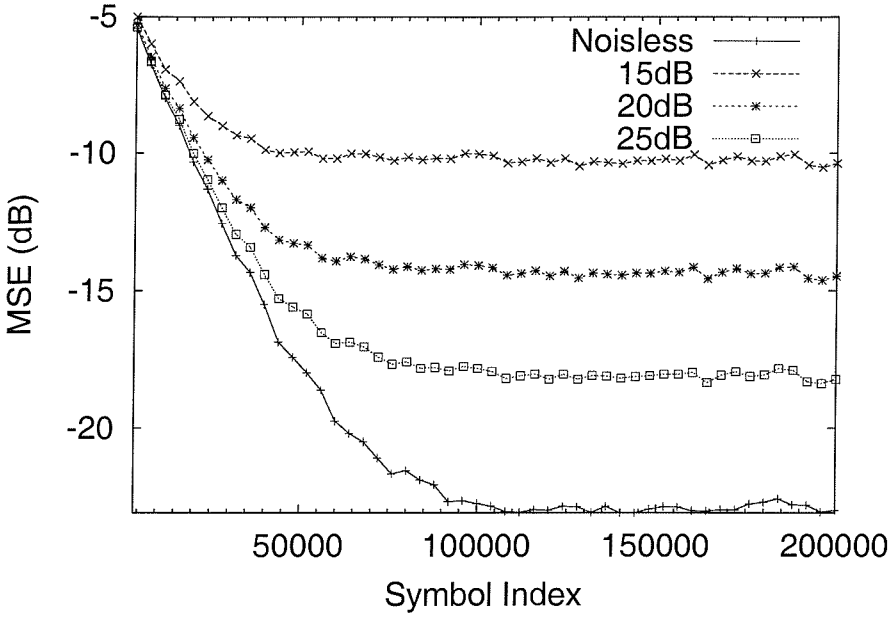


Figure 1.32: MSE learning curves for various SNR values for 16–QAM using the CMA. The channel model used is the one–symbol delay CIR given in Figure 1.27

MSE learning curves for different channel SNR values for 16–QAM using the CMA over the channel of Figure 1.27. We can readily observe that the noise is creating an SNR–dependent MSE floor level, increasing the convergence speed for lower SNRs.

Having highlighted the characteristics of the most salient Bussgang algorithms, let us now consider joint CIR and data estimation techniques.

1.3.9 Joint Channel and Data Estimation Techniques

Joint channel and data detection techniques constitute the blind equivalent of sequence estimation algorithms. These techniques have originally been used for data estimation, when the CIR was known. In this case the Viterbi algorithm is invoked to estimate the data sequence, assuming that the state machine is now produced by the channel instead of by the channel encoder. Here, instead of equalising the received signal, we estimate the CIR together with the ML data sequence, assuming that the CIR estimate is sufficiently accurate. Originally, Seshadri [6] observed that it may be risky to invoke the Viterbi algorithm [34] retaining only the strongest surviving path at each instant for the blind scenario, since by assuming perfect CIR knowledge, we would discard all the surviving paths but one. This would be a bad tactic at the initial stages of the algorithm, when the CIR estimation is poor. In fact, the less paths that are eliminated in favour of other paths the better the performance of this algorithm in the blind detection scenario. However, keeping all possible states would correspond to a computationally intensive algorithm, since the number of states grows exponentially with the channel

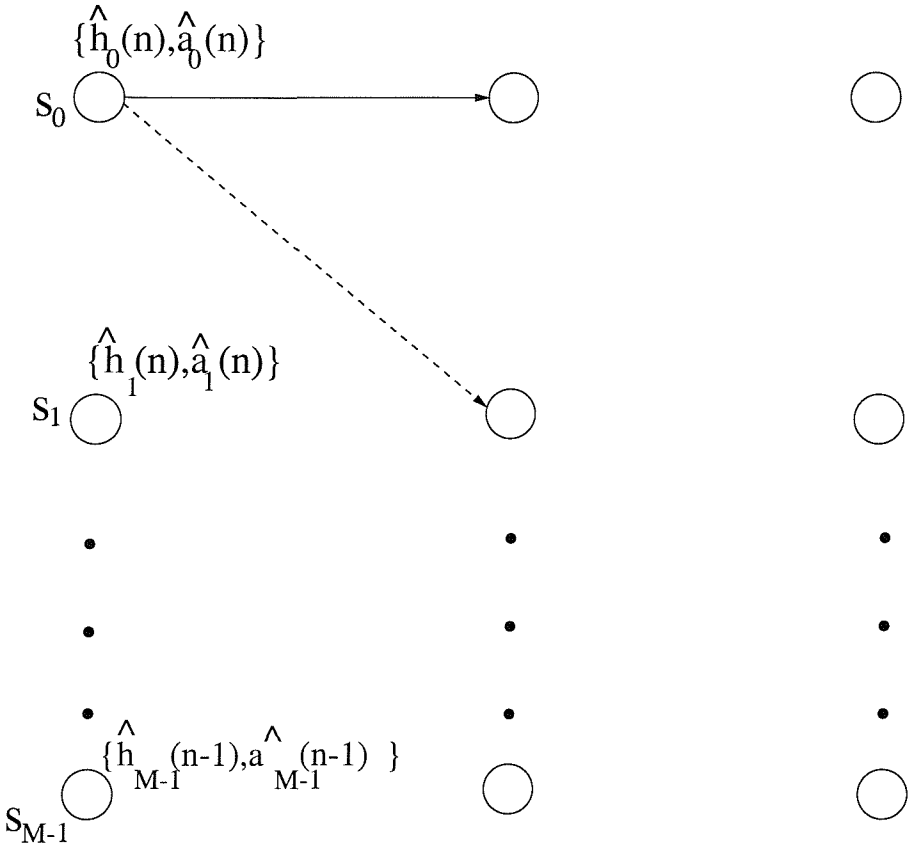


Figure 1.33: Per–survivor processing using the M–algorithm

length. Instead, Seshadri suggested that the so–called M–algorithm [140] should be used, retaining M number of surviving paths at each trellis state, as a compromise between retaining the full number of states or just one of them. He also suggested that the CIR estimation should be initialised and updated at any symbol interval using an LMS estimator. Each survivor path of the trellis should keep its own data plus CIR estimate, as illustrated in Figure 1.33. Here only the stylised trellis states and the associated M number of trellis transitions are indicated. For a deeper exposure to the M–algorithm the interested reader is referred to [140]. The LMS estimation was extended to RLS–based estimation by Xie *et al.* [141] for the case of multiuser communications, where the interference was due to K users, each having p distinct propagation paths of the dispersive channel. Raheli *et al.* [23] suggested that the robustness of this algorithm to noise would be boosted by protecting the data by channel coding and by performing joint channel decoding and sequence estimation. They introduced the term *Per–Survivor Processing (PSP)* and showed the applicability of this technique using RLS to fast fading channels, owing to its fast convergence.

In this section, we give a brief overview of joint CIR and data estimation using this technique. The definitions of the variables are the same as in Figure 1.8 and Section 1.3.5. Additionally, we define $\hat{\mathbf{h}} = [\hat{h}_{-L_1}, \cdots, \hat{h}_{L_2}]^T$ as the estimated CIR. Any transition from state s_n to state s_{n+1} in the trellis of Figure 1.33, describing the evolution of the state–machine modelling the channel, is due to the

estimated transmitted symbol vector $\hat{\mathbf{a}}(s_n \rightarrow s_{n+1}) = \hat{\mathbf{a}}(n)$. The estimated received symbol $\hat{y}(n)$ for this transition at time n is then given by the convolution of the tentatively assumed transmitted symbol vector $\hat{\mathbf{a}}(n)$ and the CIR $\hat{\mathbf{h}}^{(n)}$ assumed, which is formulated as $\hat{\mathbf{a}}^T(n) \cdot \hat{\mathbf{h}}^{(n)}$. Each surviving path in the trellis of Figure 1.33 will have its own CIR and data estimate. The LMS CIR estimation algorithm for each surviving path will then be [6]:

$$\hat{\mathbf{h}}^{(n+1)} = \hat{\mathbf{h}}^{(n)} - \lambda \cdot \hat{\mathbf{a}}^*(n) \cdot \left(y(n) - \underbrace{\sum_{i=0}^{L} \hat{\mathbf{a}}^T(n) \cdot \hat{\mathbf{h}}^{(n)}(n)}_{\hat{y}(n)} \right), \tag{1.148}$$

which is similar in its philosophy to the tap–update formula of Equation (1.135), updating $\mathbf{c}^{(n)}$ using the step–size λ depending on the error term in the round brackets (). The LMS CIR and data estimator can be replaced by an RLS estimator according to the following equations [15, 23]:

$$\begin{aligned}
\mathbf{\Phi}(s_n) &= \sum_{i=0}^{n} w^{n-i} \hat{\mathbf{a}}^*(i) \hat{\mathbf{a}}^T(i) = \mathbf{P}^{-1}(s_n) \\
\mathbf{k}(s_{n+1}) &= \frac{\mathbf{P}(s_n)\hat{\mathbf{a}}^*(n)}{w + \hat{\mathbf{a}}^T(n)\mathbf{P}(s_n)\hat{\mathbf{a}}^*(n)} \\
\mathbf{P}(s_{n+1}) &= \frac{1}{w}\Big(\mathbf{P}(s_n) - \mathbf{k}(s_{n+1})\hat{\mathbf{a}}^T(n)\mathbf{P}(s_n)\Big) \\
\epsilon(s_n \rightarrow s_{n+1}) &= \sum_{i=0}^{n} w^{n-i} |y(i) - \hat{\mathbf{a}}^T(n)\hat{\mathbf{h}}^{(n)}(i)|^2 \\
\hat{\mathbf{h}}(s_{n+1}) &= \hat{\mathbf{h}}(s_n) + \mathbf{k}(s_{n+1})\epsilon(s_n \rightarrow s_{n+1}).
\end{aligned} \tag{1.149}$$

This CIR and data estimator provides fast adaptation and renders the estimator applicable to fast fading channels, since in this scenario reliable re-converging after a fading-induced error burst is of prime importance.

The blind data and CIR estimator presented in this section has some significant differences wrt the Bussgang blind equalisers of the previous section. Namely:

- In contrast to stand–alone equalisation, there is no filter simulating the inverse of the CIR
- The convergence is significantly faster and more accurate
- This technique can be combined with channel decoding

As for the other blind equalisation techniques discussed, numerous studies have provided insight into this method. The associated practical considerations and a range of modifications were studied by Chugg and Polydoros in [36], while in [37] the associated performance was studied both theoretically and by simulations. In [38] and [39], Hidden Markov Model (HMM) theory was applied to the problem

of joint data detection as well as CIR estimation and performance results were given for the so–called Gaussian Minimum Shift Keying (GMSK) modulation invoked in a Time Division Multiple Access (TDMA) scenario. In [40] the employment of fuzzy logic was proposed for the calculation of the associated decision metrics. In [41] and [42] the acquisition performance of a PSP detector was evaluated and "smart" initialisation strategies were explored. In [43] the algorithm was adapted to make use of soft statistics and to include error prediction. Finally, in [30] a genetic algorithm was applied for estimating the CIR, and the Viterbi algorithm was then invoked. For more research on blind sequence estimation techniques the reader is also refered to [142–146].

1.3.9.1 Complexity Reduction

As we have already mentioned, when the channel's delay spread is high, most joint data detection and channel estimation techniques become prohibitive in terms of complexity. The same happens in the context of PSP. The complexity of PSP is substantial even for transmissions over low–dispersion channels, especially for high–order QAM schemes, as we will se in Section 1.3.12. In some scenarios, however, a high–delay–spread channel may consist of two or more multipath components that are far from each other. In this case, assuming a higher number of CIR taps is unnecessary. Only the effects of the taps having highest magnitude should be equalised, assuming that all others are zero. In this approach, the complexity is significantly reduced and channels exhibiting a higher dispersion can be equalised using the PSP technique. We refer to this technique as *Reduced–Channel PSP*. The task in this case is that of deciding which of the taps can be eliminated. An adaptive algorithm should be employed, which will decide when the value of a tap is very close to zero and can therefore be eliminated and when the channel estimation is inadequate and more channel estimation taps need to be allocated. Performance results will be presented for the two WATM channels of Figures 2.3(a) and (b) in Chapter 2 using this technique.

Let us now consider the family of second–order statistics based techniques.

1.3.10 Blind equalisation Using Second–Order Cyclostationary Statistics

Using second–order statistics [147], in general, is significantly more efficient, than invoking higher–order statistics. This is because less samples are needed, in order to generate the required statistics, implying higher convergence speed and better stability. Nevertheless, second–order statistics are not applicable in all cases of interest. Specifically, in the case of blind equalisation, the CIR is not identifiable from second–order statistics, when we sample at the Baud–rate, i.e. at one sample per symbol period [7]. In 1991 Tong *et al.* [28] exploited the fact that by employing oversampling, the

received signal becomes so–called *cyclostationary*, implying that the associated statistical properties become periodic in time [147]. Exploiting this cyclostationarity, Tong *et al.* proposed a method of CIR estimation, using second–order statistics. More recently, in 1995 Moulines *et al.* [29] observed that the same happens, when we are taking the input of multiple sensors – such as multiple antennas for example – at Baud–rate, i.e. without resorting to oversampling. Finally, in 1997, Tsatsanis and Giannakis [54], [48] proposed another substitute for oversampling at the receiver, namely inducing cyclostationarity at the transmitter, instead of oversampling at the receiver. All these ideas lead to a range of similar methods applicable to CIR estimation, which are based on second–order statistics.
Let us invoke oversampling and denote the symbol interval by T, while the sampling interval by T_S, so that $T = T_S \cdot \Delta$, where Δ is the oversampling factor. We also assume as in the rest of this chapter that the channel's memory extends from $-L_1T$ to L_2T. The channel memory limits do not have to be multiples of the symbol interval T, it is simply notationally convenient. We observe the system over the time interval $[nT, (n+1) \cdot T]$, using the linear model of:

$$\mathbf{y}(n) = \mathbf{H}(n) \cdot \mathbf{a}(n) + \mathbf{e}(n), \tag{1.150}$$

where

$$\mathbf{y}(n) = [y(nT), y(nT + T_S), \ldots, y(nT + (\Delta - 1)T_S)]^T \tag{1.151}$$

represents the received signal vector at time instant n,

$$\mathbf{H}(n) = \begin{bmatrix} h_{-L_1T}(n) & \ldots & h_{L_2T}(n) \\ h_{-L_1T+T_S}(n) & \ldots & h_{L_2T+T_S}(n) \\ \vdots & \vdots & \vdots \\ h_{-L_1T+(\Delta-1)T_S}(n) & \ldots & h_{L_2T+(\Delta-1)T_S}(n) \end{bmatrix} \tag{1.152}$$

is constituted by the channel's impulse response taps $\{h_i(n)\}$ for $i = -L_1T, \cdots, L_2T + (\Delta - 1)T_S$ at time nT,

$$\mathbf{a}(n) = [a((n + L_1)T), \ldots, a((n - L_2)T)]^T \tag{1.153}$$

is the transmitted data vector and

$$\mathbf{e}(n) = [e(nT), e(nT + T_S), \ldots, e(nT + (\Delta - 1)T_S)]^T \tag{1.154}$$

is the channel noise vector.

We now point out the equivalence of oversampling the received signal with rendering the transmitted signal cyclostationary and also its equivalence to involving the outputs of Δ number of 'sensors'

or antennae. The former [48] involves transmitting the same output Δ times sequentially, instead of just once. This way the transmitted signal becomes cyclostationary and the same happens to the corresponding received signal sampled at the Baud–rate, when the channel is stationary for the duration of processing. If we consider the transmitter's Baud–rate to be Δ–times lower than the receiver's Baud–rate, then the received signal would be the same as in the previous scenario.

Upon using the outputs of multiple sensors or antennae [29], we can observe that the output of each sensor – say for example that of the i–th sensor – is equivalent to the output of oversampling at time instants of $iT_S, i = 0, \cdots, \Delta - 1$ in the $[0, (\Delta - 1)T_S]$ space.

Having shown the above equivalences, we will now proceed to discussing the formation of second–order statistics based algorithms. Let us consider the autocorrelation matrix of the received signal $y(nT + iT_S)$ over the time interval iT at time instant nT:

$$\begin{aligned}
\mathbf{R_{yy}}(i;n) &= E[\mathbf{y}(n) \cdot \mathbf{y}^H(n-i)] \\
&\overset{(1.150)}{=} E[(\mathbf{H}(n) \cdot \mathbf{a}(n) + \mathbf{e}(n)) \cdot \\
&\quad (\mathbf{a}^H(n-i) \cdot \mathbf{H}^H(n-i) + \mathbf{e}^H(n-i))] \\
&= E[\mathbf{H}(n) \left(\mathbf{a}(n)\mathbf{a}^H(n-i)\right) \mathbf{H}^H(n-i) + \\
&\quad + \mathbf{H}(n)\mathbf{a}(n)\mathbf{e}^H(n-i) + \\
&\quad + \mathbf{e}(n)\mathbf{a}^H(n-i)\mathbf{H}^H(n-i) + \\
&\quad + \mathbf{e}(n)\mathbf{e}^H(n-i)], \qquad (1.155)
\end{aligned}$$

where $\mathbf{y}(n)$ was defined in Equation (1.151). The two terms in the middle of this expression have a zero mean value, since they are the product of independent variables. Thus, Equation (1.155) becomes:

$$\mathbf{R_{yy}}(i;n) = \mathbf{H}(n) \cdot \mathbf{R_{aa}}(i;n) \cdot \mathbf{H}^H(n-i) + \mathbf{R_{ee}}(i;n). \qquad (1.156)$$

The CIR estimation process consists of estimating the CIR matrix $\mathbf{H}$ using measurements of the autocorrelation matrix $\mathbf{R_{yy}}(n)$ according to Equations (1.155) and (1.156), assuming that the autocorrelation of the transmitted signal $\mathbf{R_{aa}}(i;k)$ is known. Tong *et al.* [7] invoked the assumption that the input data source produces completely uncorrelated symbols $\mathbf{a}(n)$, while Hua *et al.* [58] adapted the method to work with correlated input signals. Furthermore, while in Tong's algorithm [7] the noise was assumed to have some known properties, in [49] a variation was proposed, which assumed no knowledge concerning the channel noise. In order to estimate the CIR matrix $\mathbf{H}$ in Equation (1.152), we must also have knowledge of the noise's correlation. The noise samples cannot be assumed to be uncorrelated (white), since they are generated at a higher rate, than the Baud–rate and since

the receiver's lowpass filtering imposed autocorrelation upon it. This autocorrelation function has the same shape as the impulse response of the receiver filter, since the noise was white before this filtering and hence its autocorrelation function was a $\delta()$ function.

There are various ways of performing CIR estimation at this stage. Some algorithms, such as that proposed by Tong *et al.*, [7, 28], estimate the CIR by assuming that the noise is white – not a valid assumption in general – and the transmitted sequence is also white. In our forthcoming deliberations the channel is assumed to be static. This is an assumption stipulated, in order to simplify our study. The CIR matrix $\mathbf{H}$ of Equation (1.152) is then estimated by performing the so–called "singular value decomposition (SVD)" [148] of $\mathbf{R_{yy}}(0)$ and $\mathbf{R_{yy}}(1)$ and then by performing a range of further algebraic calculations. In other algorithms [48] iterative procedures have been proposed, exhibiting lower complexity, but having an asymptotically similar performance. After estimating the CIR matrix $\mathbf{H}$ of Equation (1.152) and ignoring the effect of noise in Equation (1.150), we can invert $\mathbf{H}$ in order to extract the original information symbols $\mathbf{a}(n)$ from the received sequence $\mathbf{y}(n)$, using the following LMS estimator:

$$\mathbf{c}_{LMS} = \mathbf{H}^t \cdot \mathbf{y}(n), \tag{1.157}$$

where $\mathbf{H}^t$ satisfies:

$$\mathbf{H}^t \cdot \mathbf{H} = \mathbf{I} \tag{1.158}$$

and it is the "pseudo–inverse" [120] of the CIR matrix. The input signals have been assumed to be i.i.d. variables with a normalised power of unity, while the noise has been neglected.

An important advantage of the second–order statistics based algorithms – compared to the family of the linear equalisers – is that they are asympotically accurate. Simulations presented in [7] demonstrate this accuracy in performance terms and also show that the second–order statistics based algorithms exhibit fast convergence, typically within a few symbols. This concludes our discussions on the second–order statistics based algorithms. For more detailed discussions on this issue the reader is referred to [7, 28] and also to [49–51]. In [52] a different method of estimating the CIR matrix $\mathbf{H}$ of Equation (1.152) is proposed for the "multiple channel" scenario, using so–called "outer–product matrix decomposition". In [53, 54], the transmitter–induced cyclostationarity algorithms are explored, while in [55] the cyclostationarity–based method is applied to an Orthogonal Frequency Division Multiplexing (OFDM) receiver. In [56], a modification of Tong's original method [7] is proposed, while in [57] a general study of the cyclostationary method is given. The effects of non–i.i.d. signal distributions are studied in [58] and in [59]. In [60] the second–order statistics based methods are adapted to the so–called source separation problem, where for example the wanted and interfering signals are separated. In [61] the subspace method is applied to the suppression of both intersymbol interference

and multiple-access interference. In [62] the case of unknown noise distributions is considered. Finally, in [63], the so-called "*Column-Anchored Zeroforcing Equalisation (CAZE)*" is proposed and studied. Having presented an introduction to the family of second-order statistics based techniques, let us now consider a range of so-called "polycepstra-oriented" algorithms in the next section.

1.3.11 Blind Channel Estimation and Equalisation Using Tricepstra

The last class of blind equalisation algorithms discussed here involves fourth-order – rather than second-order – statistics of the received signal, in order to estimate the inverse of the CIR and to equalise its effects. A number of algorithms belonging to this class were proposed by Hatzinakos and Nikias in [5] and also by Mendel in [149]. This algorithm, referred to as the *Tricepstrum Equalisation Algorithm (TEA)*, employs the complex cepstrum of the so-called fourth-order cumulants (the tricepstrum is defined in Appendix F) of the received signal sequence sampled at the Baud-rate. These algorithms are capable of identifying both so-called "minimum and maximum-phase" channels, which we will characterise more explicitly during our further discourse. Below we briefly introduce the TEA using the fourth-order cumulants and tricepstra as defined in Appendix F.

In this context we aim to equalise a channel having a z-domain channel transfer function of $H(z)$. We consider $H(z)$ as the product of a minimum-phase [15] section – where the z-domain transfer function of the channel has all its zeros inside the unit circle – and a maximum-phase section – where the z-domain transfer function of the channel has all its zeros outside the unit circle. This is formulated as [5]:

$$H(z) = A \cdot I(z^{-1}) \cdot O(z), \tag{1.159}$$

where A is a constant and $I(z^{-1})$ and $O(z)$ are polynomials of the form $\Pi_{i=1}^{M}(1 - a_i z^{-1})$ and $\Pi_{j=1}^{K}(1 - a_j z)$ respectively, with $|a_i| < 1$, M is the number of channel zeros inside the unit circle and K is the number of channel zeros outside the unit circle. A simple manifestation of a minimum-phase channel exhibits a CIR, where the main tap associated with the time instant 0 is the largest one. The polynomial $I(z^{-1})$ is a minimum-phase polynomial, while $O(z)$ is a maximum-phase polynomial. In order to equalise this channel under the so-called Zero-Forcing (ZF) constraint – implying that the combined CIR and equaliser impulse response is forced to zero at sampling instants $n \neq 0$ – we use an equaliser, having a transfer function of $C(z)$, which is the inverse of that of the channel [5]:

$$C(z) = \frac{1}{H(z)} = \frac{1}{A \cdot I(z^{-1}) \cdot O(z)}. \tag{1.160}$$

Then the cascaded channel and equaliser transfer function constitutes an ideal channel. If instead of the ZF equaliser we use a decision-feedback equaliser (DFE) having a feedforward transfer function

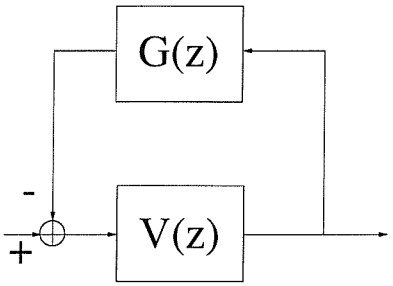


Figure 1.34: The DFE filter structure.

of $V(z)$ and a feedback transfer function of $G(z)$, then the equaliser's transfer function will have the form [5]:

$$C(z) = \frac{V(z)}{1 + G(z)}, \tag{1.161}$$

where $V(z)$ and $G(z)$ are the feedforward and feedback section's transfer function, respectively, as shown in Figure 1.34. The feedback filter must be realisable. A possible choice for $V(z)$ and $G(z)$ is [5]:

$$V(z) = \frac{O^*(z^{-1})}{AO(z)} \tag{1.162}$$

$$1 + G(z) = I(z^{-1}) \cdot O^*(z^{-1}). \tag{1.163}$$

Then the system $C(z) \cdot H(z)$ is perfectly equalised and the feedforward section $V(z)$ of the DFE is an all–pass filter having zeros inside and poles outside the unit circle, while the feedback filter $G(z)$ is a minimum–phase filter. In order to construct the DFE we have to find an estimate of the coefficients of the feedforward and feedback filters. Equivalently, we have to find an estimate of the poles and zeros of the two equaliser filters. This can be achieved using the fourth–order cumulants of the received signal.

We form the received signal $y(n)$ as the convolution of the transmitted signal $a(n)$ with the CIR $\{h_n\}$, plus a zero–mean additive Gaussian stochastic process $e(n)$, as follows:

$$y(n) = a(n) * h_n + e(n). \tag{1.164}$$

We recall from Appendix F that the tricepstrum $c_y(m, n, l)$ of the received signal $y(n)$ is related to the tricepstrum $c_h(m, n, l)$ of the CIR by [5]:

$$c_y(m, n, l) = c_h(m, n, l), \qquad (m, n, l) \neq (0, 0, 0). \tag{1.165}$$

This implies that in the tricepstrum domain the received signal is equal to the CIR. Therefore, esti-

mating the received signal's tricepstrum directly gives us the CIR's tricepstrum. The problem then becomes that of estimating the CIR in terms of its tricepstrum representation. It was shown in [5] that it is sufficient to consider tricepstra in the form of $c_y(K,0,0)$ (K integer). We recall from Appendix F the following relationship:

$$\sum_{I=1}^{p} A^{(I)}[L_y(m-I,n,l) - L_y(m+I,n+I,l+I)] +$$
$$+\sum_{J=1}^{q} B^{(J)}[L_y(m-J,n-J,l-J) - L_y(m+J,n,l)] =$$
$$= -mL_y(m,n,l)$$

where

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

This relationship is in fact a system of linear equations, which can be solved by iterative methods, as shown in Appendix F. However, the estimation of the CIR and the equaliser filters is quite an elaborate task and hence the associated derivation was relegated to Appendix F, following the approach of [5].

An estimate of these algorithms' complexity compared to the Bussgang algorithms' complexity was also given by Hatzinakos and Nikias in [5], where the complexity of the former appears to be significantly higher, a fact which was mentioned before. As a trade–off, the simulations presented in [5] indicate the superiority of the TEAs in terms of convergence speed, as well as in terms of their ability to equalise non–minimum phase channels, which are often encountered in fading mobile channels. For more detailed investigations of these techniques the reader is referred to [5, 97, 150–153].

Having presented an overview of a range of basic blind equaliser structures, we will now provide a summary of their complexity.

1.3.12 Complexity Evaluation

In this section the complexity of the various blind equalisers presented is evaluated. We commence our discussions by considering the complexity of the Bussgang algorithms. The complexity of all the Bussgang techniques is similar and it is relatively low. For simplicity, in this section we will assume that the number of equaliser taps is $2N+1$, i.e. $N_1 = N_2 = N$, according to the notation we have used so far. The equaliser's complexity depends only on the number of equaliser taps, $2N+1$, and it is on the order of N, which is indicated as $O(N)$. An estimate of the number of real additions and multiplications required for each algorithm per equalised symbol interval is presented in Table 1.1. Here, we have assumed that the complex variables are represented in the memory of the associated arithmetic unit

Algorithm	Additions	Multiplications	Memory
CMA [4]	$16L+10$	$16L+13$	$4L+4$
Sato [3]	$16L+8$	$16L+10$	$4L+4$
B–G [32]	$16L+17$	$16L+19$	$4L+4$
Modified–CMA [102]	$16L+8$	$16L+14$	$4L+4$
Stop–and–Go [21]	$16L+8$	$20L+10$	$4L+4$
NCMA [110]	$8L+17$	$8L+23$	$2L+5$
Soft–Constraint [112]	$8L+18$	$8L+20$	$2L+5$
Super–Exponential ($p=2, q=1$) [31]	$O(L^3)$	$O(L^3)$	$O(L)$

Table 1.1: Complexity estimate of the Bussgang techniques of Section 1.3.5, assuming $2N+1 \approx 2L+1$, where L is the channel's memory

	Additions	Mult/ions	Memory
Viterbi	$Q^{2L}(16L+9Q+2)$	$Q^{2L}(16L+8Q+4)$	$Q^{2L}(7+4L)$
M–algorithm	$M(16L+9Q+2)$	$M(16L+8Q+4)$	$M(7+4L)$

Table 1.2: Complexity estimate of the sequence estimation algorithms of Section 1.3.9

in terms of their real and imaginary parts. We have also assumed that the number of equaliser taps, namely $2N+1$, is approximately equal to the channel's memory of $2L+1$, i.e. $2N+1 \approx 2L+1$ and that the square root evaluation required for the computation of $|\epsilon^{DD}(n)|$ in the Benveniste–Goursat algorithm of Section 1.3.5.4 is performed with the aid of 4 real additions and 2 real multiplications. The square root in this case is calculated by the approximate formula of $\sqrt{1+x} \approx 1+\frac{x}{2}-\frac{x^2}{2}+\frac{3 \cdot x^3}{48}$, which is the first four terms of the Taylor expansion of $\sqrt{1+x} \approx \sum_{n=0}^{\infty} \frac{1 \cdot 3 \cdots (2n-3)}{2^n n!} \cdot x^n$, for x close to zero. One could take into account more terms in the series in order to render the error estimation more accurate. Finally, in Table 1.1 we have also included an estimate of the memory requirements of each algorithm. In addition to the Bussgang algorithms the table incorporates the super–exponential algorithm, which does not belong to the family of Bussgang algorithms, but exhibits similarities with the CMA. A complexity estimate of the PSP–based algorithms of Section 1.3.9 is obtained similarly to the previously introduced Bussgang algorithms of Section 1.3.5 by calculating the total number of additions and multiplications as well as the associated memory requirements. Considering that the convolution of the CIR with the estimated sequence in Equation (1.148) is only calculated once for each survivor transition and stored in memory (thus saving unnecessary further calculations), the associated complexity results are summarised in Table 1.2. These complexity figures refer to one symbol interval. The complexity arising from convolutional decoding, if channel coding is used, is ignored in these calculations. In Table 1.2 M is the number of survivors in Figure 1.33 that we retain at each step of the M–algorithm of Section 1.3.9 and Q is the number of possible signal constellation points, i.e. $Q=2^K$ for a K-bit per symbol modulation scheme.

By comparing Tables 1.1 and 1.2 we observe that the complexity of the sequence estimation techniques

(aided by LMS–based CIR estimation) is exponentially increasing with the dispersion or the memory of the channel L, i.e. it is of $O(Q^L)$, when the Viterbi algorithm is used, implying that channels having long CIRs cannot be equalised by this sequence estimation technique. By contrast, the complexity of the Bussgang techniques is only $O(L)$ and it is only linearly increasing. However, the situation changes when the M–algorithm is used instead of the full search algorithm in the context of the sequence estimation based techniques and we see from Table 1.2 that the complexity also becomes $O(L)$, except that in this case we also have a multiplicative factor of M. Note furthermore that the complexity also depends on the number of phasors in the modulation constellation according to the exponential relationship of Q^{2L}, where Q is the number of phasors in the QAM constellation. Again, from these tables we conclude that the Bussgang algorithms are attractive for long CIRs, since in this case the sequence estimation techniques exhibit an excessively high computational complexity. By contrast, for short CIRs the sequence estimation techniques offer significant performance advantages at an affordable complexity, as it will become explicit in the next chapter. However, as will be seen in Chapter 2, channels exhibiting very long CIRs cannot be equalised using Bussgang equalisers either due to the excessive enhancement of convolutional noise associated with the large number of equaliser taps required for equalisation.

Finally, the normalised linear algorithms of Sections 1.3.6 exhibit a similar complexity and properties to the Bussgang equalisers, apart from the super–exponential algorithm, which is far more complex as a result of its $O(L^3)$ dependence on L caused by the calculation of the cumulants in Equation (1.102).

Having presented a rudimentary overview of various blind equalisation methods, we note the emergence of a recent approach, based on Neural Networks (NN). For more details concerning NN–based blind methods the reader is referred to [24, 154–158]. Another approach which is not considered in this thesis is the *Eigen Vector Algorithm* by Jelonnek, Boss and Kammeyer [159].

1.4 Summary

The family of basic blind equaliser techniques which were presented in this section appear promising in terms of mitigating the channel–induced dispersion. The type of best–suited blind equaliser depends on the specific application concerned. A PSP channel estimator would be ideal for a CIR having only a few low–delay taps in conjunction with low–order modulation schemes. However, when the number of CIR taps increases, this technique becomes prohibitive in terms of its computational complexity. In this case the M–algorithm based approximation is capable of reducing the complexity of the algorithm at the cost of a degraded performance. The severity of performance degradation will depend on factors, such as the shape of the CIR, the specific QAM scheme used, the noise and the number of retained

states M. Since the available processing power offered by the state–of–the art processors increases, the affordable complexity of the channel equaliser also increases. Nonetheless, long CIRs require an exponentially increased equaliser complexity and the M–algorithm based approximation will always suffer from the reduced number of states.

On the other hand, the Bussgang techniques are adequate for channels exhibiting moderate or high dispersion, since their complexity is only linearly increased upon increasing the number of equaliser taps. Their performance, however, is not as high as that obtained by the PSP based equalisers in terms of their achievable speed of convergence and in terms of their convergence robustness in the presence of channel noise and ISI. The undesirable local minima, which have been found to exist in such equalisers, appear to be only of theoretical importance, when the equaliser has an adequate number of taps. Moreover, convergence detection techniques have been proposed and reinitialisation strategies have been suggested, in order to ensure that this issue will not affect the equaliser's performance. High–ISI scenarios are of particular importance in mobile communications applications. However, as complexity is also an issue in conjunction with light–weight portable terminals, these algorithms are amenable to such applications.

Finally, the algorithms, which explicitly use statistical estimations are quite complex and, even though they seem to be asymptotically convergent, they were only briefly touched upon in this treatise. On the other hand, blind neural network algorithms were investigated by the authors, but they were found to exhibit convergence to undesired local solutions, when the initialisation was not in the vicinity of the desired point of equilibrium and therefore they were not researched further.
Having characterised a range of blind equaliser techniques, in the next chapter we will provide performance results for some of these equalisers.

Chapter 2

Performance Results

In this section we present the associated comparative performance results for the algorithms described in Sections 1.3.5 and 1.3.9. Two different types of results are presented, commencing with Bit Error Rate (BER) learning curves, which offer a measure of the algorithms' convergence speed. The second set of results is concerned with the average BER curves of the algorithms over a dispersive Gaussian channel. The modulation schemes involved are 16–QAM and 64–QAM [9]. The associated signal constellations are shown in Figures 2.1 and 2.2.

2.1 Channel Models

Three different channels were used in our comparative study. The first one is a typical worst–case Wireless Asynchronous Transfer Mode (WATM) channel, while the second one is a Shortened WATM (SWATM) CIR, both of which are presented in Figure 2.3. These indoor CIRs were generated with the aid of finding the line–of–sight path and the four longest–delay paths in a 100x100x3m^3 hall at a WATM transmission rate of 155 Mbit/s [9]. The third channel used in our comparative study is a simple one–symbol–delay channel, which is shown in Figure 2.4. This channel may characterise a satellite link, where due to the directional parabolic antenna used, only one or two multipath components may arrive at the receiver. The difference between the WATM, SWATM and the one–symbol–delay channel is that while the former two channels exhibit multipath components at several symbols' delay, the latter exhibits only one additional multipath component at a delay of one symbol. These different CIRs will be used for demonstrating the fact that different equalisers may be appropriate for different channels. All CIRs are assumed to be real, having no imaginary part.

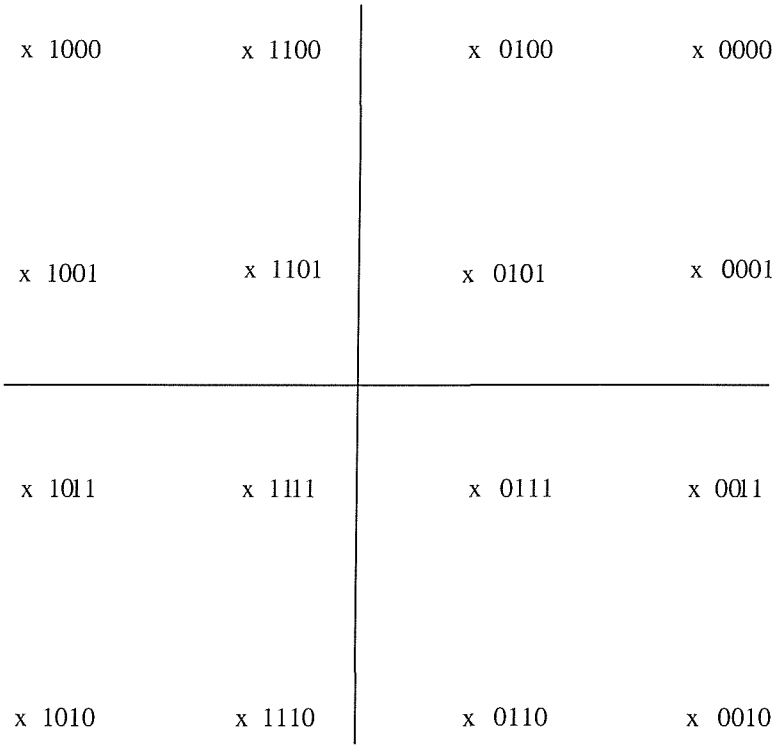


Figure 2.1: 16–QAM constellation

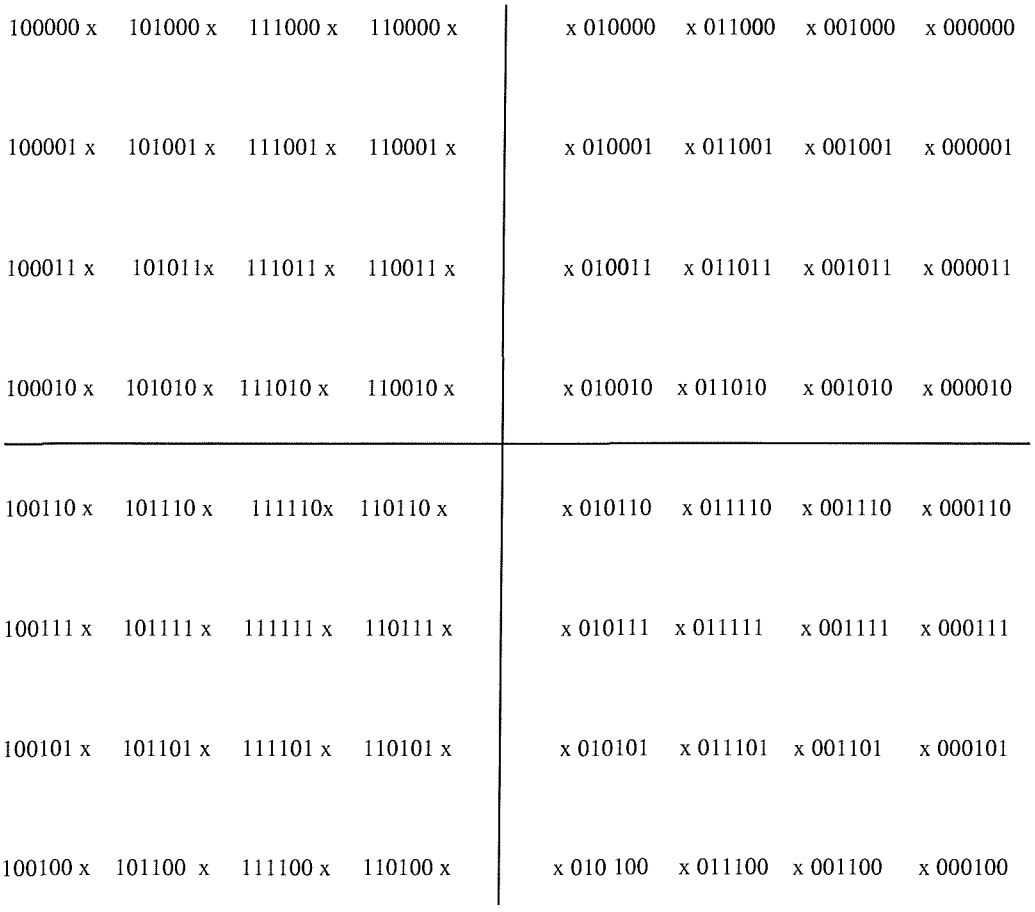
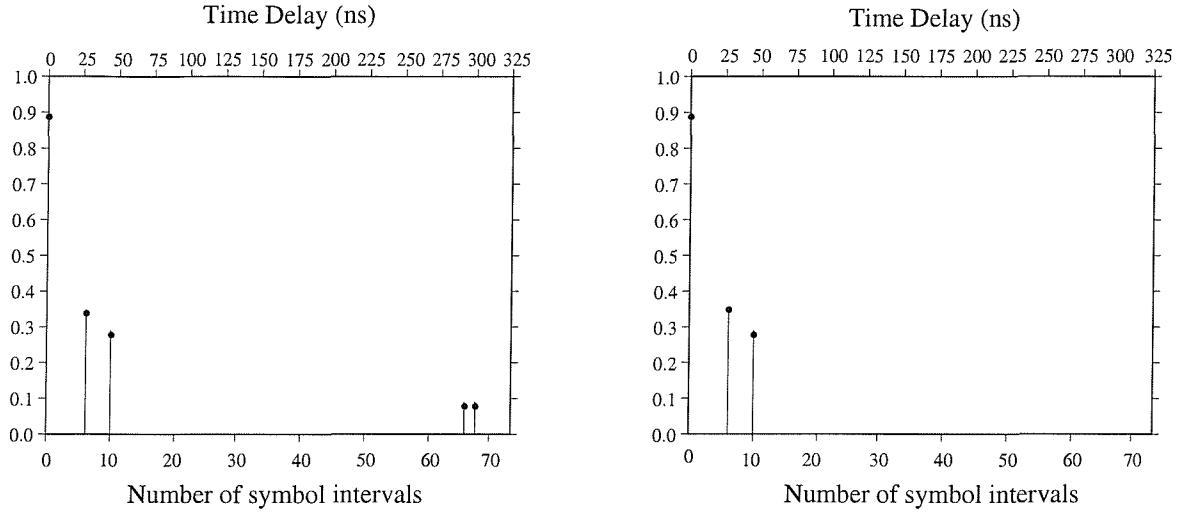


Figure 2.2: 64–QAM constellation



The WATM channel

The Shortened-WATM channel

Figure 2.3: (a) The WATM and (b) Shortened WATM channels

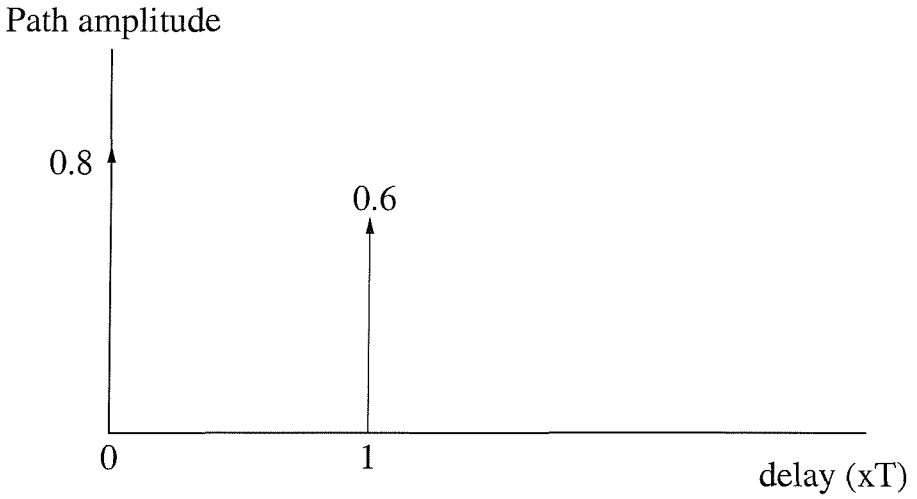


Figure 2.4: The one–symbol–delay channel used in the simulations

2.2 Learning Curves

In this section, the associated BER learning curves of the blind equalisers of Sections 1.3.5 and 1.3.9 are presented for the CIRs of Figures 2.3 and 2.4. The step–size parameter λ is common for all the Bussgang algorithms, which was chosen to be $5 \cdot 10^{-4}$, while the Benveniste–Goursat parameters of Equation (1.88) are $k_1 = 1$, $k_2 = 5$. For the PSP–based sequence estimation algorithm, the step–size in Equation (1.148) is 10^{-2} and only M=Q survivors were retained, where Q is the number of symbols in the QAM constellation. The WATM channel simulations associated with the CIR of Figure 2.3(a) are presented in Figure 2.5 for 16–QAM and in Figure 2.8 for 64–QAM. An equaliser length of 68 taps was employed and the Signal–to–Noise ratio (SNR) was $30dB$. The corresponding SWATM and one–symbol–delay CIR based results are presented in Figures 2.6 and 2.7 as well as in 2.9 and 2.10 for 16–QAM as well as 64–QAM, respectively. As we see, the CIR spread was gradually shortened from 68 to 10 and then to 2. Observe that for the M–algorithm of Figure 1.33 we only presented BER results in Figures 2.7 and 2.10 over the shortest one–symbol–delay CIR. In order to quantify the BER associated with the learning curves, we have averaged the values of BER recorded for numerous 1000–symbol intervals, which was necessary due to the data-dependent performance of the blind equalisers of this type, which typically resulted in a different performance for different 1000-symbol runs. This was a consequence of the fact that not all of the incoming data symbols drive the equaliser to the point of convergence.

From these curves we can infer a range of observations concerning the convergence speed of each of the tested algorithms.

- The M–Algorithm converges at a higher speed, than any of the Bussgang algorithms considered. A reason for this is the employment of a larger step–size value, which could not have been used for the Bussgang algorithms, since this would result in poor tracking performance.

- Sato's algorithm converges at a medium to slow speed.

- Godard's algorithm converges with about the same speed as Sato's and its convergence is faster for higher–order QAM.

- The MCMA algorithm converges at a medium speed, but its convergence is faster for higher–order QAM, as we can see from Figures 2.8 and 2.9 for 64–QAM.

- The Benveniste–Goursat algorithm converges rapidly only for low–order QAM, and we can see from Figures 2.5 and 2.6 that for 16–QAM it converges faster, than any of the other Bussgang algorithms. For higher–order constellations, such as 64–QAM, it converges significantly slower.

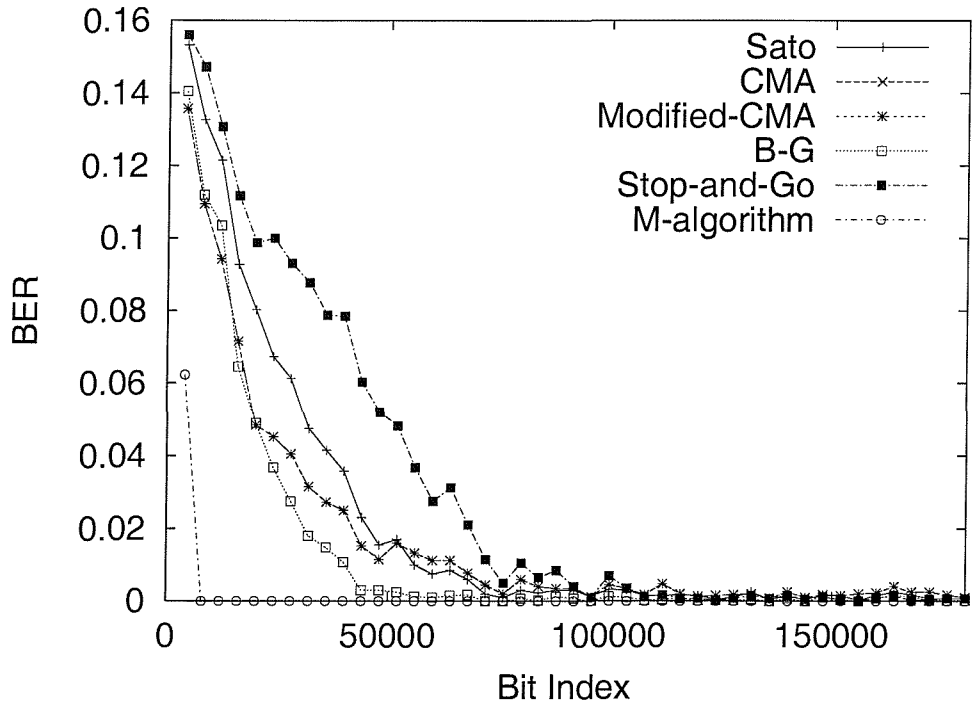


Figure 2.5: BER learning curves over the WATM channel of Figure 2.3(a) for 16–QAM at SNR=30dB using a 68–tap equaliser obeying the schematic of Figure 1.12

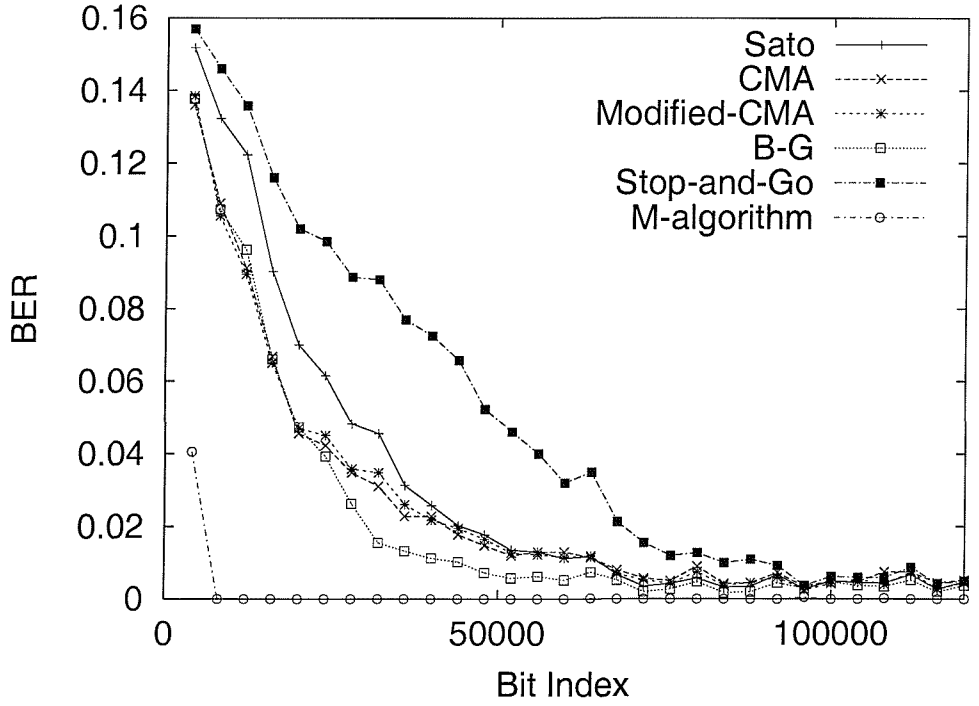


Figure 2.6: BER learning curves over the Shortened WATM channel of Figure 2.3(b) for 16–QAM at SNR=20dB using a 30–tap equaliser obeying the schematic of Figure 1.12

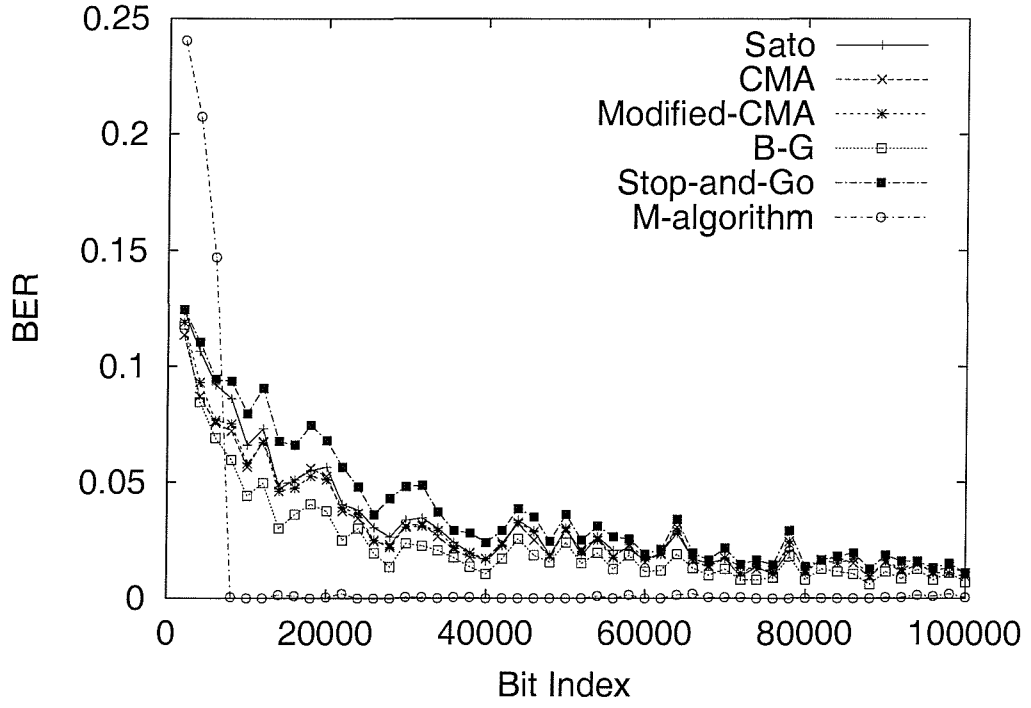


Figure 2.7: BER learning curves over the one–symbol–delay channel of Figure 2.4 for 16–QAM at SNR=20dB using a 10–tap equaliser obeying the schematic of Figure 1.12

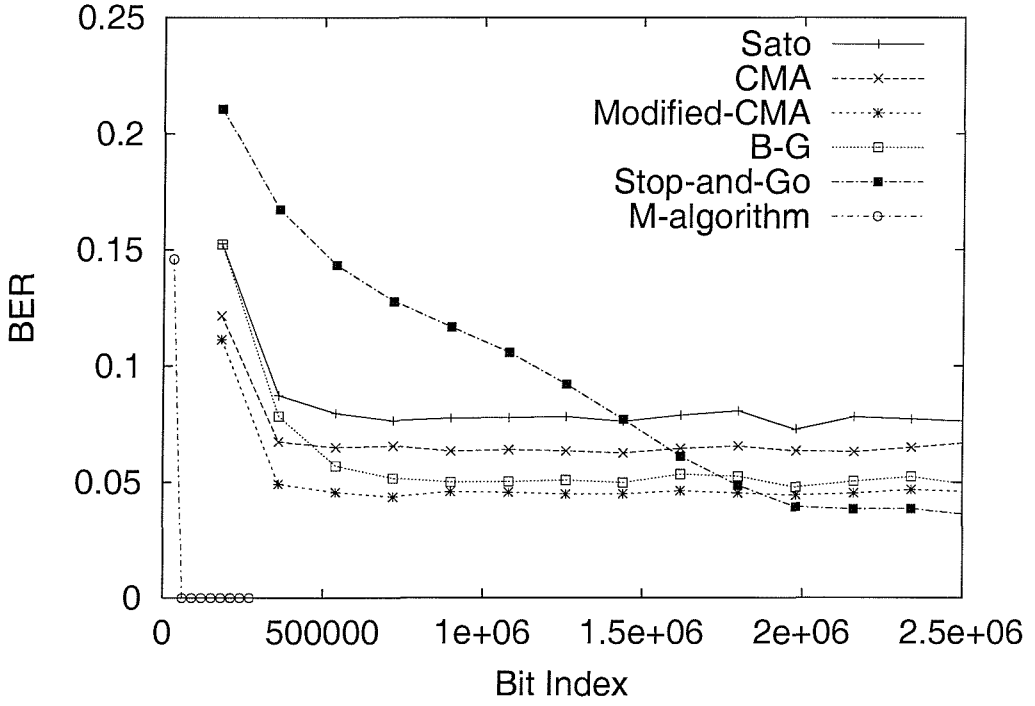


Figure 2.8: BER learning curves over the WATM channel of Figure 2.3(a) for 64–QAM at SNR=30dB using a 68–tap equaliser obeying the schematic of Figure 1.12

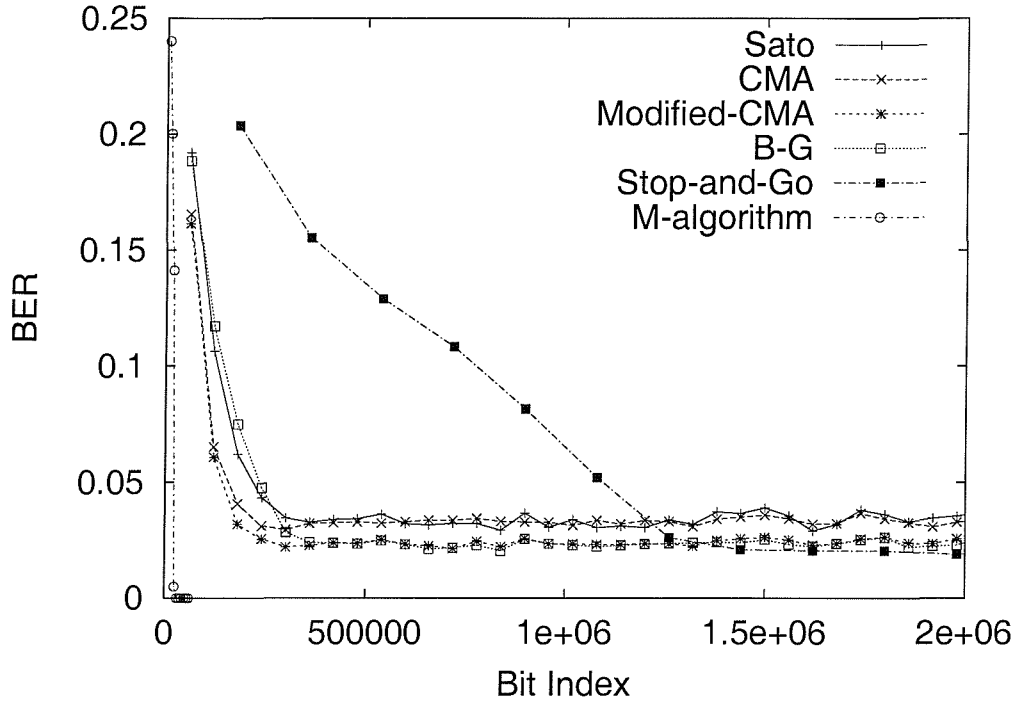


Figure 2.9: BER learning curves over the Shortened WATM channel of Figure 2.3(b) for 64–QAM at SNR=20dB using a 30–tap equaliser obeying the schematic of Figure 1.12

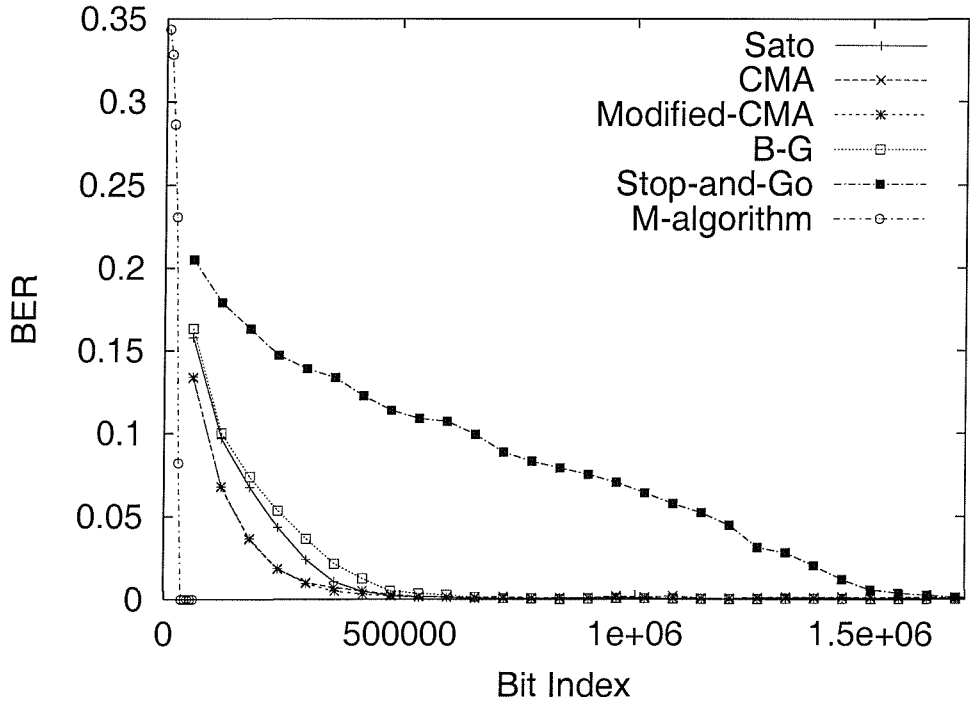


Figure 2.10: BER learning curves over the one–symbol–delay channel of Figure 2.4 for 64–QAM at SNR=30dB using a 10–tap equaliser obeying the schematic of Figure 1.12

	WATM	S–WATM	TC
M–Algorithm	550	550	3700
Sato	30000	25000	80000
CMA	35000	40000	55000
MCMA	35000	20000	60000
Benveniste–Goursat	20000	15000	-
Stop–and–Go	75000	25000	-

Table 2.1: The number of symbols required for each algorithm to converge in the context of 16–QAM transmissions over the WATM and Shortened WATM channels of Figures 2.3(a) and (b) as well as over the simple one–symbol–delay channel of Figure 2.4, which is labelled as TC (Test Channel). The convergence is detected by estimating the slope of the BER. When this slope becomes zero or positive, then convergence is assumed to have been reached. The SNR is $30dB$.

	WATM	S–WATM	TC
M–Algorithm	4150	3000	6000
Sato	50000	50000	120000
CMA	55000	35000	110000
MCMA	45000	45000	115000
Benveniste–Goursat	75000	55000	150000
Stop–and–Go	320000	200000	600000

Table 2.2: The number of symbols required for each algorithm to converge in the context of 64–QAM for transmissions over the WATM and Shortened WATM channels of Figures 2.3(a) and (b) as well as over the simple one–symbol–delay channel of Figure 2.4, which is labelled as Test Channel (TC). The convergence is detected by estimating the slope of the BER. When this slope becomes zero or positive, then convergence is assumed to have been reached. The SNR is $30dB$.

- The Stop–and–Go algorithm is definitely the slowest algorithm in all cases, since it is not always able to iterate.

In Tables 2.1 and 2.2 an estimate of the number of symbols needed for the convergence of each algorithm is given. The converged state was defined as the state, where the BER has 'just' reached its steady state value and does not change significantly thereafter. The channel SNR is kept at $30dB$.

Having presented a comparative simulation study of the convergence speed of the blind equalisers of Sections 1.3.5 and 1.3.9, we will now give an illustration of the blind equalisers' convergence by means of the associated phasor diagrams [9].

2.3 Phasor Diagrams

In this section some phasor diagrams are presented for the various algorithms considered. We are observing the phasor diagram at the equaliser's output at different stages of convergence, each giving an idea of how the equaliser is converging. In Figures 2.11 and 2.12, the phasor constellation is

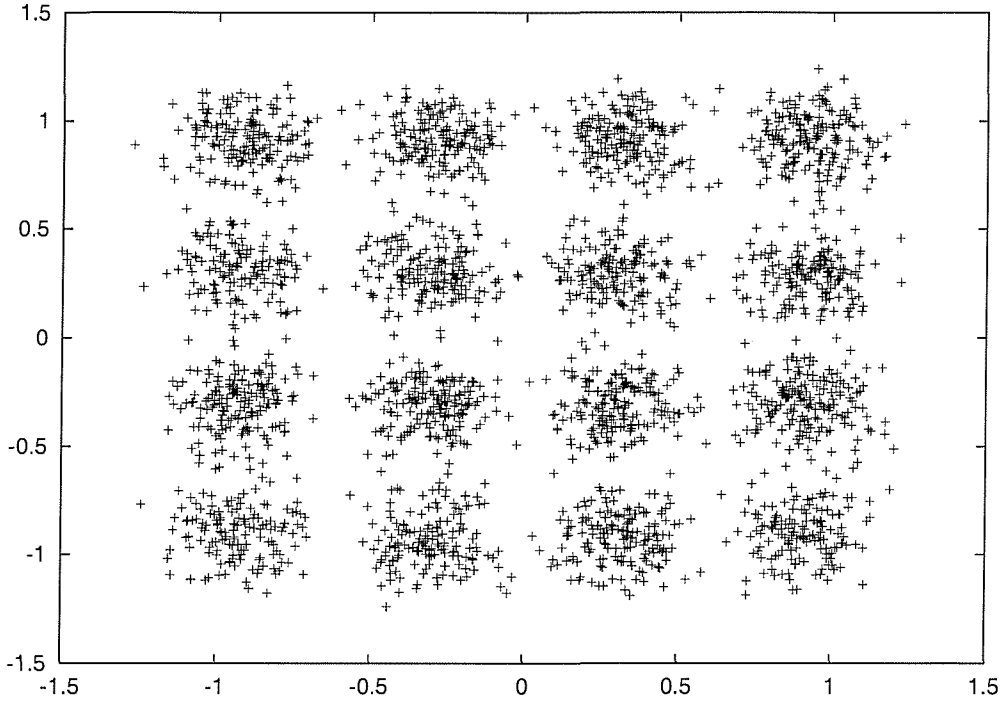


Figure 2.11: The 16–QAM phasor constellation over the one–symbol–delay channel of Figure 2.4 after 40000 symbols at an SNR=30dB using a CMA 10–tap equaliser obeying the schematic of Figure 1.12

plotted in the complex plane for a CMA–based equaliser having 10 taps, equalising the one–symbol–delay channel of Figure 2.4, when 16–QAM is used. Two snapshots are shown. In the first the equaliser is reaching convergence, but still exhibits residual ISI. In the second snapshot, the equaliser has almost converged. These snapshots demonstrate how the equaliser is adapting and slowly approaching the state of convergence. When it has converged, the residual impairments are the convolutional noise, which is rather small, and the additive channel noise, which has also been chosen to be low, so that the convergence can be better observed. At this state, the equalised signal is confined to small regions around the legitimate QAM constellation points and the diameter of these regions depends basically on the SNR and on the residual ISI, which produces convolutional noise. Finally, in Figures 2.13 and 2.14 the phasor diagram is shown for the case of 64–QAM. Having presented our comparative results for the convergence speed of the blind equalisers, we will now investigate the accuracy of convergence.

2.4 Performance over Gaussian Channels

In this section the steady–state average BER curves are presented as a function of the bit–SNR over the WATM and Shortened WATM channels of Figure 2.3, as well as over the one–symbol delay channel of Figure 2.4 using 16–QAM and 64–QAM. The bit–SNR is defined as the SNR per bit, i.e. the signal

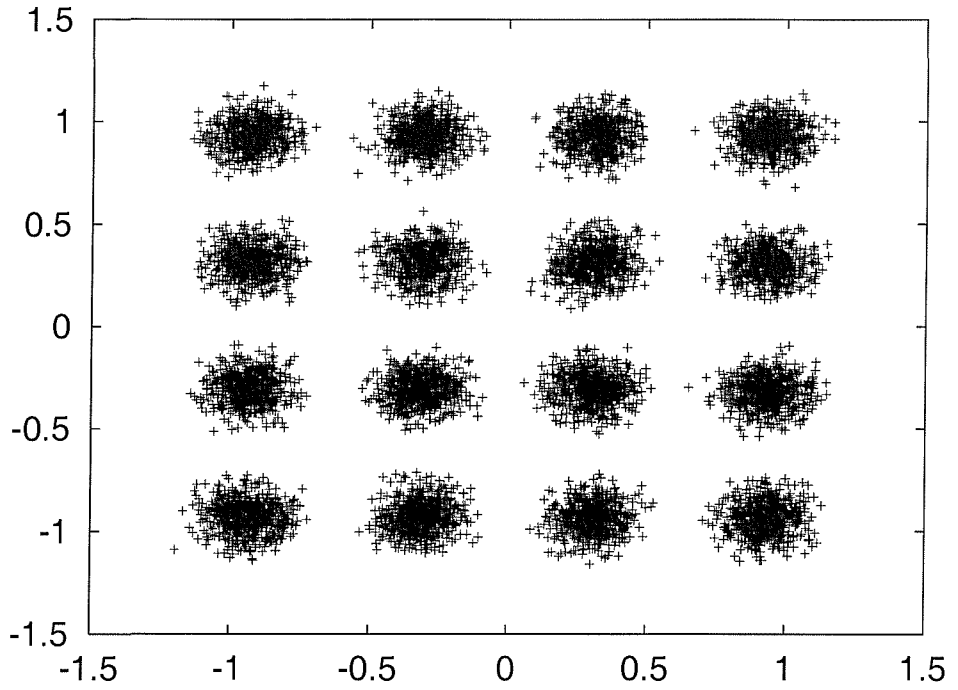


Figure 2.12: The 16–QAM phasor constellation over the one–symbol–delay channel of Figure 2.4 after 60000 symbols at an SNR=30dB using a CMA 10–tap equaliser obeying the schematic of Figure 1.12

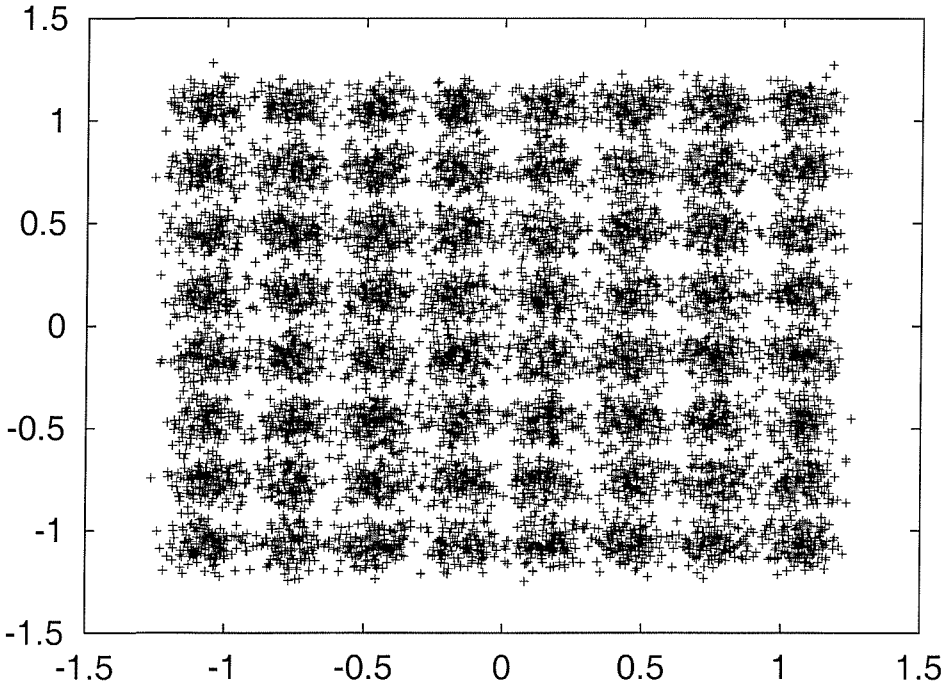


Figure 2.13: The 64–QAM phasor constellation over the one–symbol–delay channel of Figure 2.4 after 70000 symbols at SNR=35dB using a CMA 10–tap equaliser obeying the schematic of Figure 1.12

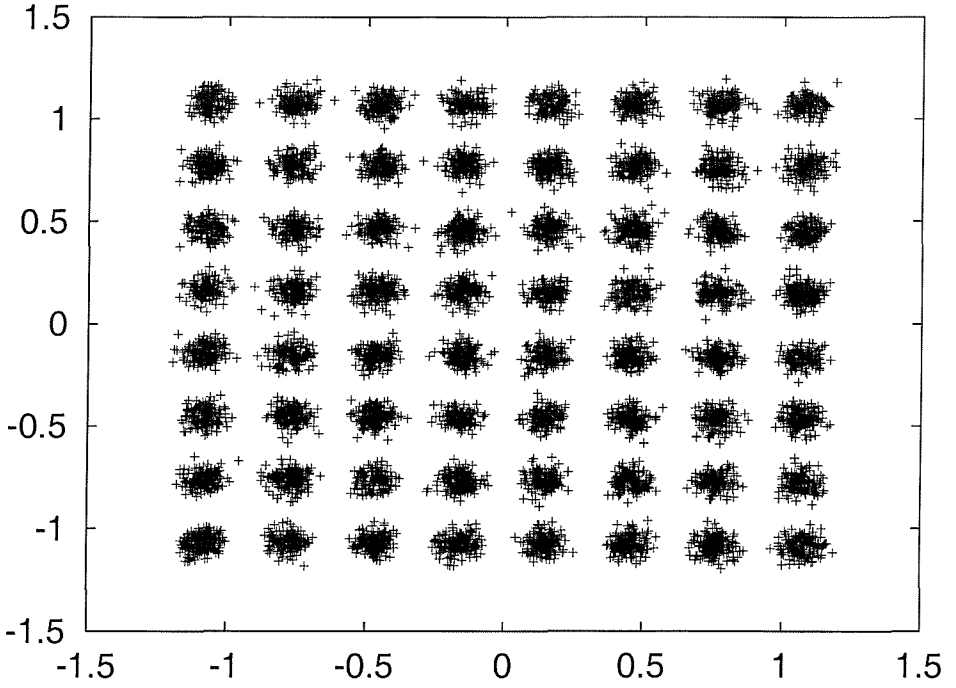


Figure 2.14: The 64–QAM phasor constellation over the one–symbol–delay channel of Figure 2.4 after 180000 symbols at an SNR=35dB using a CMA 10–tap equaliser obeying the schematic of Figure 1.12

power per bit over the noise power per bit. In mathematical terms we have:

$$\text{Bit–SNR} = \frac{SNR}{\text{Number of bits per symbol}}. \tag{2.1}$$

The equaliser characteristics are the same as in Section 2.2. For the M–algorithm, the channel estimator takes into consideration only the CIR taps that indeed exist using the 5–tap CIR of Figure 2.3(a), thus reducing the number of calculations. The associated curves are presented in Figures 2.15 to 2.20.

Note here that following the classical initialisation strategy – that is initialising the equaliser tap vector to $\mathbf{c}^o = (1.2, 0, \cdots, 0)^T$ as suggested by Godard [4] – is inadequate for the Stop–and–Go equaliser to converge to the desired equilibrium. Instead, the equaliser converges to an undesirable equilibrium associated with the combined CIR plus equaliser tap vector of $\mathbf{t}^1 = (0.74, 0.48, 0, \cdots, 0)^T$. For the equaliser to converge to the correct equilibrium, the initialisation has to be closer to the desired equilibrium. For example, initialising the equaliser with the tap vector $(1.2, -0.9, 0.6, 0, \cdots, 0)^T$ is adequate. This phenomenon is directly related to the nature of this algorithm and the undesirable equilibrium is clearly an algorithm–dependent equilibrium. This is why this phenomenon is not observed in the other equalisers. From these curves we can infer some observations concerning the accuracy of convergence for each of the tested algorithms, which exhibits itself in terms of the residual BER at a given SNR

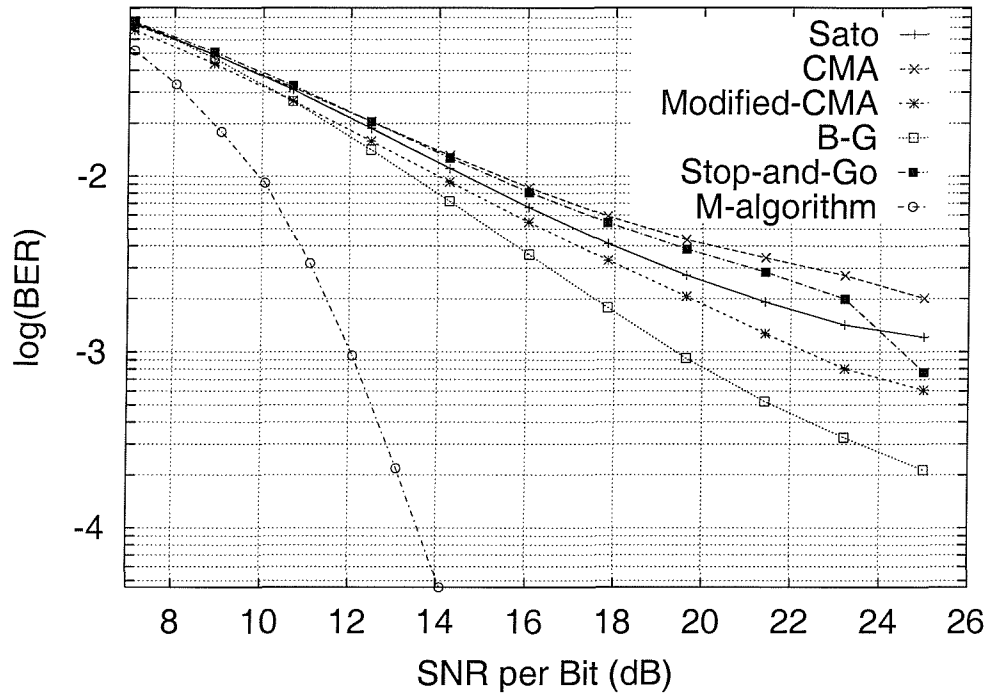


Figure 2.15: Gaussian BER versus SNR curves for the WATM channel of Figure 2.3(a) for 16–QAM using a 68–tap equaliser obeying the schematic of Figure 1.12 and for the M–Algorithm obeying the trellis of Figure 1.33, using the 5–path CIR of Figure 2.3(a).

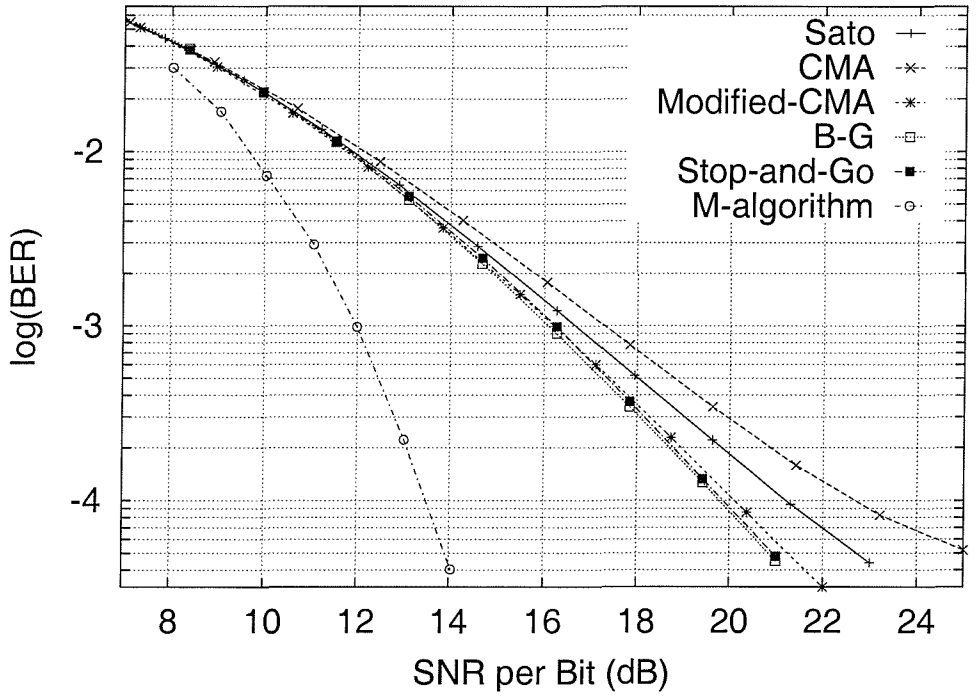


Figure 2.16: Gaussian BER versus SNR curves for the Shortened WATM channel of Figure 2.3(b) for 16–QAM using a 30–tap equaliser obeying the schematic of Figure 1.12 and for the M–Algorithm obeying the trellis of Figure 1.33 using the 3–path CIR of Figure 2.3(b).

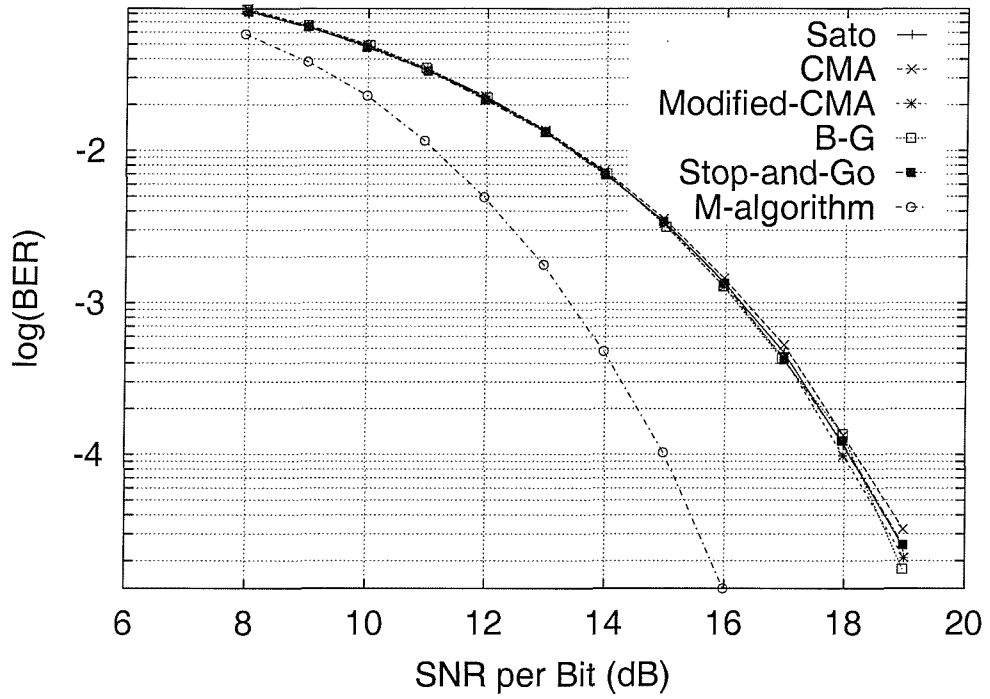


Figure 2.17: Gaussian BER versus SNR curves for the one–symbol–delay channel of Figure 2.4 for 16–QAM using a 10–tap equaliser, obeying the schematic of Figure 1.12 and for the M–Algorithm obeying the trellis of Figure 1.33 using the 2–path CIR of Figure 2.4.

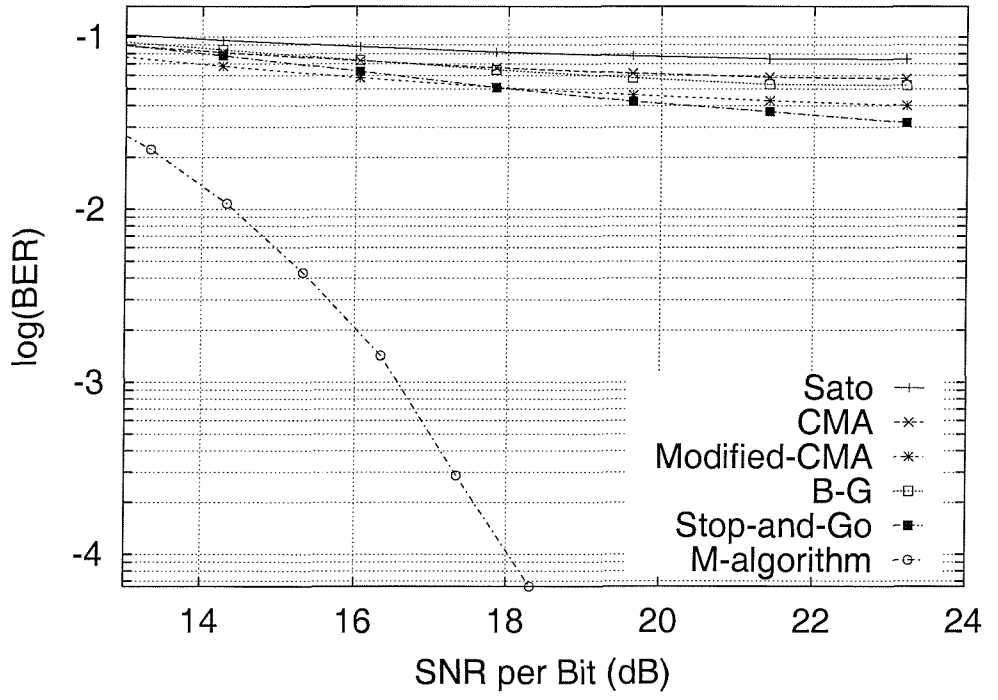


Figure 2.18: Gaussian BER versus SNR curves for the WATM channel of Figure 2.3(a) for 64–QAM using a 68–tap equaliser obeying the schematic of Figure 1.12 and for the M–Algorithm obeying the trellis of Figure 1.33 using the 5–path CIR of Figure 2.3(a).

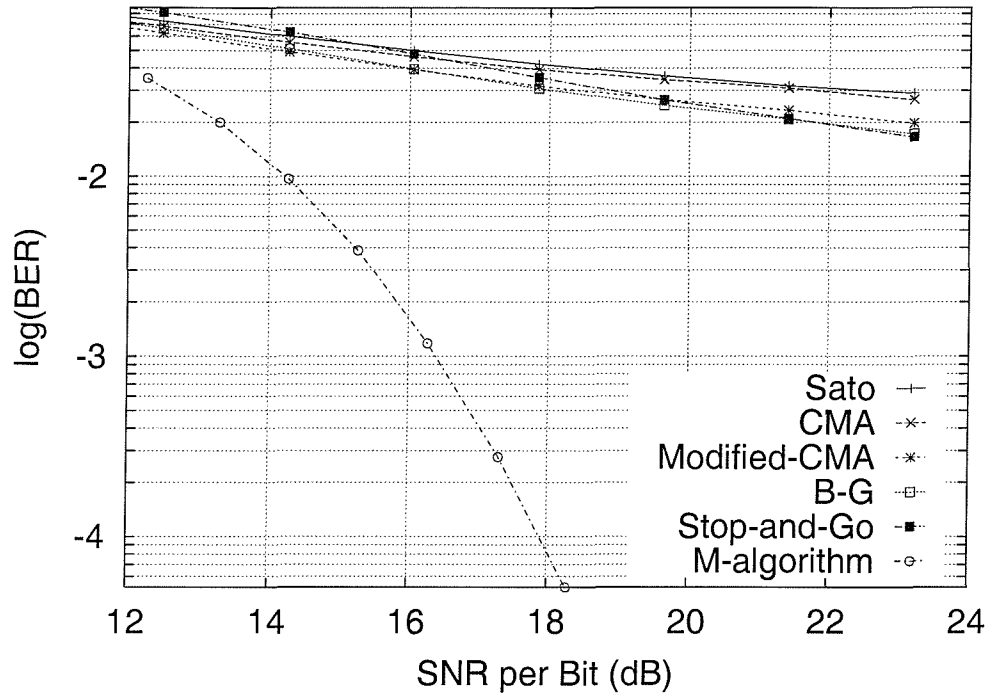


Figure 2.19: Gaussian BER versus SNR curves for the Shortened WATM channel of Figure 2.3(b) for 64–QAM using a 30–tap equaliser obeying the schematic of Figure 1.12 and for the M–Algorithm obeying the trellis of Figure 1.33 using the 3–path CIR of Figure 2.3(b).

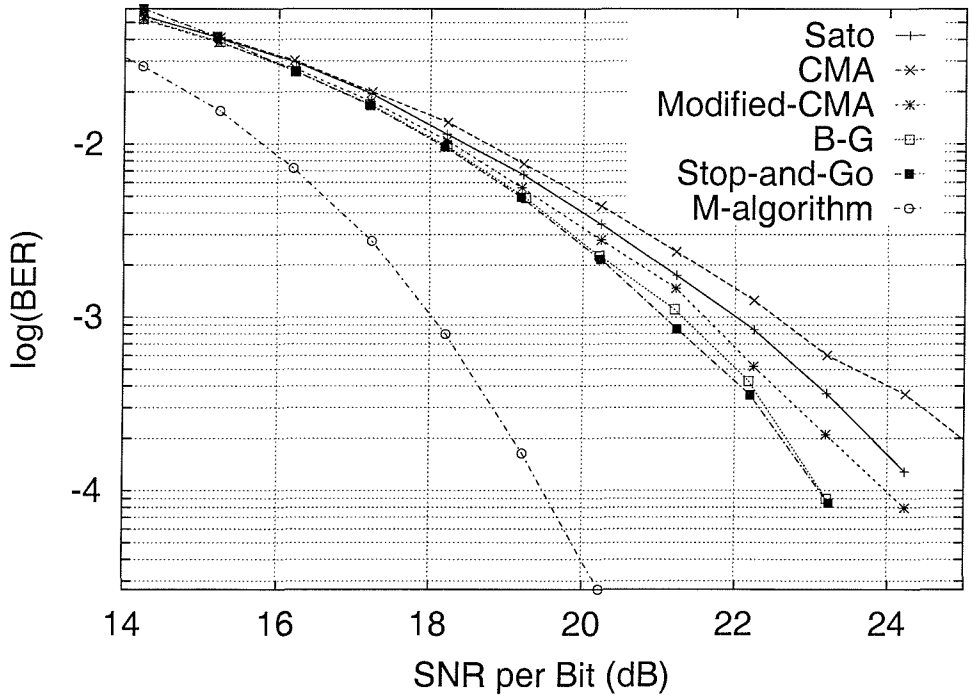


Figure 2.20: Gaussian BER versus SNR curves for the one–symbol–delay channel of Figure 2.4 for 64–QAM using a 10–tap equaliser, obeying the schematic of Figure 1.12 and and for the M–Algorithm obeying the trellis of Figure 1.33 using the 2–path CIR of Figure 2.4.

after reaching the steady–state. We note furthermore that at the SNR concerned, namely at $30dB$, the converged–state–accuracy of the various techniques becomes explicit also from Figures 2.5 to 2.9.

- The M–algorithm provides the best performance.
- Sato's algorithm converges with a medium accuracy.
- Godard's algorithm converges with a medium convergence accuracy, which is improved for low–order QAM.
- The Modified–CMA algorithm offers a good accuracy, which is even better for higher–order QAM.
- The Benveniste–Goursat algorithm exhibits excellent accuracy, especially in conjunction with higher–order QAM.
- The Stop–and–Go algorithm has a good convergence accuracy, especially in the context of BPSK.
- While for 16–QAM the BER curves do not tend to exhibit residual errors, tending toward BER= 0 for high SNRs, the same does not hold for 64–QAM over the WATM channel. This is, because these channels contain multipath components spread to several symbols' delays. This, in turn, means that the equaliser should also have a high number of taps. However, a blind equaliser having a high number of taps is usually not feasible due to their limited accuracy. We can extend the order of these equalisers and improve the associated BER, but only up to the point, where the equaliser starts to enhance the ISI and the associated convolutional noise.
- The WATM channel is a channel containing multipath components scattered over a delay of 67 symbols. The equalisation of this channel would involve a very long equaliser of about 80 taps. In the blind scenario, this is impractical, since the equaliser's resolution is not sufficiently high. Hence beyond a certain length, the taps of these equalisers which are located far from the center tap do not effectively contribute to the equalisation process. Furthermore, they enhance the convolutional noise by increasing the ISI. Besides that, the speed of convergence is linearly dependent on the equaliser order and it becomes low for such high–order equalisers. This can be viewed in the Figure 2.21, where the MSE is plotted against the equaliser order for a specific example. In this case, the algorithms used were the CMA and the LMS or Wiener filter, which is a trained benchmarker used for comparison. The modulation was 16–QAM and the environment was noiseless.

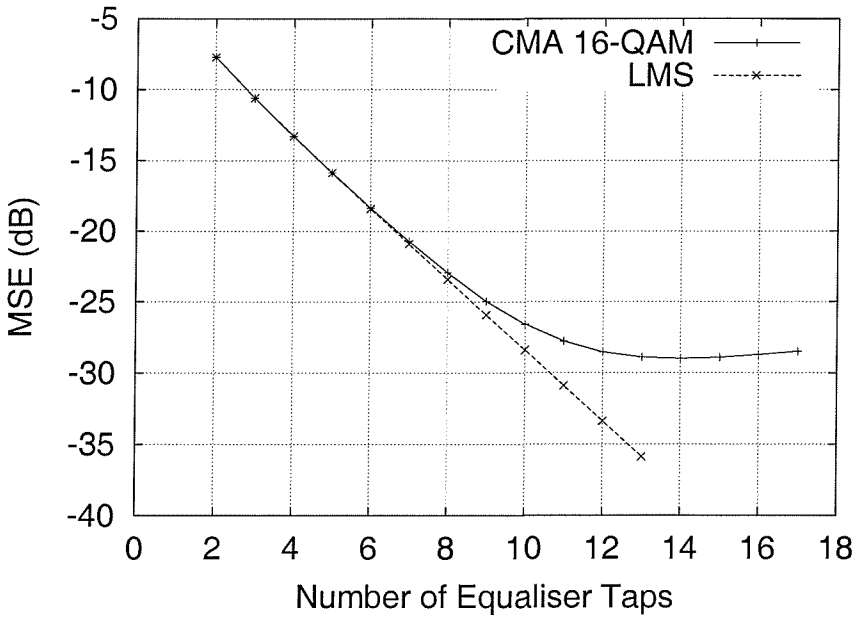


Figure 2.21: MSE as a function of the equaliser order for 16–QAM using the CMA–based equaliser and a noiseless scenario. The channel model used is the one–symbol delay channel as given in Figure 2.4. The LMS–based MSE is given as a lower limit, as this is a non–blind equaliser which gives a benchmark performance. Clearly, for small number of taps the blind equaliser curve coincides with the LMS benchmark curve, while for larger number of taps the CMA produces large MSE, attaining a minimum at about 14 taps, which is the ideal setting for this scenario.

2.5 Simulations with Decision–Directed Switching

In this section, we briefly explore the possibility of switching to decision–directed equalisation after the convergence of the blind equaliser. It is expected that for low SNR values this would drive the equaliser away from convergence – since the blind decision–directed equaliser is generally unstable – and therefore, at low SNRs switching to DD mode would be disastrous. Nevertheless, when the SNR is sufficiently high, switching to DD mode is expected to assist the equaliser in converging with a better accuracy. This is indeed what we observed in our investigations. Explicitly, this technique improved the performance of blind equalisation and the improvement was higher, when the order of the QAM constellation was higher. This can be explained by the fact that when the SNR is high, the DD technique is powerful. This property can be exploited more readily in the context of higher–order QAM constellations, than in lower–order schemes, since the SNR is typically higher for the higher–order QAM schemes, where the distances between constellation points are smaller. In Figures 2.22 and 2.23 the associated improvement is shown for the case of the CMA and for 16–QAM as well as for 64–QAM, respectively. It can be observed that the improvement is modest for 16–QAM, but it is around $1dB$ for 64–QAM at a BER of 10^{-3}. Having studied the range of blind equalisation solutions in the previous subsections, a prominent application of the algorithms in the context of the

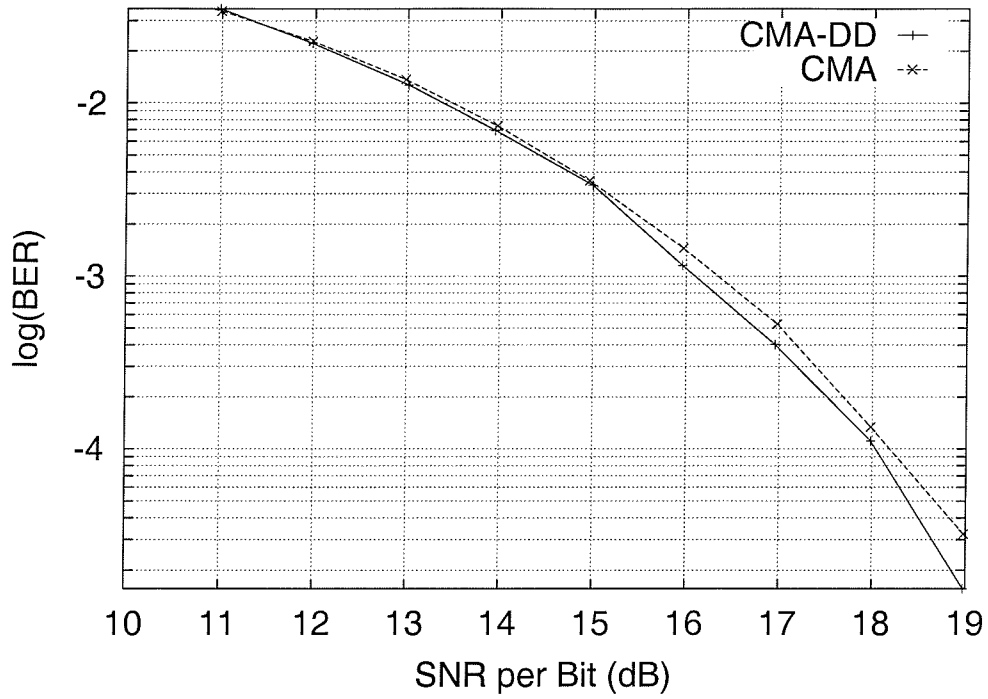


Figure 2.22: Gaussian BER versus SNR curves for the one–symbol–delay channel of Figure 2.4 for 16–QAM using a 10–tap equaliser obeying the schematic of Figure 1.12 for the CMA using switching to decision–directed equalisation after convergence

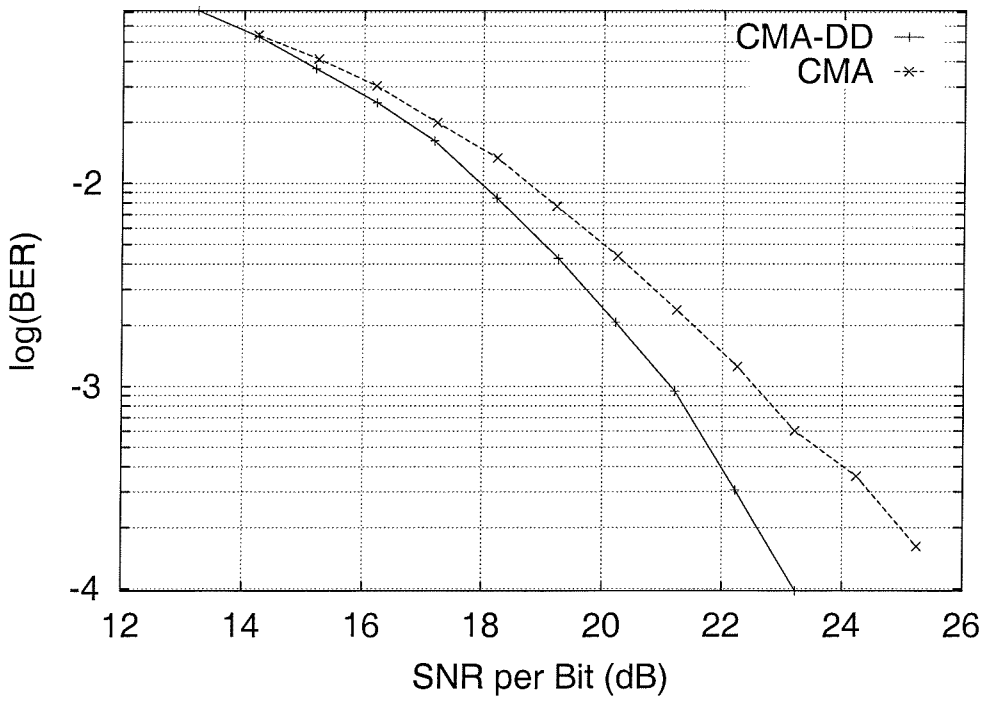


Figure 2.23: Gaussian BER versus SNR curves for the one–symbol–delay channel of Figure 2.4 for 64–QAM using a 10–tap equaliser obeying the schematic of Figure 1.12 for the CMA using switching to decision–directed equalisation after convergence

Pan–European Satellite–based Digital Video Broadcast (DVB–S) [18] system [10] will be presented in the next section.

2.6 Application to Digital Video Broadcasting[1]

2.6.1 Introduction

In recent years three harmonised Digital Video Broadcasting (DVB) standards have emerged in Europe for terrestrial [19], cable–based [20] and satellite–oriented [18] delivery of DVB signals. The dispersive wireless propagation environment of the terrestrial system requires concatenated Reed–Solomon [130, 160] (RS) and rate compatible punctured convolutional coding [130, 160] (RCPCC) combined with Orthogonal Frequency Division Multiplexing (OFDM) based modulation [161]. The satellite–based system employs the same concatenated channel coding arrangement, as the terrestrial scheme, while the cable–based system refrains from using concatenated channel coding, opting for RS coding only. Both of the latter schemes employ, furthermore, blind–equalised multi–level modems [161]. Lastly, the video codec used in all three systems is the Motion Pictures Expert Group's MPEG–2 codec. These standardisation activities were followed by a variety of system performance studies in the open literature [162–165]. Against this background, in this section we employ turbo–coding based improvements to the satellite–based DVB system [18] and present performance studies of the proposed system under dispersive channel conditions in conjunction with a variety of blind channel equalisation algorithms. The transmitted power requirements of the standard system employing convolutional codecs can be reduced upon invoking more complex, but more powerful turbo codecs. Alternatively, the standard quaternary or 2–bit/symbol system's bit error rate (BER) versus signal–to–noise ratio (SNR) performance can almost be matched by a turbo–coded 4–bit/symbol 16–level quadrature amplitude modulation (16–QAM) based scheme, while doubling the achievable bit rate within the same bandwidth and hence improving the associated video quality. This is achieved at the cost of an increased system complexity. This system offers a testbed for a real–life application of blind equalisation techniques under mild channel dispersions.

The remainder of the section is organised as follows. A succinct overview of the turbo–coded and standard DVB satellite scheme is presented in Section 2.6.2, while our channel model is described in Section 2.6.3. A brief summary of the blind equaliser algorithms employed is presented in Section 2.6.4. Finally, the performance of the improved DVB satellite system is examined for transmission over a dispersive two–path channel in Section 2.6.5.

[1]This section is based on joint work with my colleague Chee–Siong Lee [10], whose contributions in the field of both video compression and channel coding are gratefully acknowledged

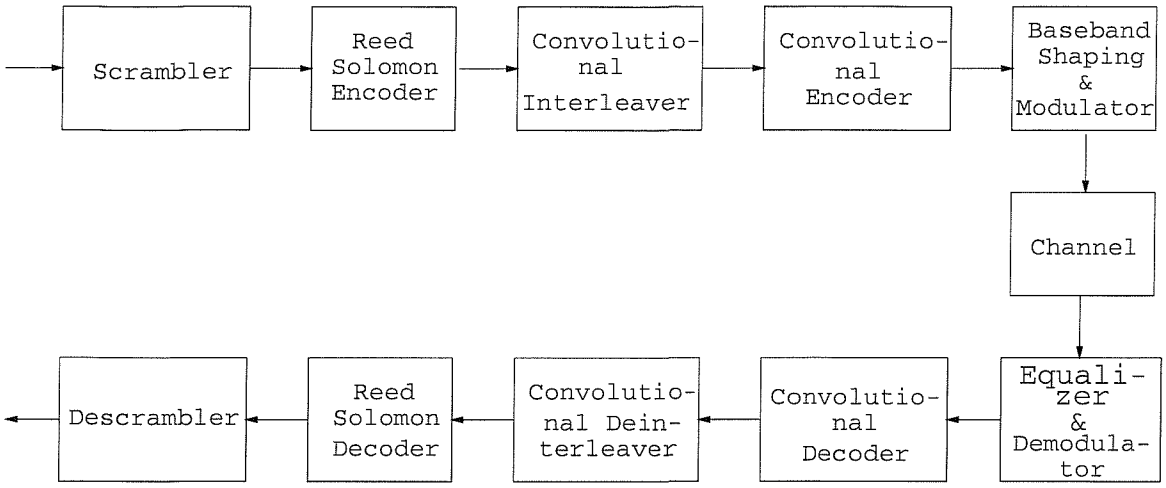


Figure 2.24: Schematic of the DVB satellite system.

2.6.2 DVB Satellite Scheme

The block diagram of the DVB satellite (DVB–S) system [18] is shown in Figure 2.24, which is composed of a MPEG–2 video encoder (not shown in the diagram), channel coding modules and a quadrature phase shift keying (QPSK) modem [161]. The bitstream generated by the MPEG–2 encoder is packetised into frames of 188–byte long, which are then randomised by the scrambler, the details of which can be obtained from the DVB–S standard [18].

Due to the poor error resilience of the MPEG–2 video codec, powerful concatenated channel coding is employed. The concatenated channel codec comprises a shortened Reed–Solomon (RS) outer code and an inner convolutional encoder. The 188–byte MPEG–2 video packet is extended by the Reed–Solomon encoder [130, 160] with parity information to facilitate error recovery to form a 204–byte packet. The Reed–Solomon decoder can then correct up to eight erroneous bytes for each 204–byte packet. Following this, the RS–coded packet is interleaved by a convolutional interleaver and further protected by a half–rate inner convolutional encoder with a constraint length of 7 [130, 160].

Furthermore, the overall code rate of the concatenated coding scheme can be adapted by variable puncturing, not shown in the figure, which supports code rates of 1/2 (no puncturing) as well as 2/3, 3/4, 5/6 and 7/8. The parameters of the convolutional encoder are summarised in Table 2.3.

Convolutional Coder Parameters	
Code Rate	1/2
Constraint Length	7
n	2
k	1
Generator Polynomials (octal format)	171, 133

Table 2.3: Parameters of the CC(n,k,K) convolutional inner encoder of the DVB–S modem.

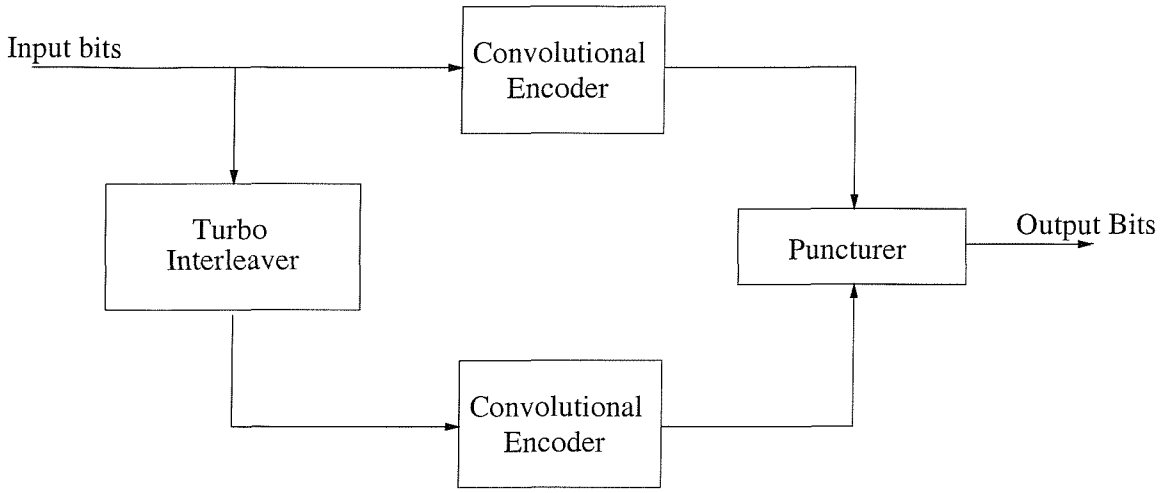


Figure 2.25: Block diagram of turbo encoder.

In addition to implementing the standard DVB–S system as a benchmark, we have improved the system's performance with the aid of a turbo codec [135, 166]. The block diagram of the turbo encoder is shown in Figure 2.25. The turbo encoder is constructed of two component encoders. Each component encoder is a half–rate convolutional encoder, whose parameters are listed in Table 2.4. The two component encoders are used to encode the same input bits, although the input bits of the second component encoder are interleaved before encoding. The output bits of the two component codes are punctured and multiplexed, in order to form a single output bitstream. The component encoder used here is known as a half–rate recursive systematic convolutional encoder (RSC) [167]. It generates one parity bit and one systematic output bit for every input bit. In order to provide an overall coding rate of one half, half the output bits from the two encoders must be punctured. The puncturing arrangement used in our work is to transmit all the systematic bits from the first encoder and every other parity bit from both encoders.

Readers interested in further details of the DVB–S system are referred to the DVB–S standard [18]. The performance of the standard DVB–S system and that of the turbo coded system is characterised in Section 2.6.5. Let us now briefly consider the multipath channel model used in our investigations.

2.6.3 Channel Model

The DVB–S system was designed to operate in the 12 GHz frequency band (K–band). Within this frequency band, tropospheric effects such as the transformation of electromagnetic energy into thermal energy due to induction of currents in rain and ice crystals lead to signal attenuations [169,170]. In the past 20 years, various researchers have concentrated their efforts on attempting to model the satellite channel, typically within a land mobile satellite channel scenario. However, the majority of the work

Turbo Coder Parameters	
Turbo Code Rate	1/2
Input block length	17952 bits
Interleaver Type	Random
Number of turbo decoder iterations	8
Turbo Encoder Component Code Parameters	
Component Code Encoder Type	Convolutional Encoder (RSC)
Component Code Decoder Type	Log–MAP [168]
Constraint Length	3
n	2
k	1
Generator Polynomials (octal format)	7, 5

Table 2.4: Parameters of the inner turbo encoder used to replace the DVB–S system's convolutional coder (RSC: recursive systematic code).

conducted for example by Vogel and his colleagues [171–174] concentrated on modelling the statistical properties of a narrowband satellite channel in lower frequency bands, such as the 870MHz UHF band and the 1.5GHz L–band.

However, our high bitrate DVB satellite system requires a high bandwidth, hence the video bitstream is exposed to dispersive wideband propagation conditions. Recently, Saunders *et. al.* [175, 176] have proposed the employment of multipath channel models to study the satellite channel, although their study was concentrated on the L–band and S–band only.

Due to the dearth of reported work on wideband satellite channel modelling in the K–band, we have adopted a simpler approach. The channel model employed in this study was the two–path (nT)–symbol spaced impulse response, where T is the symbol–duration. In our studies we used $n = 1$ and $n = 2$. This corresponds to a stationary dispersive transmission channel. Our channel model assumed that the receiver had a direct line–of–sight with the satellite as well as a second path caused by a single reflector probably from a nearby building or due to ground reflection. The ground reflection may be strong, if the satellite receiver dish is only tilted at a low angle.

Based on these channel models, we studied the ability of a range of blind equaliser algorithms to converge under various path delay conditions. In the next section we provide a brief overview of the various blind equalisers employed in our experiments, noting that the readers who are mainly interested in the system's performance may proceed directly to our performance analysis section, namely to Section 2.6.5.

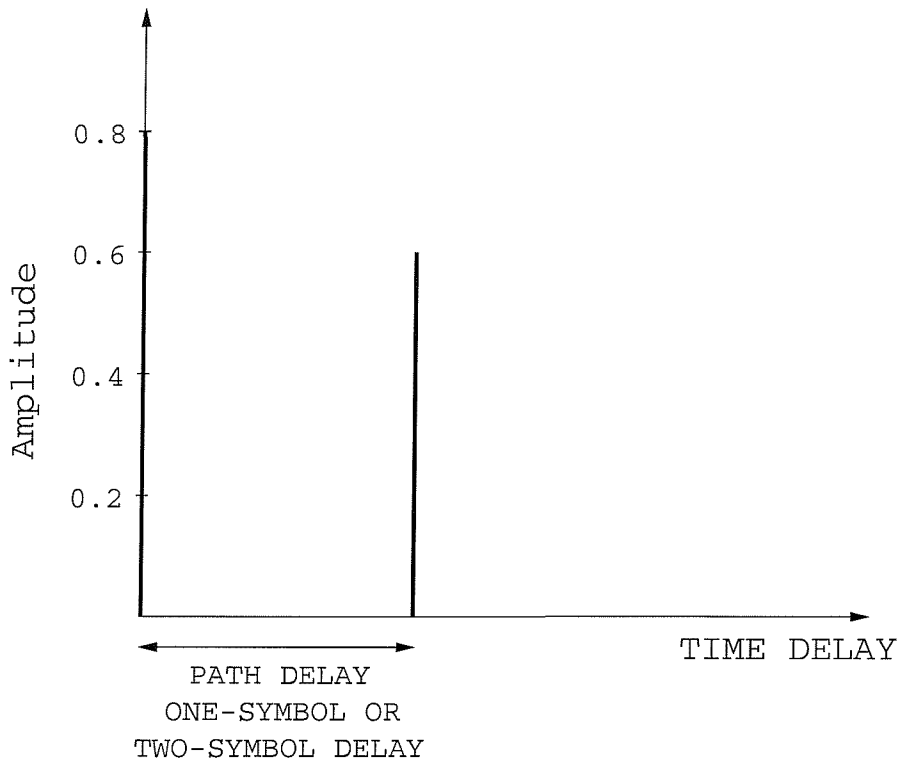


Figure 2.26: Two–path satellite channel model with either a one–symbol or two–symbol delay.

2.6.4 The Blind Equalisers

In this section the blind equalisers used in the system are presented. The following blind equalisers have been studied:

- The Modified Constant Modulus Algorithm (MCMA) of Section 1.3.5.3 [102]
- The Benveniste–Goursat Algorithm (B–G) of Section 1.3.5.4 [32]
- The Stop–and–Go Algorithm (S–a–G)of Section 1.3.5.5 [21]
- The Per–Survivor Processing (PSP) Algorithm of Section 1.3.9 [23], using the M–algorithm with M equal to the number of symbols in the constellation.

A summary of the various equalisers' parameters is given in Table 2.5.

Having described the components of our enhanced DVB–S system, let us now consider the overall system's performance.

2.6.5 Performance of the DVB–S Scheme

In this section, the performance of the DVB–S system was evaluated by means of simulations. Two modulation types were used, namely the standard QPSK and the enhanced 16–QAM schemes [161].

	Step–size λ	No. of Equal. Taps	Initial Tap–Vector
Benveniste–Goursat	$5x10^{-4}$	10	$(1.2, 0, \cdots, 0)$
Modified–CMA	$5x10^{-4}$	10	$(1.2, 0, \cdots, 0)$
Stop–and–Go	$5x10^{-4}$	10	$(1.2, 0, \cdots, 0)$
PSP (1 sym delay)	10^{-2}	2	$(1.2, 0)$
PSP (2 sym delay)	10^{-2}	3	$(1.2, 0, 0)$

Table 2.5: Summary of the equaliser parameters used in the simulations. The tap–vector $(1.2, 0, \cdots, 0)$ indicates that the first equaliser coefficient is initialised to the value 1.2, while the others to 0

The channel model of Figure 2.26 was employed. The first channel model had a one–symbol second–path delay, while in the second one the path–delay corresponded to the period of two symbols. The average BER versus SNR per bit performance was evaluated after the equalisation and demodulation process, as well as after Viterbi [130] or turbo decoding [166]. The SNR per bit or E_b/N_o is defined as follows:

$$\text{SNR per bit} = 10log_{10}\frac{\bar{S}}{\bar{N}} + \delta \tag{2.2}$$

where $\bar{S}$ is the average received signal power, $\bar{N}$ is the average received noise power and δ, which is dependent on the type of modulation scheme used and channel code rate (R), is defined as follows:

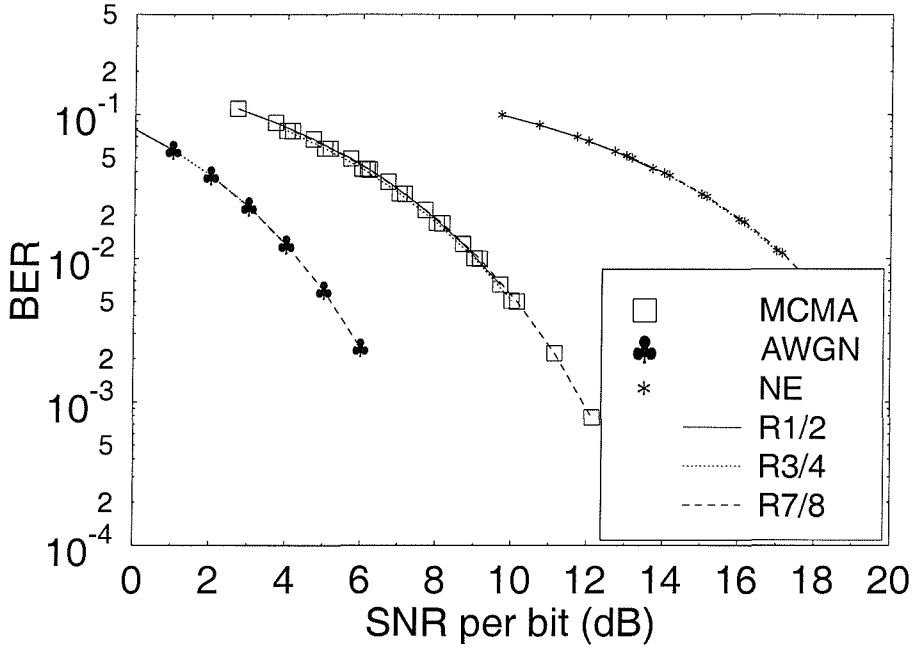
$$\delta = 10log_{10}\frac{1}{\text{R} \times \text{Bits per modulation symbol}}. \tag{2.3}$$

Our results are further divided into two subsections for ease of discussion. First, we will present the system performance over the one–symbol delay two–path channel in Section 2.6.5.1. Next, the system performance over the two–symbol delay two–path channel is presented in Section 2.6.5.2. Lastly, a summary of the system performance is provided in Section 2.6.5.3.

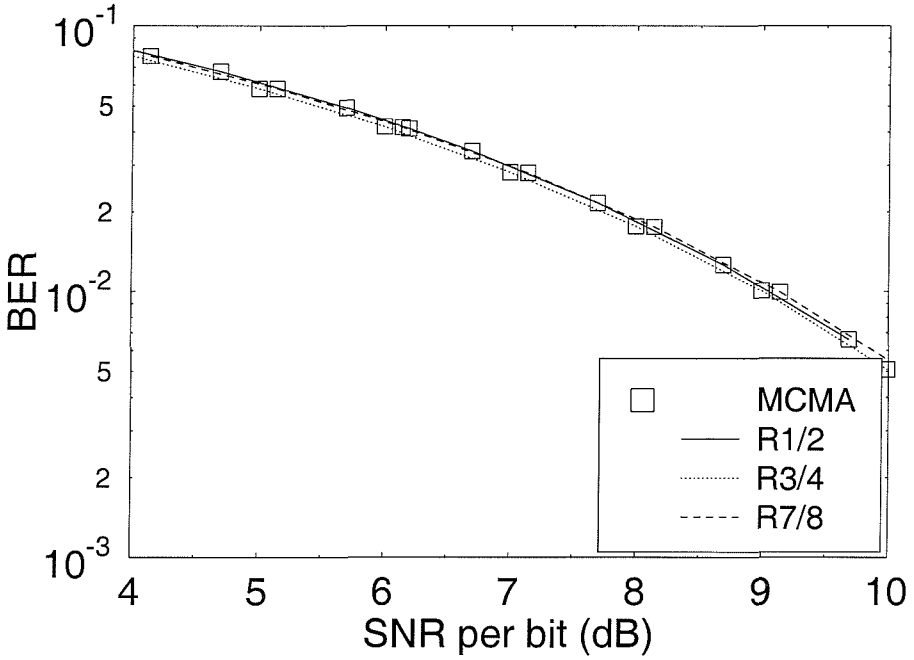
2.6.5.1 Transmission Over the Symbol–spaced Two–path Channel

The linear equalisers' performance was quantified and compared using QPSK modulation over the one–symbol delay two–path channel model of Figure 2.27. Since all the equalisers' BER performance was similar, only the Modified CMA results are shown in the figure.

The equalised performance over the one symbol–spaced channel was inferior to that over the non–dispersive AWGN channel. However, as expected, it was better than without any equalisation. Another observation for Figure 2.27 was that the different punctured channel coding rates appeared to give slightly different bit error rates after equalisation. This was because the linear blind equalisers required uncorrelated input bits in order to converge. However, the input bits were not entirely random,



(a) After equalisation and demodulation



(b) Same as (a) but enlarged in order to show performance difference of the blind equaliser, when different convolutional code rates are used.

Figure 2.27: Average BER versus SNR per bit performance after equalisation and demodulation employing **QPSK** modulation and **one–symbol delay channel** (**NE**: Non–Equalised; **MCMA**: Modified Constant Modulus Algorithm).

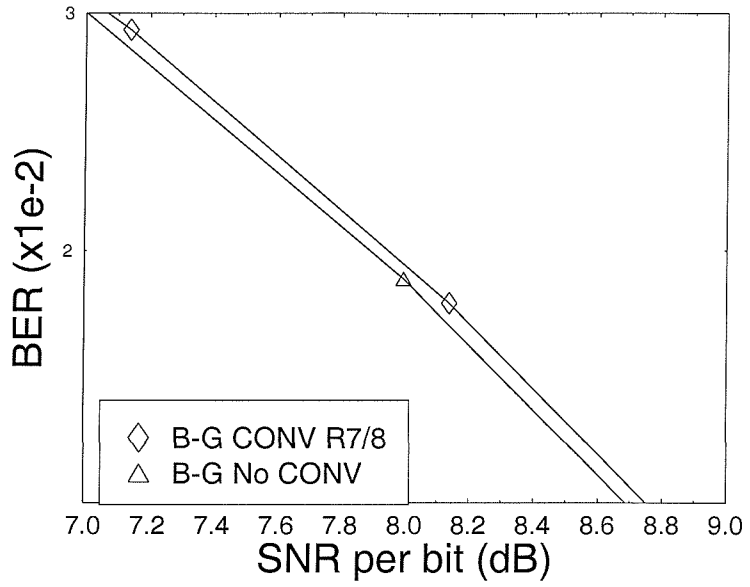


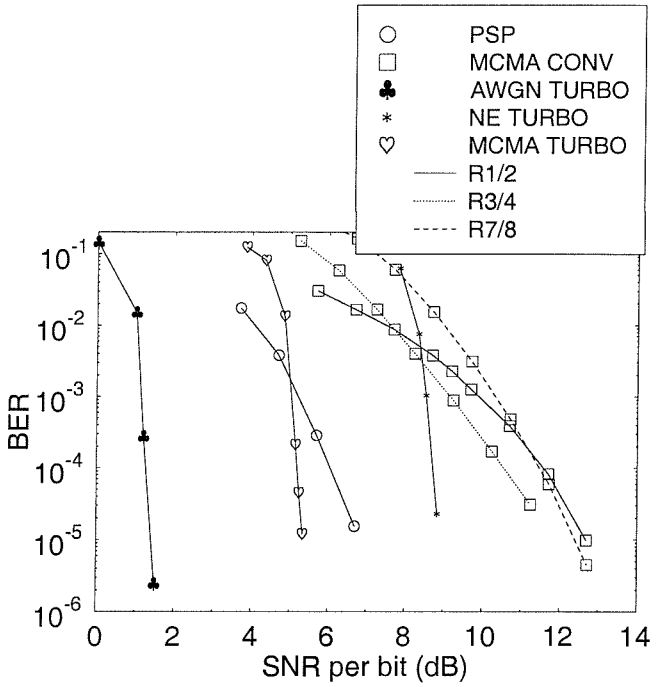
Figure 2.28: Average BER versus SNR per bit performance after equalisation and demodulation employing **QPSK** modulation and the **one–symbol delay two–path channel** of Figure 2.26, for the Benveniste–Goursat algorithm, where the input bits are random (No CONV) or correlated (CONV 7/8) as a result of convolutional coding having a coding rate of 7/8.

when convolutional coding was used. The consequences of violating the zero–correlation constraint are not generally known. Nevertheless, two potential problems were apparent. Firstly, the equaliser may diverge from the desired equaliser equilibrium [22]. Secondly, the performance of the equaliser is expected to degrade, owing to the violation of the randomness requirement, which is imposed on the input bits in order to ensure that the blind equalisers will converge.

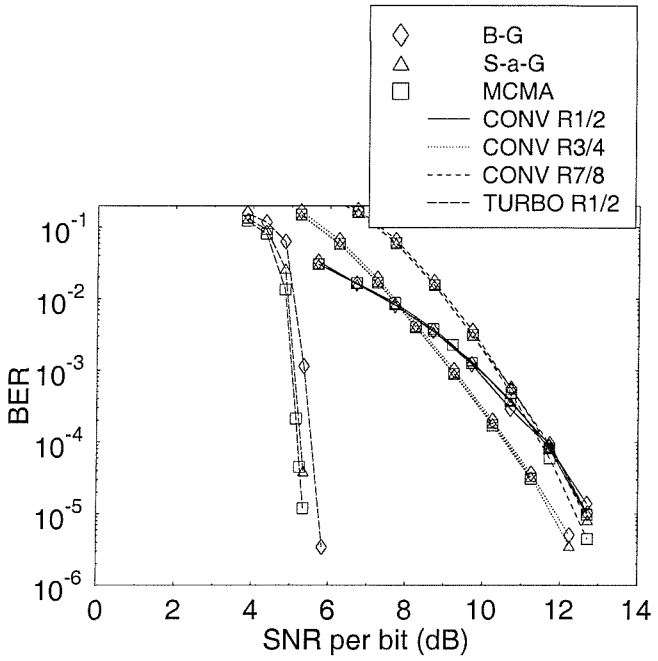
Since the channel used in our investigations was static, the first problem was not encountered. Instead, the second problem was what we actually observed. Figure 2.28 quantifies the equalisers' performance degradation due to convolutional coding. We can observe a $0.1dB$ SNR degradation, when the convolutional codec creates correlation among the bits for this specific case.

The average BER curves after Viterbi or turbo decoding are shown in Figure 2.29(a). In this figure, the average BER over the non–dispersive AWGN channel after turbo decoding constitutes the best case performance, while the average BER of the one–symbol delay two–path MCMA–equalised rate 7/8 convolutionally coded scenario exhibits the worst case performance. Again, in this figure only the Modified–CMA was featured for simplicity. The performance of the remaining equalisers was characterised in Figure 2.29(b). Clearly, the performance of all the linear equalisers investigated was similar.

It is observed in Figure 2.29(a) that the combination of the Modified CMA blind equaliser with turbo decoding exhibited the best SNR performance over the one–symbol delay two–path channel. The only



(a) PSP and linear equalisers



(b) Linear equalisers only

Figure 2.29: Average BER versus SNR per bit performance after convolutional or turbo decoding for **QPSK** modulation and **one–symbol delay channel** (**NE**: Non–Equalised; **B–G**: Benveniste–Goursat; **S–a–G**: Stop–and–Go; **MCMA**: Modified Constant Modulus Algorithm; **PSP**: Per–Survivor–Processing).

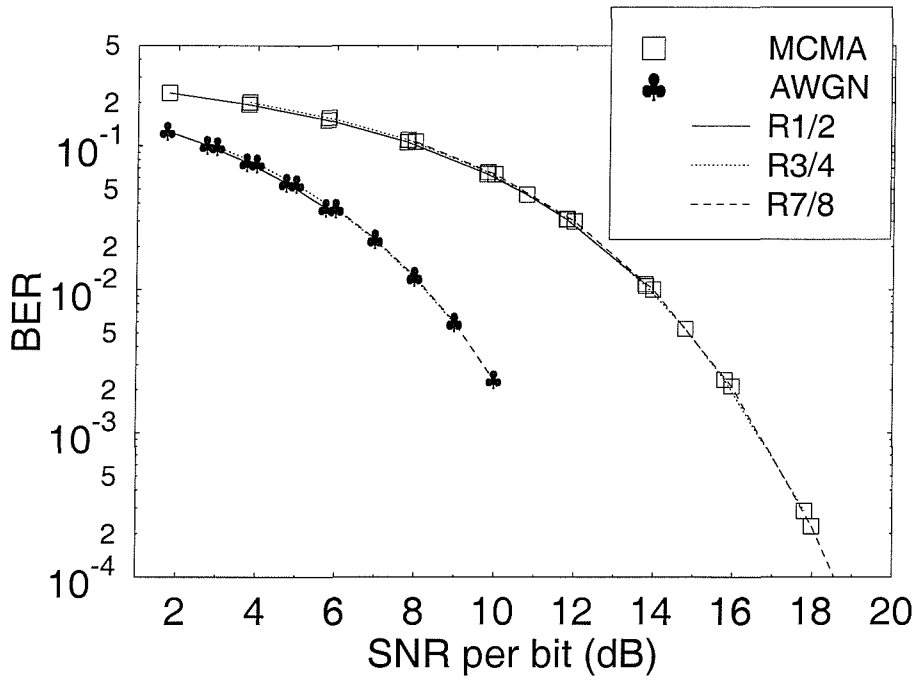
comparable alternative was the PSP algorithm. Although the performance of the PSP algorithm was better at low SNRs, the associated curves cross over and the PSP algorithm's performance became inferior below the average BER of 10^{-3}. This happens due to the iterative nature of turbo decoding, which is in contrast to the combined equalisation and non–iterative channel decoding of PSP. It is well recognised that iterative channel decoders typically exhibit a better performance for high SNRs, than their non–iterative counterparts. In this case, despite the fact that the Modified CMA exhibites considerably worse performance than PSP, if it is used in conjunction with non–iterative channel decoding, when turbo coding is employed, its performance is dramatically improved, especially for high SNRs. Although not shown in Figure 2.29, the Reed–Solomon decoder, which was concatenated to either the convolutional or the turbo decoder, became effective, when the average BER of its input was below approximately 10^{-4}. In this case, the PSP algorithm performed by at least $1dB$ worse in the area of interest, which is at an average BER of 10^{-4}.

A final observation in the context of Figure 2.29(a) is that when convolutional decoding was used, the associated E_b/N_o performance of the rate 1/2 convolutional coded scheme appeared slightly inferior to that of the rate 3/4 and the rate 7/8 scenarios beyond certain E_b/N_o values. This was deemed to be a consequence of the fact that the 1/2–rate encoder introduced more correlation into the bitstream than its higher rate counterparts and this degraded the performance of the blind channel equalisers, which performed best, when fed with random bits.

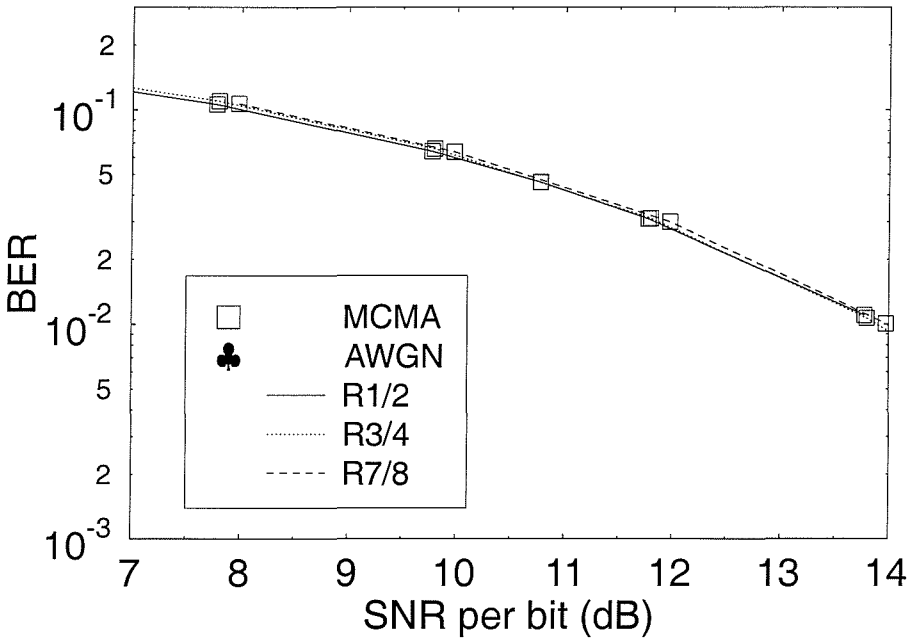
Having considered the QPSK case, we shall now concentrate on the enhanced system, which employed 16–QAM under the same channel and equaliser conditions. In Figure 2.30 and Figure 2.31, the performance of the DVB system employing 16–QAM is presented. Again, for simplicity, only the Modified CMA results are given. In this case the ranking order of the different coding rates followed our expectations more closely in the sense that the lowest coding rate of 1/2 was the best performer, followed by rate 3/4 codec, in turn followed by the least powerful rate 7/8 codec.

The Stop–and–Go algorithm has been excluded from these results, since it does not converge for high SNR values. This happens, because the equalisation procedure is only activated, when there is a high probability of correct decision–directed equaliser update. In our case, the equaliser is initialised far from its convergence point and hence the decision–directed updates are unlikely to be correct. In the absence of noise this leads to the update algorithm being permanently de–activated. If noise is present though, then some random perturbations from the point of the equaliser's initialisation can activate the Stop–and–Go algorithm and can lead to convergence. We made this observation at medium SNR values in our simulation study. For high SNR values though, the algorithm did not converge.

It is also interesting to compare the performance of the system for the QPSK and 16–QAM schemes. When the one–symbol delay two–path channel model of Figure 2.26 was considered, the system was



(a) After equalisation and demodulation



(b) Same as (a) but enlarged in order to show performance difference of the blind equaliser, when different convolutional code rates are used.

Figure 2.30: Average BER versus SNR per bit after equalisation and demodulation for **16–QAM** over the **one–symbol delay two–path channel** of Figure 2.26 (**MCMA**: Modified Constant Modulus Algorithm).

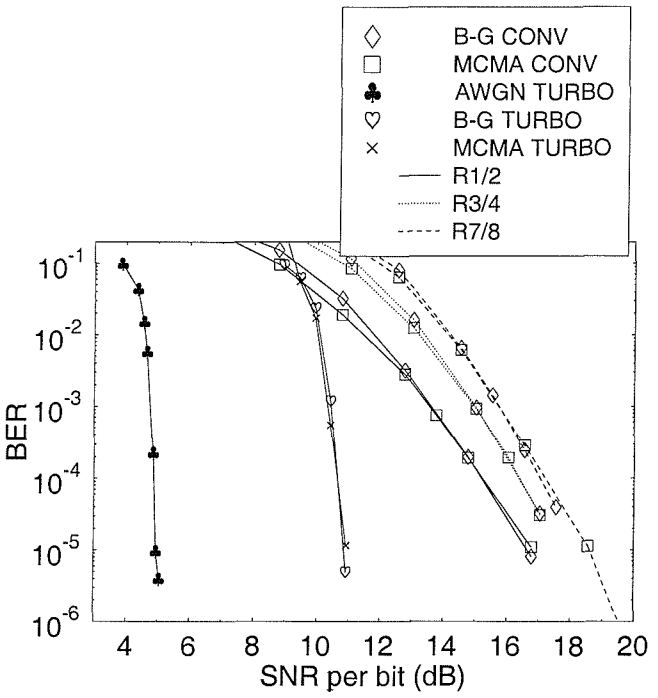


Figure 2.31: Average BER versus SNR per bit after Viterbi or turbo decoding for **16–QAM** over the **one–symbol delay two–path channel** of Figure 2.26 (**B–G**: Benveniste–Goursat; **S–a–G**: Stop–and–Go; **MCMA**: Modified Constant Modulus Algorithm; **PSP**: Per–Survivor–Processing).

capable of supporting the use of 16-QAM with the provision of an additional SNR per bit of approximately $4 - 5dB$. This observation was made by comparing the performance of the DVB system when employing the Modified CMA and the half-rate convolutional or turbo code in Figure 2.29 and Figure 2.31 at a BER of 10^{-4}. Although the original DVB–Satellite system only employs QPSK modulation, our simulations had shown that 16–QAM can be employed equally well for the range of blind equalisers that we have used in our work. This allowed us to double the video bitrate and hence to substantially improve the video quality. The comparison of Figures 2.29 and 2.31 also reveals that the extra SNR requirement of approximately $4 - 5dB$ of 16–QAM over QPSK can be eliminated by employing turbo coding at the cost of a higher implementational complexity. This allowed us to accommodate a doubled bitrate within a given bandwidth, which improved the video quality.

2.6.5.2 Transmission Over the Two–symbol Delay Two–path Channel

In Figures 2.32 (only for the Benveniste–Goursat algorithm for simplicity) and 2.33 the corresponding BER results for the two–symbol delay two–path channel of Figure 2.26 are given for QPSK. The associated trends are similar to those in Figures 2.27 and 2.29, although some differences can be observed, as listed below:

- The "cross–over point", beyond which the performance of the PSP algorithm was inferior to that of the Modified CMA in conjunction with turbo decoding is now at 10^{-4}, which is in the range, where the RS decoder guarantees an extremely low probability of error.

- The rate 1/2 convolutional decoding was now the best performer, when convolutional decoding is concerned, while the rate 3/4 scheme exhibited the worst performance.

Finally, in Figure 2.34, the associated 16–QAM results are presented. Notice that the Stop–and–Go algorithm was again excluded from the results. Furthermore, we observe a high performance difference between the Benveniste–Goursat algorithm and the Modified CMA. In the previous cases we did not observe such a significant difference. The difference in this case is that the channel exhibits an increased delay spread. This illustrated the capability of the equalisers to cope with more widespread multipaths, while keeping the equaliser order constant at 10. The Benveniste–Goursat equaliser was more efficient, than the Modified CMA in this case.

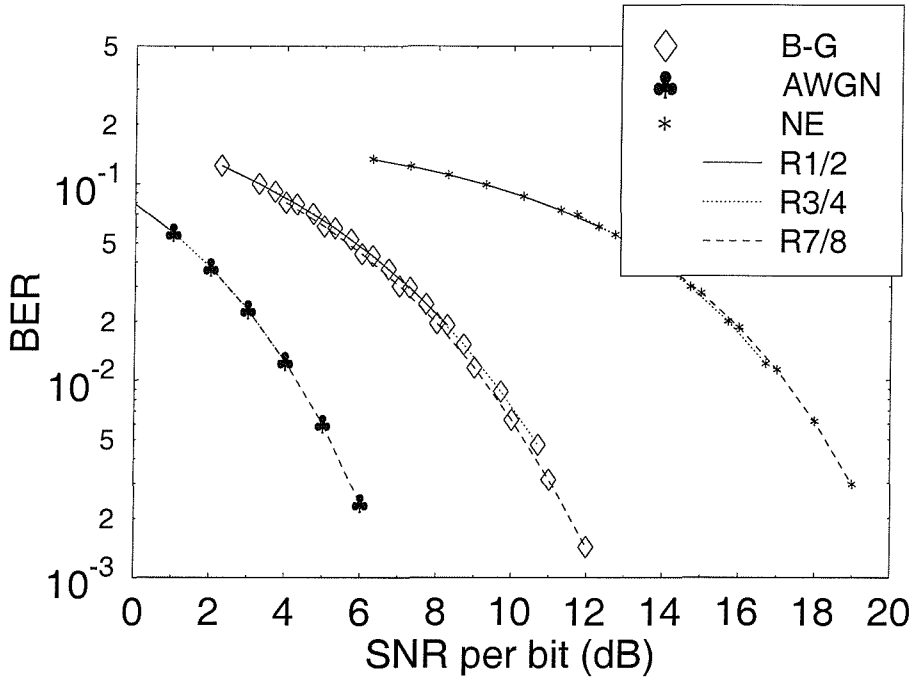
It is interesting to note that in this case, the performance of the different coding rates was again in the expected order, the rate 1/2 being the best, followed by the rate 3/4 and then the rate 7/8 scheme.

If we compare the performance of the system employing QPSK and 16–QAM over the two–symbol delay two–path channel of Figure 2.26, we again observe that **16–QAM can be incorporated into the DVB system, if an extra $5dB$ of SNR per bit is affordable in power budget terms. However, only the B–G algorithm is worthwhile considering here out of the three linear equalisers of Table 2.5.** This observation was made by comparing the performance of the DVB system when employing the Benveniste–Goursat equaliser and the half–rate convolutional coder in Figure 2.33 and Figure 2.34.

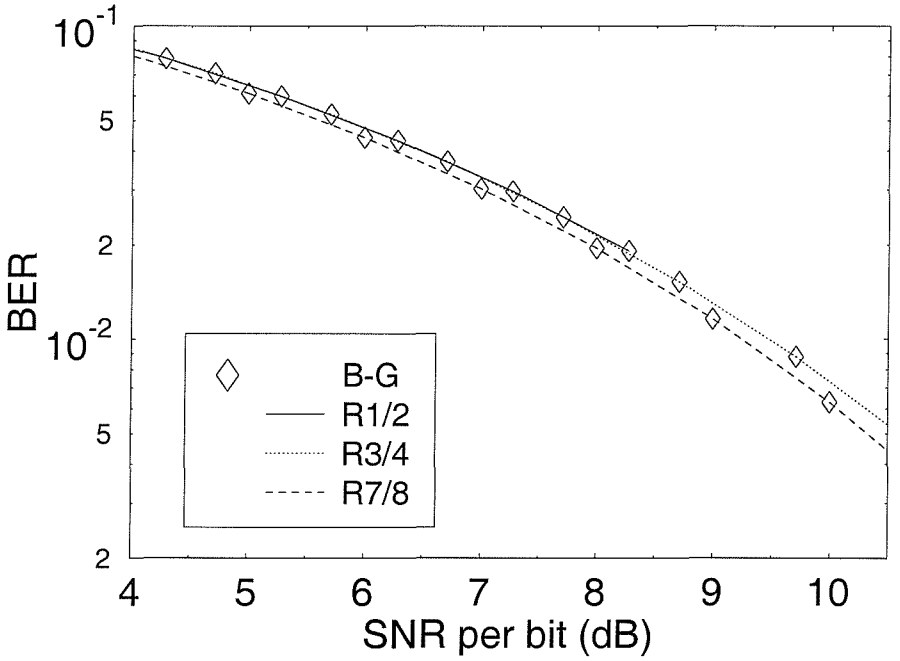
2.6.5.3 Performance Summary of the DVB–S System

Table 2.6 provides an approximation of the convergence speed of each blind equalisation algorithm of Table 2.5. It is clear that PSP exhibited the fastest convergence, followed by the Benveniste–Goursat algorithm. In our simulations the convergence was quantified by observing the slope of the BER curve, and finding when this curve was reaching the associated residual BER, implying that the BER has reached its steady–state value. Figure 2.35 gives an illustrative example of the equaliser's convergence for 16–QAM. The Stop–and–Go algorithm converges significantly slower than the other algorithms, which can also be seen from Table 2.6. This happens because, during the startup, the algorithm is de–activated most of the time, an effect which becomes more severe with an increasing QAM order.

We now define the average peak signal–to–noise ratio (PSNR) for the one–symbol delay and two–

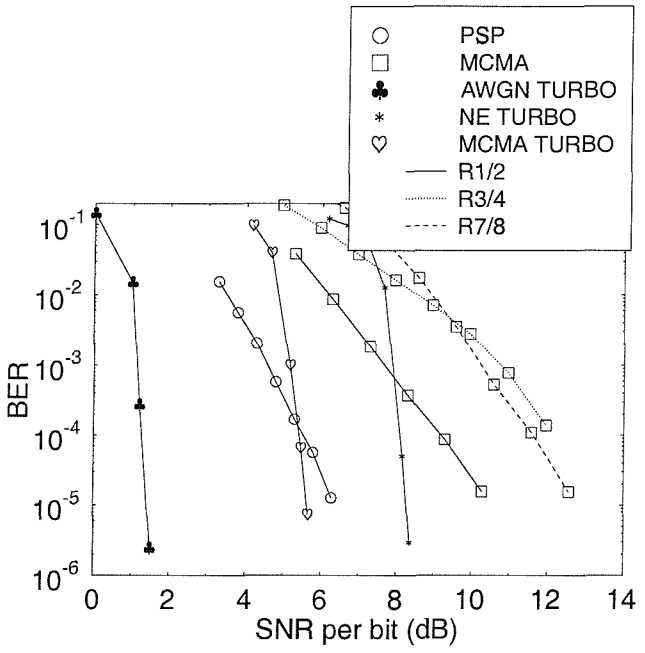


(a) After equalisation and demodulation

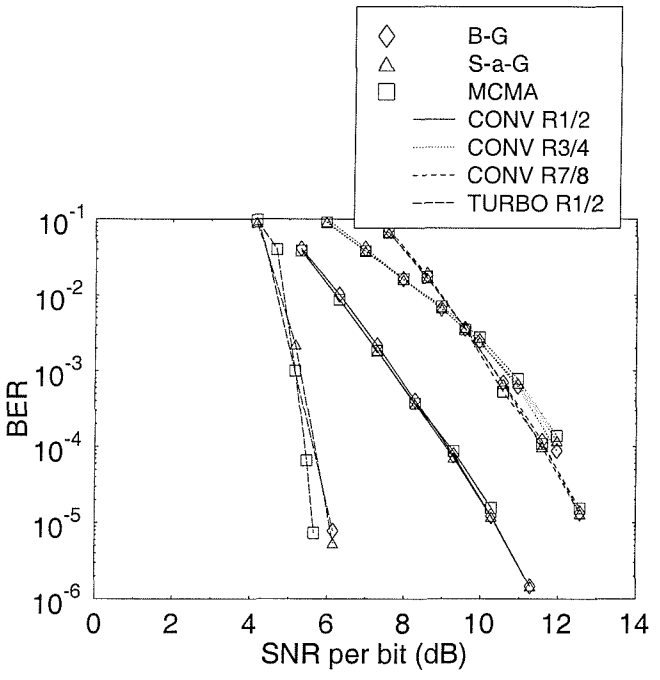


(b) Same as (a) but enlarged in order to show performance difference of the blind equaliser, when different convolutional code rates are used.

Figure 2.32: Average BER versus SNR per bit performance after equalisation and demodulation for **QPSK** modulation over the **two–symbol delay two–path channel** of Figure 2.26 (**B–G**: Benveniste–Goursat).

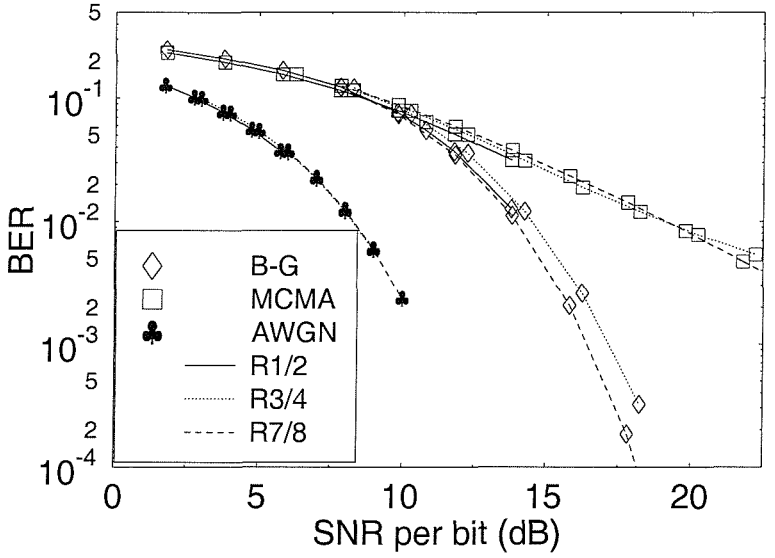


(a) PSP and linear equalisers

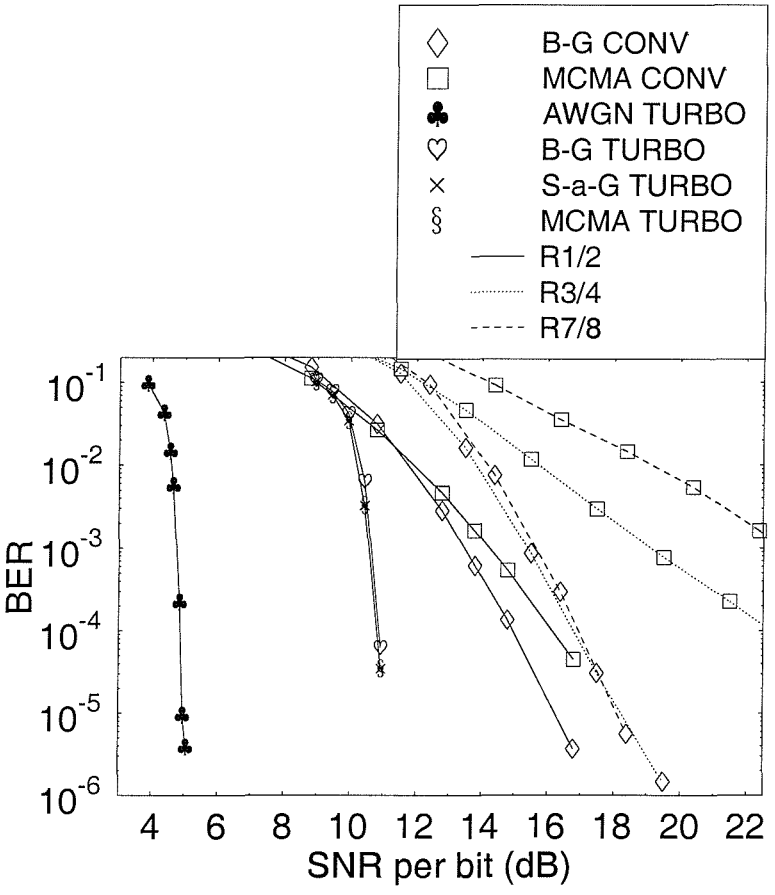


(b) Linear equalisers only

Figure 2.33: Average BER versus SNR per bit performance after convolutional or turbo decoding for **QPSK** modulation over the **two–symbol delay two–path channel** of Figure 2.26 (**B–G**: Benveniste–Goursat; **S–a–G**: Stop–and–Go; **MCMA**: Modified Constant Modulus Algorithm; **PSP**: Per–Survivor–Processing).



(a) After equalisation and demodulation



(b) After viterbi or turbo decoding

Figure 2.34: Average BER versus SNR per bit performance (a) after equalisation and demodulation and (b) after Viterbi or turbo decoding for **16–QAM** over the **two–symbol delay two–path channel** of Figure 2.26 (**B–G**: Benveniste–Goursat; **S–a–G**: Stop–and–Go; **MCMA**: Modified Constant Modulus Algorithm; **PSP**: Per–Survivor–Processing).

	B–G	MCMA	S–a–G	PSP
QPSK 1 sym	73	161	143	0.139
QPSK 2 sym	73	143	77	0.139
16–QAM 1 sym	411	645	1393	
16–QAM 2 sym	359	411	1320	

Table 2.6: Equaliser convergence speed (in miliseconds) measured in the simulations, given as an estimate of time required for convergence when 1/2 rate puncturing is used (x sym: x–symbol delay two–path channel and x can take either the value 1 or 2).

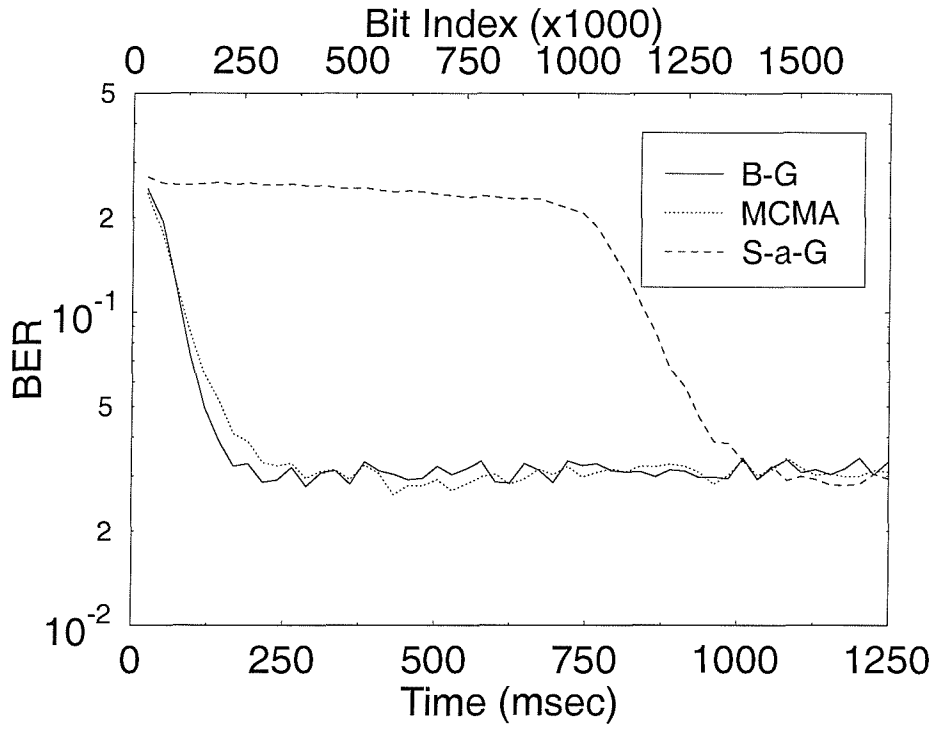


Figure 2.35: Learning curves for 16–QAM, one–symbol delay two–path channel at SNR=18dB.

symbol delay two–path channel model of Figure 2.26 as:

$$\text{PSNR} = 10log_{10}\frac{\sum_{n=0}^{N}\sum_{m=0}^{M}255^2}{\sum_{n=0}^{N}\sum_{m=0}^{M}\Delta^2}, \tag{2.4}$$

where Δ is the difference between the uncoded pixel value and the reconstructed pixel value. The variables M and N refer to the dimension of the image. The maximum possible 8–bit represented pixel luminance value of 255 was used in Equation 2.4 in order to mitigate the PSNR's dependence on the video material used. The average PSNR is then the mean of the PSNR values computed for all the images constituting the video sequence.

Tables 2.7 and 2.8 provide a summary of the DVB–Satellite system's performance tolerating a PSNR degradation of $2dB$, which was deemed to be nearly imperceptible in terms of subjective video degradations. The average BER values quoted in the tables refer to the average BER achieved after Viterbi or turbo decoding. The channel SNR is quoted in association with the $2dB$ average video PSNR degradation, since the viewer will begin to perceive video degradations due to erroneous decoding of the received video around this threshold.

Mod.	Equaliser	Code	CSNR (dB)	E_b/N_0
QPSK	PSP(Rate 1/2)		5.3	5.3
QPSK	MCMA	Turbo (1/2)	5.2	5.2
16QAM	MCMA	Turbo (1/2)	13.6	10.6
QPSK	MCMA	Conv (1/2)	9.1	9.1
16QAM	MCMA	Conv (1/2)	17.2	14.2
QPSK	MCMA	Conv (3/4)	11.5	9.7
16QAM	MCMA	Conv (3/4)	20.2	15.4
QPSK	B–G	Conv (7/8)	13.2	10.8
16QAM	B–G	Conv (7/8)	21.6	16.2

Table 2.7: Summary of performance results over the dispersive one–symbol delay two–path AWGN channel of Figure 2.26 tolerating a PSNR degradation of $2dB$.

Tables 2.9 and 2.10 provide a summary of the SNR per bit required for the various system configurations. The BER threshold of 10^{-4} was selected here, since at this average BER after Viterbi or turbo decoding, the RS decoder becomes effective, guaranteeing near error–free performance. This also translates into near unimpaired reconstructed video quality.

Finally, in Table 2.11 the QAM symbol rate or Baud rate is given for different puncturing rates and for different modulation schemes, based on the requirement of supporting a video bit rate of 2.5 Mbit/sec. We observe that the Baud rate is between 0.779 and 2.73 MBd, depending on the coding rate and the number of bits per modulation symbol.

Mod.	Equaliser	Code	CSNR (dB)	E_b/N_0
QPSK	PSP(Rate 1/2)		4.7	4.7
QPSK	B–G	Turbo (1/2)	5.9	5.9
16QAM	B–G	Turbo (1/2)	13.7	10.7
QPSK	B-G	Conv (1/2)	8.0	8.0
16QAM	B–G	Conv (1/2)	17.0	14.0
QPSK	B–G	Conv (3/4)	12.1	10.3
16QAM	B–G	Conv (3/4)	21.1	16.3
QPSK	B–G	Conv (7/8)	13.4	11.0
16QAM	MCMA	Conv (7/8)	29.2	23.8

Table 2.8: Summary of performance results over the dispersive two–symbol delay two–path AWGN channel of Figure 2.26 tolerating a PSNR degradation of $2dB$.

Mod.	Equaliser	Code	E_b/N_0
QPSK	PSP(Rate 1/2)		6.1
QPSK	MCMA	Turbo (1/2)	5.2
16QAM	MCMA	Turbo (1/2)	10.7
QPSK	MCMA	Conv (1/2)	11.6
16QAM	MCMA	Conv (1/2)	15.3
QPSK	MCMA	Conv (3/4)	10.5
16QAM	MCMA	Conv (3/4)	16.4
QPSK	B–G	Conv (7/8)	11.8
16QAM	B–G	Conv (7/8)	17.2

Table 2.9: Summary of system performance results over the dispersive one–symbol delay two–path AWGN channel of Figure 2.26 tolerating an average BER of 10^{-4}, which was evaluated after Viterbi or turbo decoding but before RS decoding.

Mod.	Equaliser	Code	E_b/N_0
QPSK	PSP(Rate 1/2)		5.6
QPSK	B–G	Turbo (1/2)	5.7
16QAM	B–G	Turbo (1/2)	10.7
QPSK	B–G	Conv (1/2)	9.2
16QAM	B–G	Conv (1/2)	15.0
QPSK	B–G	Conv (3/4)	12.0
16QAM	B–G	Conv (3/4)	16.8
QPSK	B–G	Conv (7/8)	11.7
16QAM	MCMA	Conv (7/8)	26.0

Table 2.10: Summary of system performance results over the dispersive two–symbol delay two–path AWGN channel of Figure 2.26 tolerating an average BER of 10^{-4}, which was evaluated after Viterbi or turbo decoding but before RS decoding.

Punctured Rate	4–QAM Baud Rate (MBd)	16–QAM Baud Rate (MBd)
1/2	2.73	1.37
3/4	1.82	0.909
7/8	1.56	0.779

Table 2.11: The channel bit rate for the three different punctured coding rates and for the two modulation schemes used.

2.7 Summary

This chapter has characterised the performance of selected blind equalisers. The Busssgang equalisers exhibit a low complexity, but require a considerable number of symbols to converge. Moreover, their convergence accuracy is mediocre. By contrast, the M–algorithm appears to be the best choice in performance terms, giving the best convergence speed and tracking performance at the cost of a higher complexity, depending on the channel's delay spread and the number of survivors M.

An application example of using blind equalisation techniques was also presented in the context of a satellite–based DVB system. It was found that the employment of turbo channel coding resulted in about $5dB$ gain in comparison to the standard convolutional coding based scheme. The associated power budget savings can be invested for example into employing 16–QAM instead of 4–QAM, which exhibites a factor two higher throughput. This doubled throughput can then be used either for doubling the number of DVB channels supported in a given bandwidth, or for increasing the video quality of each fixed–bandwidth channel. Further research has to confirm these findings also in different wireless propagation scenarios. A range of further wireless video communications issues are addressed in [10, 177].

Chapter 3

Soft Decision–Feedback Equalisation Using the Constant Modulus Algorithm

3.1 Introduction

Having introduced a range of blind equalisers and having studied their performance, in this chapter we propose a novel blind equalisation technique, based on the well-known CMA, which was characterised in Section 1.3.5.2. We extend the CMA–based equaliser [4] by making use of soft feedback information input to a DFE similar to the equalisers dicsussed in Section 1.2.0.5. This equaliser is constituted by an infinite impulse response filter and as such it is potentially unstable. Similar work has been reported in [178], where a lattice–form CMA based Decision–Feedback Equaliser (DFE) using the Shtrom–Fan cost function was explored and in [179], where a predictive DFE–CMA was proposed. Furthermore, in [180] two different blind DFE systems were explored using combinations of the CMA and the Shtrom-Fan cost functions for the forward and the feedback parts of the equaliser, respectively. Finally, in [181], a modified version of the CMA, namely the MCMA was extended to a DFE.
The chapter is organised as follows. The proposed equaliser is described in Section 3.2. In Section 3.3 the associated convergence issues of this equaliser are discussed and in Section 3.4 the equaliser's performance is evaluated with the aid of computer simulations.

3.2 System Description

The communications system under consideration is shown in Figure 3.1. In the proposed scheme the

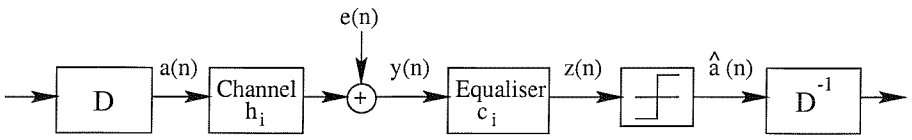


Figure 3.1: Equalised communications system

information bits are differentially encoded in order to be able to demodulate the information, when the blind equaliser converges to a sign-reversed solution. The differentially coded bits are then mapped to the Quadrature Amplitude Modulation (QAM) symbols $a(n)$. These symbols are convolved with the CIR h_i and are contaminated by the channel noise $e(n)$, yielding the received symbols $y(n)$ as:

$$y(n) = \sum_{i=-L_1}^{L_2} h_i \cdot a(n-i) + e(n). \tag{3.1}$$

The restoration of the original information bits is performed by the DFE–CMA equaliser. We will describe the operation of this equaliser below. The DFE–CMA equaliser performs channel equalisation and delivers the equalised signal $z(n)$ as the output of a feedforward and a feedback filter. In mathematical terms the equalised signal is given by:

$$z(n) = \mathbf{c}^T \cdot \mathbf{y}(n) + \mathbf{w}^T \cdot \mathbf{z}(n-1) \tag{3.2}$$

where $\mathbf{c} = [c_0, c_1, \cdots, c_N]^T$ and $\mathbf{w} = [w_0, w_1, \cdots, x_M]^T$ are the feedforward and feedback equaliser coefficients respectively, $\mathbf{y}(n) = [y(n), y(n-1), \cdots, y(n-N)]^T$ is the received signal vector and $\mathbf{z}(n-1) = [z(n-1), z(n-2), \cdots, z(n-N-1)]^T$ is the equalised signal vector. If we consider the z-transform of this relationship, we arrive at:

$$Z(z) = \frac{C(z)}{1+W(z)} \cdot Y(z) = \frac{C(z)}{1+W(z)} \cdot H(z) \cdot A(z), \tag{3.3}$$

where $Z(z)$, $Y(z)$ and $A(z)$ are the z-transforms of the equalised, received and transmitted signals respectively, while $H(z)$, $C(z)$ and $W(z)$ are the CIR and the equaliser's feedforward and feedback vectors in the z-domain, respectively. If the channel was described by a linear model having a finite number of taps, then all that this system would have to do to equalise $Y(z)$ would be to find $W(z)$, so that $1 + W(z) = H(z)$. By contrast, if the channel's transfer function also contains poles, then the feedforward section would also have to be active, so that $(1 + W(z))/C(z) = H(z)$. By considering

the CMA cost function [4], we can similarly formulate the cost function for this algorithm as:

$$J(n) = |e(n)|^2 = \left(|z(n)|^2 - R_2\right)^2, \tag{3.4}$$

where R_2 is the constant used in the CMA. By utilising the classic gradient descent algorithm, we can now find the equaliser coefficient update procedure by differentiating the cost function of Equation (3.4) with respect to (wrt) both the feedforward and feedback coefficients, yielding the update algorithm of:

$$\mathbf{c}^{(n)} = \mathbf{c}^{(n-1)} - \lambda \cdot \frac{\partial J(n-1)}{\partial \mathbf{c}} \tag{3.5}$$

$$\mathbf{w}^{(n)} = \mathbf{w}^{(n-1)} - \lambda \cdot \frac{\partial J(n-1)}{\partial \mathbf{w}}, \tag{3.6}$$

where λ is the step-size parameter, controling the learning rate of the equaliser. Before proceeding further, we need to find the expressions for the derivatives of Equation (3.4) wrt $\mathbf{c}$ and $\mathbf{w}$:

$$\frac{\partial J(n)}{\partial \mathbf{c}} = 2 \cdot e(n) \cdot [z^*(n) \cdot \frac{\partial \mathbf{z}^T(n-1)}{\partial \mathbf{c}} \cdot \mathbf{w} + z(n) \cdot (\mathbf{y}^*(n) + \frac{\partial \mathbf{z}^H(n-1)}{\partial \mathbf{c}} \cdot \mathbf{w}^*)] \tag{3.7}$$

and

$$\frac{\partial J(n)}{\partial \mathbf{w}} = 2 \cdot e(n) \cdot [z^*(n) \cdot \frac{\partial \mathbf{z}^T(n-1)}{\partial \mathbf{w}} \cdot \mathbf{w} + z(n) \cdot (\mathbf{z}^*(n-1) + \frac{\partial \mathbf{z}^H(n-1)}{\partial \mathbf{w}} \cdot \mathbf{w}^*)]. \tag{3.8}$$

As we observe from Equations (3.7) and (3.8), the calculation of the gradients requires an estimate of the derivative of the equalised vectors with respect to the equaliser feedforward and feedback tap-vectors. These derivatives are constituted by matrices and their elements will be close to zero, when the equaliser is close to convergence, however these low–valued elements are needed in order to bring the equaliser to this point. In order to estimate these derivatives we develop a recursive loop. Firstly, we define the following variable matrices and vectors:

$$\mathbf{c} = \mathbf{c_R} + j\mathbf{c_I} \tag{3.9}$$

$$\mathbf{w} = \mathbf{w_R} + j\mathbf{w_I} \tag{3.10}$$

$$\begin{pmatrix} \mathbf{C_{RR}}(n) & \mathbf{C_{RI}}(n) \\ \mathbf{C_{IR}}(n) & \mathbf{C_{II}}(n) \end{pmatrix} = \begin{pmatrix} \frac{\partial \mathbf{z}_R^T(n)}{\partial \mathbf{c}_R} & \frac{\partial \mathbf{z}_R^T(n)}{\partial \mathbf{c}_I} \\ \frac{\partial \mathbf{z}_I^T(n)}{\partial \mathbf{c}_R} & \frac{\partial \mathbf{z}_I^T(n)}{\partial \mathbf{c}_I} \end{pmatrix} \tag{3.11}$$

$$\mathbf{C}(n) = \frac{\partial \mathbf{z}^T(n)}{\partial \mathbf{c}} = \mathbf{C_{RR}}(n) - \mathbf{C_{II}}(n) + j(\mathbf{C_{RI}}(n) + \mathbf{C_{IR}}(n))$$

$$\bar{\mathbf{C}}(n) = \frac{\partial \mathbf{z}^H(n)}{\partial \mathbf{c}} = \mathbf{C_{RR}}(n) + \mathbf{C_{II}}(n) + j(\mathbf{C_{RI}}(n) - \mathbf{C_{IR}}(n)) \tag{3.12}$$

$$\begin{pmatrix} \mathbf{W_{RR}}(n) & \mathbf{W_{RI}}(n) \\ \mathbf{W_{IR}}(n) & \mathbf{W_{II}}(n) \end{pmatrix} = \begin{pmatrix} \frac{\partial \mathbf{z}_R^T(n)}{\partial \mathbf{w}_R} & \frac{\partial \mathbf{z}_R^T(n)}{\partial \mathbf{w}_I} \\ \frac{\partial \mathbf{z}_I^T(n)}{\partial \mathbf{w}_R} & \frac{\partial \mathbf{z}_I^T(n)}{\partial \mathbf{w}_I} \end{pmatrix} \tag{3.13}$$

$$\begin{aligned} \mathbf{W}(n) = \frac{\partial \mathbf{z}^T(n)}{\partial \mathbf{w}} &= \mathbf{W_{RR}}(n) - \mathbf{W_{II}}(n) + j(\mathbf{W_{RI}}(n) + \mathbf{W_{IR}}(n)). \\ \bar{\mathbf{W}}(n) = \frac{\partial \mathbf{z}^H(n)}{\partial \mathbf{w}} &= \mathbf{W_{RR}}(n) + \mathbf{W_{II}}(n) + j(\mathbf{W_{RI}}(n) - \mathbf{W_{IR}}(n)). \end{aligned} \tag{3.14}$$

With these definitions, Equations (3.7) and (3.8) become:

$$\begin{aligned} \frac{\partial J(n)}{\partial \mathbf{c}} &= 4 \cdot e(n) \cdot [0.5 \cdot z(n) \cdot \mathbf{y}^*(n) + z_R(n) \cdot ((\mathbf{C_{RR}} \cdot \mathbf{w_R} - \mathbf{C_{IR}} \cdot \mathbf{w_I}) + \\ &\quad z_I(n) \cdot (\mathbf{C_{RR}} \cdot \mathbf{w_I} + \mathbf{C_{IR}} \cdot \mathbf{w_R}) + j(z_R(n) \cdot ((\mathbf{C_{RI}} \cdot \mathbf{w_R} - \mathbf{C_{II}} \cdot \mathbf{w_I}) + \\ &\quad z_I(n) \cdot (\mathbf{C_{RI}} \cdot \mathbf{w_I} + \mathbf{C_{II}} \cdot \mathbf{w_R}))))] \\ &= 4 \cdot e(n) \cdot [0.5 \cdot z(n) \cdot \mathbf{y}^*(n) + \begin{pmatrix} \mathbf{w_R} & -\mathbf{w_I} \\ j\mathbf{w_I} & j\mathbf{w_R} \end{pmatrix} \begin{pmatrix} \mathbf{C_{RR}} & -\mathbf{C_{RI}} \\ \mathbf{C_{IR}} & \mathbf{C_{II}} \end{pmatrix} \cdot \begin{pmatrix} z_R(n) \\ z_I(n) \end{pmatrix} \end{aligned} \tag{3.15}$$

$$\begin{aligned} \frac{\partial J(n)}{\partial \mathbf{w}} &= 4 \cdot e(n) \cdot [0.5 \cdot z(n) \cdot \mathbf{z}^*(n-1) + z_R(n) \cdot ((\mathbf{W_{RR}} \cdot \mathbf{w_R} - \mathbf{W_{IR}} \cdot \mathbf{w_I}) + \\ &\quad z_I(n) \cdot (\mathbf{W_{RR}} \cdot \mathbf{w_I} + \mathbf{W_{IR}} \cdot \mathbf{w_R}) + j(z_R(n) \cdot ((\mathbf{W_{RI}} \cdot \mathbf{w_R} - \mathbf{W_{II}} \cdot \mathbf{w_I}) + \\ &\quad z_I(n) \cdot (\mathbf{W_{RI}} \cdot \mathbf{w_I} + \mathbf{W_{II}} \cdot \mathbf{w_R}))))] \\ &= 4 \cdot e(n) \cdot [0.5 \cdot z(n) \cdot \mathbf{z}^*(n-1) + \begin{pmatrix} \mathbf{w_R} & -\mathbf{w_I} \\ j\mathbf{w_I} & j\mathbf{w_R} \end{pmatrix} \begin{pmatrix} \mathbf{W_{RR}} & -\mathbf{W_{RI}} \\ \mathbf{W_{IR}} & \mathbf{W_{II}} \end{pmatrix} \cdot \begin{pmatrix} z_R(n) \\ z_I(n) \end{pmatrix}. \end{aligned} \tag{3.16}$$

From these definitions and from Equation (3.2) we can derive the recursion for the gradients as:

$$\begin{aligned} \mathbf{C_{RR}}(n) \cdot \mathbf{f} &= 0.5 \cdot \mathbf{y_R}(n) + (\mathbf{C_{RR}}(n-1) \cdot \mathbf{w_R} - \mathbf{C_{IR}}(n-1) \cdot \mathbf{w_I}) \\ \mathbf{C_{RI}}(n) \cdot \mathbf{f} &= -0.5 \cdot \mathbf{y_I}(n) + (\mathbf{C_{RI}}(n-1) \cdot \mathbf{w_R} - \mathbf{C_{II}}(n-1) \cdot \mathbf{w_I}) \\ \mathbf{C_{IR}}(n) \cdot \mathbf{f} &= 0.5 \cdot \mathbf{y_I}(n) + (\mathbf{C_{IR}}(n-1) \cdot \mathbf{w_R} + \mathbf{C_{RR}}(n-1) \cdot \mathbf{w_I}) \\ \mathbf{C_{II}}(n) \cdot \mathbf{f} &= 0.5 \cdot \mathbf{y_R}(n) + (\mathbf{C_{II}}(n-1) \cdot \mathbf{w_R} + \mathbf{C_{RI}}(n-1) \cdot \mathbf{w_I}) \\ \mathbf{W_{RR}}(n) \cdot \mathbf{f} &= 0.5 \cdot \mathbf{z_R}(n-1) + (\mathbf{W_{RR}}(n-1) \cdot \mathbf{w_R} - \mathbf{W_{IR}}(n-1) \cdot \mathbf{w_I}) \\ \mathbf{W_{RI}}(n) \cdot \mathbf{f} &= -0.5 \cdot \mathbf{z_I}(n-1) + (\mathbf{W_{RI}}(n-1) \cdot \mathbf{w_R} - \mathbf{W_{II}}(n-1) \cdot \mathbf{w_I}) \\ \mathbf{W_{IR}}(n) \cdot \mathbf{f} &= 0.5 \cdot \mathbf{z_I}(n-1) + (\mathbf{W_{IR}}(n-1) \cdot \mathbf{w_R} + \mathbf{W_{RR}}(n-1) \cdot \mathbf{w_I}) \end{aligned} \tag{3.17}$$

Step 1:	Calculate the equalised signal according to (3.2)
Step 2:	Calculate the update of the derivatives according to (3.19) and (3.20)
Step 3:	Shift matrices $\mathbf{C}$ and $\mathbf{W}$ one column deleting the least recent column and replace with the column from the recursion of the previous step
Step 4:	Update the equaliser coefficients according to (3.5), (3.6) and (3.15), (3.16)

Table 3.1: The steps of the algorithm.

$$\mathbf{W_{II}}(n)\cdot\mathbf{f} \;=\; 0.5\cdot\mathbf{z_R}(n-1)+(\mathbf{W_{II}}(n-1)\cdot\mathbf{w_R}+\mathbf{W_{RI}}(n-1)\cdot\mathbf{w_I})\,, \tag{3.18}$$

where $\mathbf{f}=[1,0,0,\cdots,0]^T$ is a column vector used to extract the first column of the matrix, which it multiplies. The matrices $\mathbf{C}(n)$ and $\mathbf{W}(n)$ actually contain the N most recent columns of the derivative of the received and equalised signals respectively, with respect to the relevant equaliser tap-vector. Therefore, at every update, these columns are shifted so that the most obsolete one is replaced by the recently updated column from Equations (3.17) and (3.18). We can further streamline Equations (3.17) and (3.18) as:

$$\begin{pmatrix}\mathbf{C_{RR}} & \mathbf{C_{RI}}\\ \mathbf{C_{IR}} & \mathbf{C_{II}}\end{pmatrix}(n)\cdot\mathbf{f}=0.5\cdot\begin{pmatrix}\mathbf{y_R} & -\mathbf{y_I}\\ \mathbf{y_I} & \mathbf{y_R}\end{pmatrix}(n)+\begin{pmatrix}\mathbf{w_R} & -\mathbf{w_I}\\ \mathbf{w_I} & \mathbf{w_R}\end{pmatrix}\cdot\begin{pmatrix}\mathbf{C_{RR}} & \mathbf{C_{RI}}\\ \mathbf{C_{IR}} & \mathbf{C_{II}}\end{pmatrix}(n-1) \tag{3.19}$$

$$\begin{pmatrix}\mathbf{W_{RR}} & \mathbf{W_{RI}}\\ \mathbf{W_{IR}} & \mathbf{W_{II}}\end{pmatrix}(n)\cdot\mathbf{f}=0.5\cdot\begin{pmatrix}\mathbf{z_R} & -\mathbf{z_I}\\ \mathbf{z_I} & \mathbf{z_R}\end{pmatrix}(n-1)+\begin{pmatrix}\mathbf{w_R} & -\mathbf{w_I}\\ \mathbf{w_I} & \mathbf{w_R}\end{pmatrix}\cdot\begin{pmatrix}\mathbf{W_{RR}} & \mathbf{W_{RI}}\\ \mathbf{W_{IR}} & \mathbf{W_{II}}\end{pmatrix}(n-1). \tag{3.20}$$

The equaliser tap-update algorithm is therefore given by:

$$\mathbf{c}^{(n)} \;=\; \mathbf{c}^{(n-1)}-\lambda\cdot e(n)\cdot(z^*(n)\cdot\bar{\mathbf{C}}(n-1)\cdot\mathbf{w}+z(n)\,(\mathbf{y}^*(n)+\mathbf{C}(n-1)\cdot\mathbf{w}^*)) \tag{3.21}$$

$$\mathbf{w}^{(n)} \;=\; \mathbf{w}^{(n-1)}-\lambda\cdot e(n)\cdot(z^*(n)\cdot\bar{\mathbf{W}}(n-1)\cdot\mathbf{w}+z(n)\,(\mathbf{z}^*(n-1)+\mathbf{W}(n-1)\cdot\mathbf{w}^*))\,, \tag{3.22}$$

The proposed equaliser coefficients-update algorithm is given in the form of a set of steps, which are summarised in Table 3.1.

3.3 Convergence Issues

In this section we discuss the convergence properties of the DFE–CMA algorithm. The stationary points of the algorithm are identified and the similarities with the CMA are highlighted. We commence the analysis by considering the total transfer function of the channel plus equaliser system denoted by $\mathbf{t}=[t_0,t_1,\cdots,t_K]^T$. Naturally, K may tend to infinity, but we will consider it to be finite for the moment. With this notation, the stationary points of the algorithm can be found by setting the

derivatives of the cost function with respect to the equaliser coefficients, which are given by Equations (3.7) and (3.8), to zero:

$$\frac{\partial J(n)}{\partial \mathbf{c}} = 2 \cdot e(n) \cdot [z^*(n) \cdot \frac{\partial \mathbf{z}^T(n-1)}{\partial \mathbf{c}} \cdot \mathbf{w} + z(n) \cdot (\mathbf{y}^*(n) + \frac{\partial \mathbf{z}^H(n-1)}{\partial \mathbf{c}} \cdot \mathbf{w}^*)] = \mathbf{0} \quad (3.23)$$

$$\frac{\partial J(n)}{\partial \mathbf{w}} = 2 \cdot e(n) \cdot [z^*(n) \cdot \frac{\partial \mathbf{z}^T(n-1)}{\partial \mathbf{w}} \cdot \mathbf{w} + z(n) \cdot (\mathbf{z}^*(n-1) + \frac{\partial \mathbf{z}^H(n-1)}{\partial \mathbf{w}} \cdot \mathbf{w}^*)] = \mathbf{0}. \quad (3.24)$$

We will assume that we can ignore the contributions of the terms $\frac{\partial \mathbf{z}^T(n-1)}{\partial \mathbf{w}}$, $\frac{\partial \mathbf{z}^H(n-1)}{\partial \mathbf{w}}$ and $\frac{\partial \mathbf{z}^T(n-1)}{\partial \mathbf{c}}$, $\frac{\partial \mathbf{z}^H(n-1)}{\partial \mathbf{c}}$ since they are relatively small compared to the received and the equalised signals, respectively. This is a valid assumption when the equaliser is close to convergence, i.e. in the vicinity of a desirable equilibrium, since in this case the equalised symbols are rather insensitive to the equaliser's coefficients. In the implementation of this algorithm these terms are not taken into account at all, which implies that the assumption of ignoring them is justified. However, we have no evidence as to whether these terms may actually create undesirable local minima for the equaliser. Under this assumption, Equations (3.23) and (3.24) become:

$$e(n) \cdot z(n) \cdot \mathbf{y}^*(n) = \mathbf{0} \quad (3.25)$$

$$e(n) \cdot z(n) \cdot \mathbf{z}^*(n-1) = \mathbf{0}. \quad (3.26)$$

The first of these equations is the same as the relevant equations for the CMA, namely Equation (1.117), which was originally studied in [22, 27] and [102] in the context of a modified CMA version. The stationary points defined by Equation (3.25) are as follows:

- Local minima are:
  - the points of the form: $\mathbf{t} = [0, \cdots, 0, e^{j \cdot \phi}, 0, \cdots, 0]^T$ and
  - the channel–dependent local minima exhibited by equalisers having a finite length.
- Saddle points are:
  - the point $\mathbf{t} = \mathbf{0}$ and
  - the points of the form $\mathbf{t} = [0, \cdots, 0, e^{j \cdot \phi_1}, \cdots e^{j \cdot \phi_M}, 0, \cdots, 0]^T$.

Note furthermore that for a point to be a local minimum of the DFE–CMA algorithm in terms of the cost function of Equation (3.4) it has to be a local minimum of both Equations (3.25) and (3.26). We will now consider Equation (3.26) to investigate the associated stationary points. After some averaging calculations over all possible transmitted symbols $\mathbf{a}(n)$, assuming that they are independent

identically distributed (i.i.d), this Equation (3.26) reduces to the following system:

$$\begin{pmatrix} t_0^* & t_1^* & t_2^* & \cdots & t_{G-1}^* \\ 0 & t_0^* & t_1^* & \cdots & t_{G-2}^* \\ 0 & 0 & t_0^* & \cdots & t_{G-3}^* \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & t_0^* \end{pmatrix} \begin{pmatrix} f(t_1) \\ f(t_2) \\ f(t_3) \\ \vdots \\ f(t_G) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \tag{3.27}$$

where $f(t_i)$ is defined as $f(t_i) = (\mu_4(|t_i|^2 - 1) + 2\mu_2^2 \sum_{l \neq i} |t_l|^2) \cdot t_i$, where again we have defined $\mu_i = E[|a(n)|^i]$, with $a(n)$ representing the complex input QAM symbols. By examining Equation (3.27) we readily see that it accepts the trivial solution $\mathbf{t} = \mathbf{0}$, as well as solutions, which obey $t_i = 0$ for $i < \delta$ and $t_i \neq 0$, $f(t_i) = 0$ for $i \geq \delta$. This solution gives the following system of equations for $i \geq \delta$:

$$\mu_4(|t_i|^2 - 1) + 2\mu_2^2 \sum_{l \neq i} |t_l|^2 = 0, \quad i = \delta, \cdots, G. \tag{3.28}$$

Equation (3.28) can be rewritten as:

$$\begin{pmatrix} \mu_4 & 2\mu_2^2 & \cdots & 2\mu_2^2 \\ 2\mu_2^2 & \mu_4 & \cdots & 2\mu_2^2 \\ \vdots & \vdots & \ddots & \vdots \\ 2\mu_2^2 & 2\mu_2^2 & \cdots & \mu_4 \end{pmatrix} \begin{pmatrix} |t_\delta|^2 \\ |t_{\delta+1}|^2 \\ \vdots \\ |t_G|^2 \end{pmatrix} = \begin{pmatrix} \mu_4 \\ \mu_4 \\ \vdots \\ \mu_4 \end{pmatrix}. \tag{3.29}$$

The solution of Equation (3.29) is given by:

$$|t_i|^2 = P^2 = \frac{\mu_4}{\mu_4 + 2\mu_2^2(G - \delta)}, \quad i = \delta, \cdots, G, \tag{3.30}$$

which corresponds exactly to the CMA's stationary points, as it was shown in [27]. We have found that the stationary points can assume one of the following forms:

- $[0, \cdots, 0, e^{j \cdot \phi_\delta}, \cdots, e^{j \cdot \phi_{G-\delta}}, 0, \cdots]^T$, where $G - \delta + 1$ defines the number of non-zero vector entries,
- $[0, \cdots, 0]^T$,

which are similar to the CMA's stationary points, having the sole difference that in this case the CIR does not play a multiplicative role. Therefore the requirement of an infinite number of equaliser taps is not necessary in this case. At this stage we have to ascertain which of the stationary points are minima, saddle points and maxima. Explicitly, we only want the points of the form $[0, \cdots, 0, e^{j \cdot \phi}, 0, \cdots, 0]^T$ to be local minima and all the other points to be unstable equilibria. It is easy to see that this is true, since the cost function of this algorithm is the same as the cost function of the CMA. Foschini [27] has

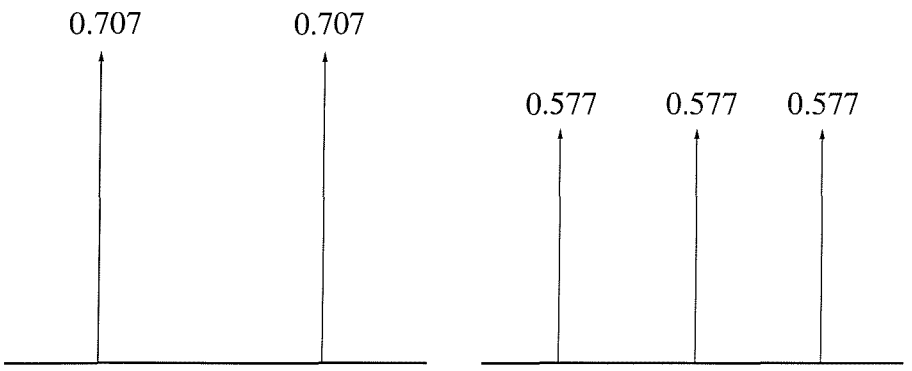


Figure 3.2: Two and three–path channel models used in the performance comparison.

studied the stationary points of this cost function, which are the same for the proposed DFE–CMA as well. Foschini also proved that only the tap vector having a single non-zero element - i.e. the vectors of the form $[0, \cdots, 0, e^{j\cdot\phi_0}, 0, \cdots, 0]^T$- constitute a minimum and all the others - i.e. the vectors of the form $[0, \cdots, 0]^T$ as well as $[0, \cdots, 0, e^{j\cdot\phi_\delta}, \cdots, e^{j\cdot\phi_{G-\delta}}]^T$, $G \neq \delta$ - are saddle points. This also proves that the DFE–CMA cost function, which is the same, also assumes only the above non-zero tap vector as a minimum.

What we can observe from the study of this equaliser's convergence is that under the assumption that the terms of Equations (3.19) and (3.20) are not used in the equaliser update process, the equaliser does not converge to any length–dependent local minima, unlike the CMA. This is due to the fact that the feedback introduced to its input is constituted by the convolution of the total system's impulse response with the input symbols, but it does not incorporate the filtering effect of the channel, as evidenced by Equation (3.26), where the vectorial term multiplying the error $e(n)$ is now $\mathbf{z}^*(n-1)$, instead of the term $\mathbf{y}^*(n)$ of Equation (3.25). As we have already observed in Section 1.3.7, it is exactly this term which gives rise to the length–dependent undesirable equilibria, inherent in the CMA. Having discussed the convergence of the DFE–CMA, we will now characterise its performance.

3.4 Performance Results

In this section the proposed algorithm is benchmarked against the classic CMA [4], demonstrating the improved performance of the DFE–CMA, when the channel is difficult to equalise, since the CIR contains two or three equal paths. In Figure 3.2 the channel models used are shown. In Figure 3.3 the equaliser's Mean Squared Error (MSE) learning curves are plotted for Quadrature Phase Shift Keying (QPSK) using the two-path channel model of Figure 3.2 for a Signal-to-Noise-Ratio (SNR) of $30dB$. Observe the substantially improved performance of the DFE–CMA, explained by the fact that a reduced number of taps is needed, thus inflicting less convolutional noise. The step–size parameter

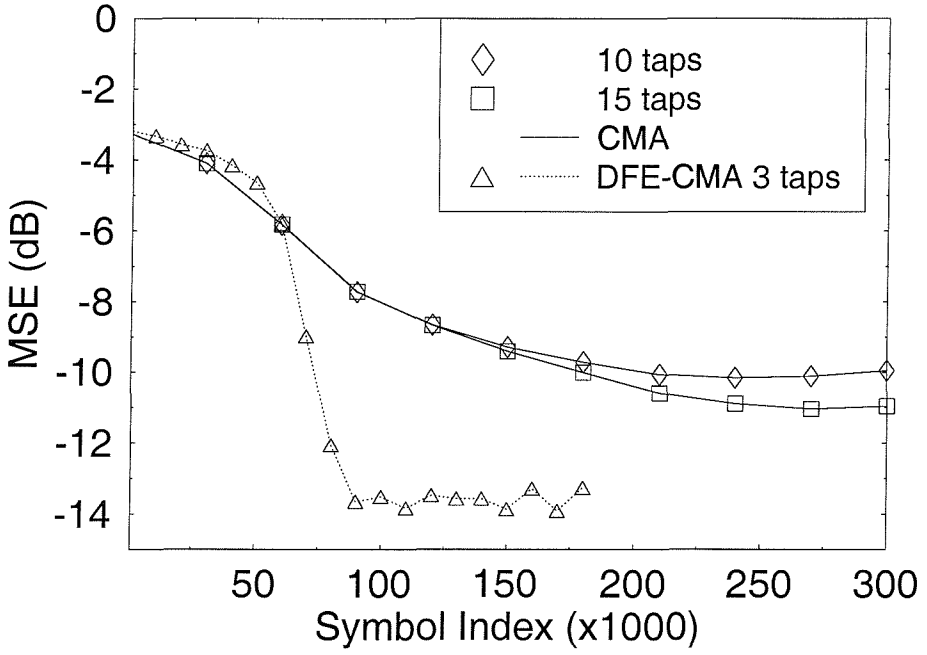


Figure 3.3: The MSE learning curves for QPSK, using the two-path model of Figure 3.2. The DFE–CMA has two feedback taps and one feedforward tap.

λ is $5\text{x}10^{-4}$. In Figure 3.4 the same performance curves are given for 16–QAM and for the three-path channel model of Figure 3.2. Again, we observe better performance for the DFE–CMA, combined in this case with slower convergence. It is expected that the convergence of the DFE–CMA will be faster than that of the CMA, as a consequence of its reduced number of taps. On the other hand, since a DFE is typically less stable than its open–loop counterpart in the sense that it can diverge more easily from the desirable point of equilibrium, it may require more time to move towards the point of equilibrium. We can also observe in Figure 3.4 that the MSE of the DFE–CMA is not monotonically decreasing after convergence. This is due to the statistical variation of the algorithm's operation due to the statistically different nature of the data processed. The MSE curve will fluctuate around the mean value after the algorithm has converged. Following the above arguments, what we actually see in Figure 3.4 and also in Figure 3.3 is that after an initial period of slow or no convergence the DFE "switches" to fast convergence mode and then converges in a "waterfall" fashion. On the other hand, the conventional CMA would require more taps to give a similar performance, and hence the convolutional noise would increase the MSE. In Table 3.2 we give a complexity estimate for the update of the DFE–CMA and the classic CMA, based on the number of real additions and multiplications required at each symbol interval. In this table M is the number of feedback taps, while N is the number of feedforward taps.

It is clear from Table 3.2 that in general the DFE–CMA is more complex, because its complexity depends on M^2, even when we use only one feedforward tap. However, in the situations of interest

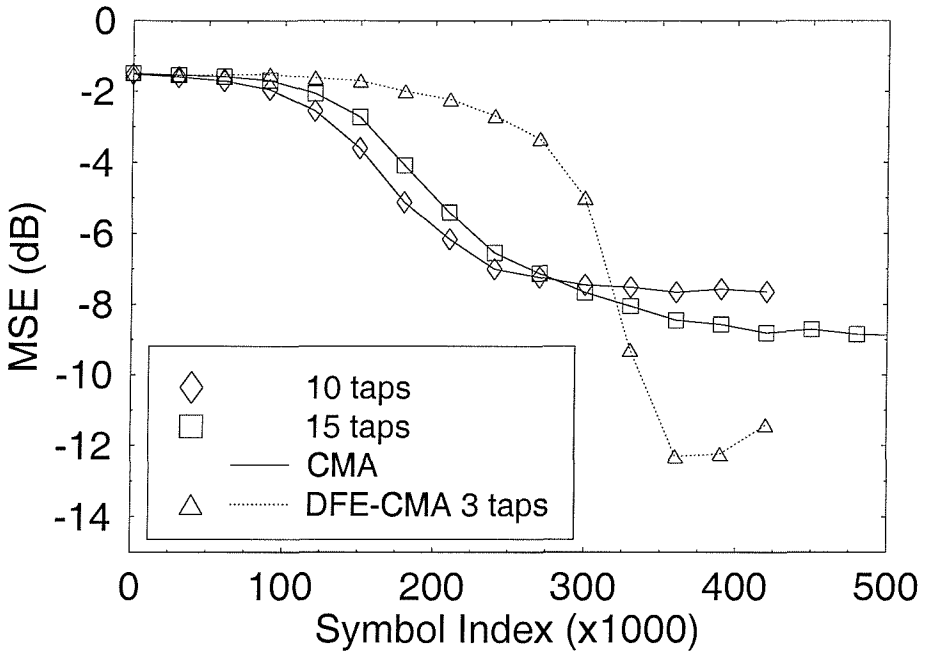


Figure 3.4: The MSE learning curves for 16–QAM, using the three-path CIR model of Figure 3.2. The DFE–CMA has two feedback taps and one feedforward tap.

Algorithm	Additions	Multiplications
DFE–CMA	$16M(M+N)+31M+15N-16$	$8M(M+N)+29M+13N$
DFE–CMA ($N=1$)	$16M^2+47M+15$	$8M^2+37M+21$
CMA	$4N+2$	$4N+4$

Table 3.2: Complexity estimation of the DFE–CMA and the CMA.

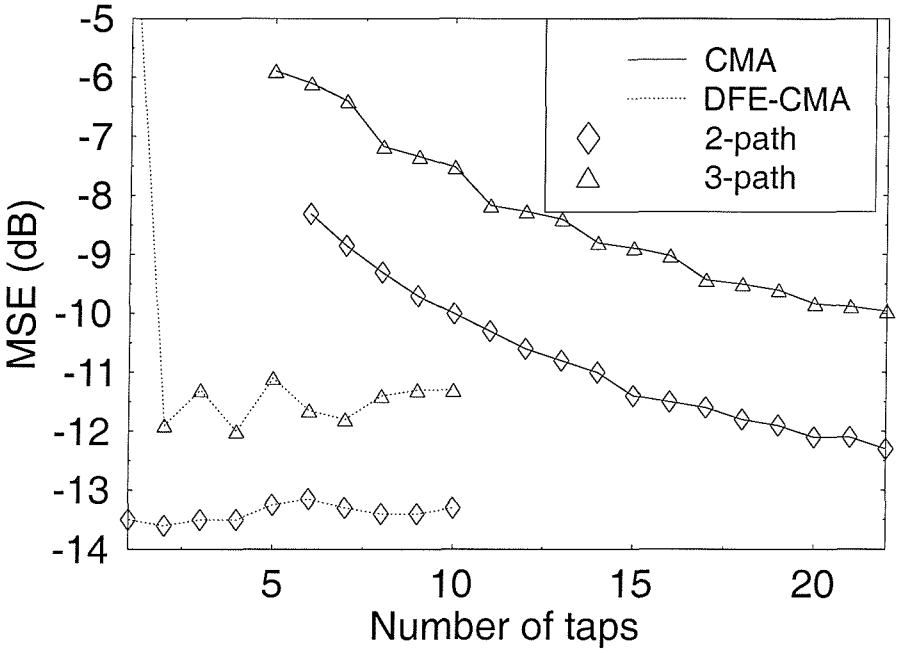


Figure 3.5: The minimum MSE as a function of the number of equaliser taps for 16–QAM using the two- and three-path channel models of Figure 3.2 at an SNR of $30dB$. The DFE–CMA has two feedback taps and one feedforward tap.

the number of feedback taps required is determined by the channels's delay spread in terms of the number of symbol intervals and it is rather small, while for these channels a high-order CMA based equaliser would be required. As an example, in Figure 3.5 we plot the minimum MSE as a function of the number of equaliser taps at an SNR of $30dB$. We observe that the DFE–CMA MSE function has a minimum, depending on the channel model and also on the SNR. In this graph we observe the expected minimum introduced as a result of two factors:

- The additive channel noise, implying that the MSE cannot be less than a certain threshold and
- The convolutional noise, which exists because of:
  - the finite equaliser length and
  - due to the fact that this is a blind equaliser and it does not converge with infinite precision, especially in the presence of additive channel noise.

Similar minima exist in the CMA MSE function at a larger number of taps. It is clear that in the two-path channel scenario the DFE–CMA requires only one feedback tap, while the CMA requires at least 20 taps. In Table 3.2 we also observe that the corresponding complexities are 78 real additions and 64 real multiplications for the DFE–CMA and 82 real additions and 84 real multiplications for

the CMA. Therefore, in this case the CMA is more complex while its performance is inferior to that of the proposed algorithm.

3.5 Summary

In this chapter a novel blind equaliser was presented and studied, extending the conventional CMA to the DFE–CMA scheme. This equaliser provides a solution to the problem of reversing the effects of a nonminimum phase channel, such as the channels described by a Z–transform of having zeros on the unit circle. The impulse response of the equaliser required to by such channels is long and thus the equaliser requires a high number of taps. In the case of CMA–based blind adaptation, due to the fact that during the equaliser's initialisation such channels inflict severe ISI, often slow and inaccurate convergence is encountered. The benefit of this equaliser in comparison to the conventional CMA is that its feedback section provides implicit training information in the absence of transmission errors. In this case there is no need for a high number of taps. Instead, the number of equaliser taps required is equal to the number of CIR taps. For transmissions over other types of channels similar performance ia attained to that of the CMA, but at a potentially higher complexity. This complexity increase is typically on the order of L, L being the number of CIR taps. Finally, when the approximation of neglecting the terms of Equations (3.19) and (3.20) from the calculation of the update terms in Equation (3.22) is used, then this equaliser, unlike the CMA, is incapable of converging to any channel–(otherwise referred to as "length–") dependent undesirable local minima, due to its feedback section. We have to note here, however, that this equaliser, as a DFE, is more prone to instabilities, than the conventional CMA, in particular when the initialisation is far from the desired point of equilibrium and the value of the step–size is not sufficiently low.

Finally, the application of this equaliser in various channel environments requires optimisation with respect to the number of feedforward and feedback taps. Further research related to this technique has to consider this optimisation together with the possibility of blindly detecting the type of the CIR, in order to automatically "decide" on the required tap–vector lengths.

Having presented the DFE–CMA equaliser in this chapter, we will now focus our attention on presenting a novel class of blind equalisers, combining PSP based channel estimation and channel decoding in an iterative fashion.

Chapter 4

Combined Blind Equalisation and Channel Decoding: Turbo–PSP

4.1 Introduction

Blind equalisation has attracted significant research interests during recent years. The blind equaliser proposed in this treatise belongs to the class of sequence estimation techniques. It incorporates a Per–Survivor Processing (PSP) based equaliser [6], modified appropriately, in order to produce soft outputs as in [135, 182, 183] and it involves channel coding not only for protecting the transmitted data from the additive noise induced by the channel, but also for assisting the PSP equaliser during convergence by utilising a feedback loop.

As we will see in Section 4.5, this novel blind equalisation technique exhibits good performance in terms of its output Bit Error Rate (BER). Explicitly, a BER comparable to that of a trained turbo equaliser is attained at the cost of a modest complexity increase.

This chapter is organised as follows. The communications system is characterised in Section 4.2, while in Section 4.3 the proposed turbo–PSP equaliser is described. Specifically, in Sections 4.3.1 – 4.3.2 its separate modules are discussed, while in Section 4.4 the associated phase ambiguity problem is addressed. Finally, in Section 4.5 performance results are provided for both static and fading dispersive channels.

4.2 System Description

The communications system under consideration is shown in Figure 4.1. The information bits are encoded by a convolutional encoder and then punctured in order to produce the necessary coding

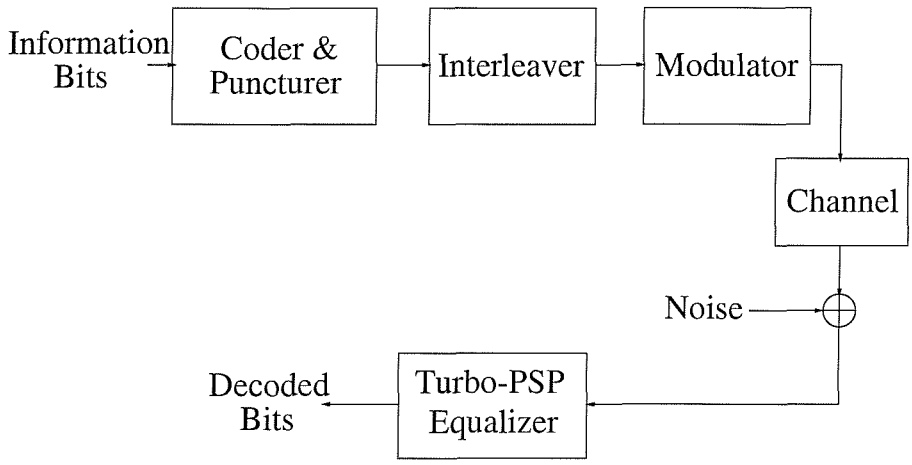


Figure 4.1: Equalised communications system

rate. The coded bits are then interleaved in order to disperse the channel's bursty errors as well as to enhance the turbo–equaliser's performance and then mapped to the QAM constellation symbols $a(n)$. These symbols are convolved with the Channel's Impulse Response (CIR) h_i and then the channel noise $e(n)$ is added, yielding the received symbols $y(n)$ as:

$$y(n) = \sum_{i=-L_1}^{L_2} h_i \cdot a(n-i) + e(n) \tag{4.1}$$

The restoration of the original information bits is performed by the turbo–PSP equaliser. This equaliser performs two basic functions: blind channel equalisation and channel decoding, in an iterative fashion. We will describe the operation of the turbo–PSP equaliser in the next section.

4.3 Turbo–PSP Equaliser Description

The turbo–PSP equaliser performs joint channel equalisation and channel decoding using the schematic shown in Figure 4.2. In this figure the symbol Π denotes the interleaver. The system consists of a PSP [6] equaliser, forwarding the so–called *LogLikelihood Ratio* values LLR_{eq} to the subsequent de–interleaver, which are defined as:

$$LLR(Bit) = ln\left[\frac{Prob(Bit=1)}{Prob(Bit=0)}\right]. \tag{4.2}$$

The operational principle of the system is as follows. Firstly, the PSP equaliser attempts to remove the Intersymbol Interference (ISI) from the received signal and at the same time performs demodulation, providing the bit LLR values, which are input to the channel decoder. Subsequently, the channel decoder provides better soft estimates of the PSP equaliser's output, which is then used as

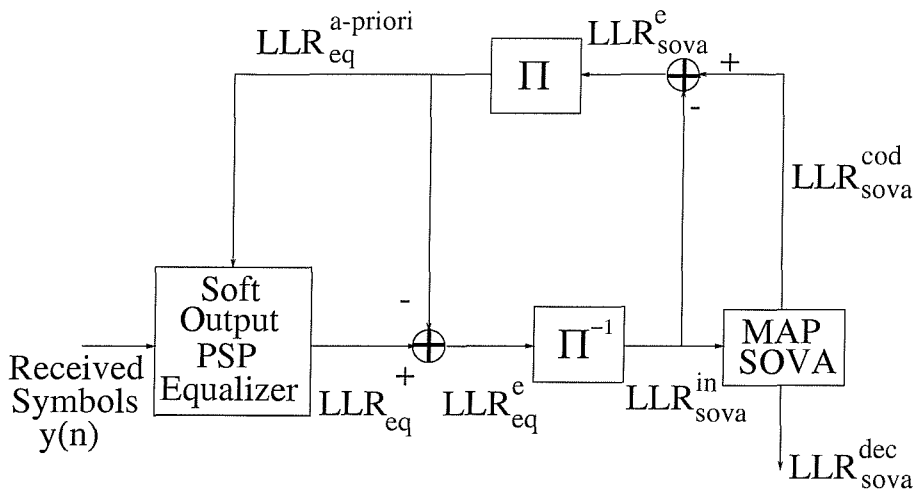


Figure 4.2: The turbo–PSP equaliser

"a–priori" information for the next iteration of the PSP equaliser. Following a number of iterations, the impairments of the channel are gradually removed from the signal.

Let us now discuss the turbo–PSP equaliser in more detail. The equaliser's a–priori LLR values, $LLR_{eq}^{a-priori}$ are used as a–priori knowledge, assisting the module in providing improved confidence values at its output. These values stem from the channel decoded information, which are then substracted from the equaliser's output LLR_{eq} in order to provide the input LLR_{eq}^{e} of the subsequent de–interleaver. This subtraction takes place so that in the next iteration the consequent channel decoder takes into account only the equaliser's contribution, but not the equaliser's a–priori values, which had been provided by the channel decoder itself. This concept is similar to that used in trained turbo equalisation [182], although the lack of training information requires the replacement of the MAP equaliser by a PSP equaliser. Following the PSP–based equalisation, the de–interleaver rearranges the LLR_{eq}^{e} values according to the interleaving algorithm used. Following the de–interleaver, the Soft Output Viterbi Algorithm (SOVA) performs channel decoding based on the *Maximum A–Posteriori (MAP)* criterion and provides the LLR_{sova}^{cod} as well as LLR_{sova}^{dec} values for its coded and decoded outputs, respectively. The coded LLR values are again interleaved, so as to align them with the channel output symbols. The resultant LLR values $LLR_{eq}^{a-priori}$ are then fed back to the PSP equaliser as a–priori values.

Having described the functions of the component modules of the turbo–PSP equaliser in Figure 4.2, we will now proceed to highlighting the operation of these components. We commence with the most important module, namely the PSP equaliser.

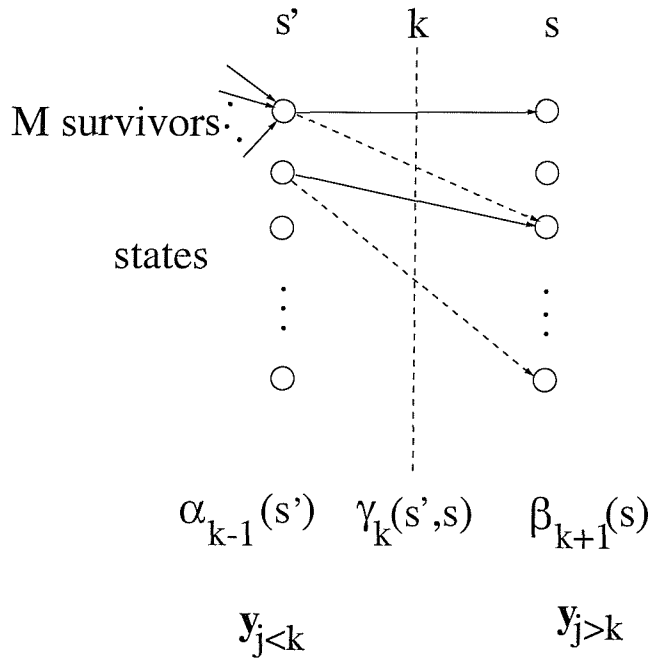


Figure 4.3: State transitions from time $k-1$ to k.

4.3.1 The PSP equaliser

Per–survivor processing was initially proposed by Seshadri in [6] in 1991. Seshadri modified the Viterbi algorithm [34], in order to render it applicable to the scenario where no channel information is available. This was achieved by considering that each surviving sequence in the trellis carries its own channel estimation, which is updated at every symbol instant. Additionally, Seshadri proposed the use of the M–algorithm [140] in order to invoke several surviving estimates and to counteract the possibility of a CIR initialisation drastically different from the actual CIR. More explicitly, when this algorithm is utilised, M surviving sequences are kept at every stage of the trellis search for each state. This approach makes the estimation process less vulnerable to the initial CIR estimate. Clearly, when M is equal to the number of states this is the special case which corresponds to the classic Viterbi Algorithm (VA). This algorithm was termed Per–Survivor Processing (PSP) by Polydoros *et al.* [23]. A novel variant of this algorithm is invoked in the context of the turbo–PSP equaliser, in order to provide soft outputs. This is achieved [183] by considering the following equation:

$$LLR_{eq}(x_k) = ln\left[\frac{Prob(x_k = +1|\mathbf{y},\mathbf{d})}{Prob(x_k = -1|\mathbf{y},\mathbf{d}}\right], \tag{4.3}$$

where $\mathbf{y}$ represents the received symbol vector and $\mathbf{d}$ represents the a–priori values of the equaliser, generated by the channel decoder's SOVA. In order to estimate the above probabilities, following the approach of [183], we consider the state transitions from time $k-1$ to k and then to $k+1$, as seen in Figure 4.3. As we can see from Figure 4.3, the probability $Prob(x_k = +1|\mathbf{y},\mathbf{d})$ can be viewed as that

of an event which can take place only if three separate events coincide. Thus, this probability can be expressed as the product of three probabilities. The first one is the probability $\alpha_{k-1}(s')$ of being in state s' at time $k-1$, given the received symbol vector $\mathbf{y_{j<k}}$ for time $i < k$, while the second one is the probability $\gamma_k(s', s)$ of moving to state s at time k, given that at time $k-1$ we were at state s' and that the received symbol vector $\mathbf{y}$. Finally, the third associated probability is that of being in state s at time k, namely $\beta_k(s)$, given the received symbol vector $\mathbf{y_{j>k}}$ for time $i > k$. With these considerations, the LLR of Equation (4.3) becomes:

$$LLR(x_k) = ln\left[\frac{\sum_{s'\to s, x_k=+1} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}{\sum_{s'\to s, x_k=-1} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}\right]. \tag{4.4}$$

The calculation of the values of α and β can be performed recursively, using the values of γ, according to the relationships [183]:

$$\alpha_k(s) = \sum_{s'} \gamma_k(s', s) \cdot \alpha_{k-1}(s') \tag{4.5}$$

$$\beta_{k-1}(s') = \sum_{s} \gamma_k(s', s) \cdot \beta_k(s). \tag{4.6}$$

Initial values have to be assumed for the recursive calculation of α and β in Equations (4.5) and (4.6). To set these values we can set the encoder to the all–zero state at both the beginning and end of each transmission frame. In our case, the serially concatanated encoder is considered to be the channel, and this initialisation implies transmitting "tailing" symbols, which correspond to all–zero bits at both the beginning and at the end of the frame. The number of these tailing symbols has to match the total delay spread of the CIR. This can be viewed as training, as it suggests transmitting information, which is known a–priori to the receiver. Nonetheless, tailing bits are required in order to provide initialisation and they are typically part of trained systems as well. In this case, the initialisation takes place as:

$$\alpha_0(s) = 1 \tag{4.7}$$

$$\alpha_k(s) = 0, \quad k \neq 0 \tag{4.8}$$

$$\tag{4.9}$$

$$\beta_K(s) = 1 \tag{4.10}$$

$$\beta_k(s) = 0, \quad k \neq K, \tag{4.11}$$

where K is the last symbol interval of the frame. It can easily be observed that at the beginning and end of the frame the initialisation allows any state to exist. This is because the PSP equaliser can converge to any phase–shifted solution and in this case the initial and final states are different from the zero state. Phase–shifted solutions are not encouraged, but they can always exist and this

initialisation protects the equaliser in any case. Finally, the values of γ can be obtained during the per–survivor processing, according to the following equation:

$$\gamma_k(s^{'}, s) = F_1(k, s^{'}, s) \cdot F_2(k, s^{'}, s) \tag{4.12}$$

$$F_1(k, s^{'}, s) = exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{M}|y(k) - \hat{y}_k^{(i)}(s^{'}, s)|^2\right) \tag{4.13}$$

$$F_2(k, s^{'}, s) = exp\left(\frac{1}{2}\sum_{i=1}^{Q}\hat{x}_i \cdot L(x_i)\right), \tag{4.14}$$

where σ^2 is the noise variance, the transmitted signal energy is assumed to be normalised to unity, $L(x_i)$ denotes the equaliser's a–priori values $LLR_{eq}^{a-priori}$ seen in Figure 4.2, $\hat{x}_i$ is the estimated input bit value associated with the transition from state $s^{'}$ to state s (it can be either -1 or $+1$), Q is the number of bits per symbol in the QAM constellation, $y(k)$ is the received symbol at time k, $\hat{y}_k^{(i)}(s^{'}, s)$ is the estimated received symbol at time k if the previous state was $s^{'}$, the survivor index was i and the current state is s. Finally, M is the number of survivors per state. This equation is an extension of the associated equation for the trained turbo equaliser, for which the channel estimate is global [183].

Finally, the CIR estimate update, which takes place at every symbol instant, can be performed by using the Least Mean Squares (LMS) [6] or the Recursive Least Squares (RLS) [23, 141] algorithm, where the latter provides fast convergence in exchange for a higher complexity. We opted for the LMS algorithm, according to which the CIR vector $\hat{\mathbf{h}}^{(n)}$ at time n is updated as:

$$\hat{\mathbf{h}}^{(n)} = \hat{\mathbf{h}}^{(n-1)} + \lambda \cdot \hat{\mathbf{a}}^*(n) \cdot \left(y(n) - \hat{\mathbf{h}}^{(n-1)} \cdot \hat{\mathbf{a}}(n)\right), \tag{4.15}$$

where λ is the step–size controlling the speed and the accuracy of convergence, $\hat{\mathbf{a}}(n)$ is the estimated transmitted symbol vector at the state concerned, $y(n)$ is the received symbol at symbol instant n and * denotes the complex conjugate.

The complexity of this algorithm depends directly on the number of states to be processed. This number is a function of $N^{L_1+L_2}$, where N is the number of symbols in the QAM constellation and $L_1 + L_2 + 1$ is the number of channel paths. The exponential makes the algorithm computationally demanding for channels with long impulse responses. However, the M–algorithm approximation can deliver similar performance for such channels, by retaining a small number of states.

4.3.2 The Soft Output Viterbi Algorithm

The SOVA of the channel decoder operates similarly to the corresponding module of the trained turbo equaliser [135, 182, 183]. Bearing in mind the above point, it is readily seen that the α and β values can be calculated according to the trellis state transitions, described by the same recursive formulae, namely Equations (4.5) and (4.6) as for the PSP equaliser. The difference is in the calculation of the γ values, which are now given by:

$$\gamma_k(s^{'}, s) = exp\left(\frac{1}{2} \cdot \sum_{i=1}^{n} \hat{x}_i(k) L(x_i(k))\right), \tag{4.16}$$

where n is the number of coded bits output by the convolutional encoder for each k–bit input word, $\hat{x}_i(k)$ is the estimated input bit at instant k during the transition from state $s^{'}$ to state s, which can be either $+1$ or -1, and $L(x_i(k))$ is its LLR value, which is the input of the SOVA. Finally, as for the case of the PSP equaliser, the LLR of the decoded output bit u_k at instant k is given by:

$$LLR(u_k) = ln\left[\frac{\sum_{s^{'}\to s, u_k=+1} \alpha_{k-1}(s^{'}) \cdot \gamma_k(s^{'}, s) \cdot \beta_k(s)}{\sum_{s^{'}\to s, u_k=-1} \alpha_{k-1}(s^{'}) \cdot \gamma_k(s^{'}, s) \cdot \beta_k(s)}\right]. \tag{4.17}$$

The LLR values of the coded bits also have to be fed back to the PSP equaliser, improved by the decoder, which are given by:

$$LLR(x_k) = ln\left[\frac{\sum_{s^{'}\to s, x_k=+1} \alpha_{k-1}(s^{'}) \cdot \gamma_k(s^{'}, s) \cdot \beta_k(s)}{\sum_{s^{'}\to s, x_k=-1} \alpha_{k-1}(s^{'}) \cdot \gamma_k(s^{'}, s) \cdot \beta_k(s)}\right]. \tag{4.18}$$

Another module used in our system is the puncturer, which is capable of providing different code rates, while keeping the convolutional coding rate fixed. The puncturer selects a number of bits from the output block of the convolutional encoder, which is less than the block length. By doing so, some bits are actually discarded and this results in a decreased code rate. The inverse procedure is carried out by the depuncturer, which inserts zero LLRs (i.e. bits that are '1' with a probability of 0.5) in the locations, where bits have been discarded by the puncturer, implying that for these locations no information is available for the relevant bits. However, in this contribution no puncturing was used.

4.3.2.1 The Turbo–Coded Turbo–PSP

An extension of the turbo–PSP equaliser employing turbo coding [135] instead of convolutional has also been considered. In a turbo encoder the convolutional encoder is replaced by two half–rate convolutional encoders, as shown in Figure 4.4. Half of the input bits, namely the systematic bits,

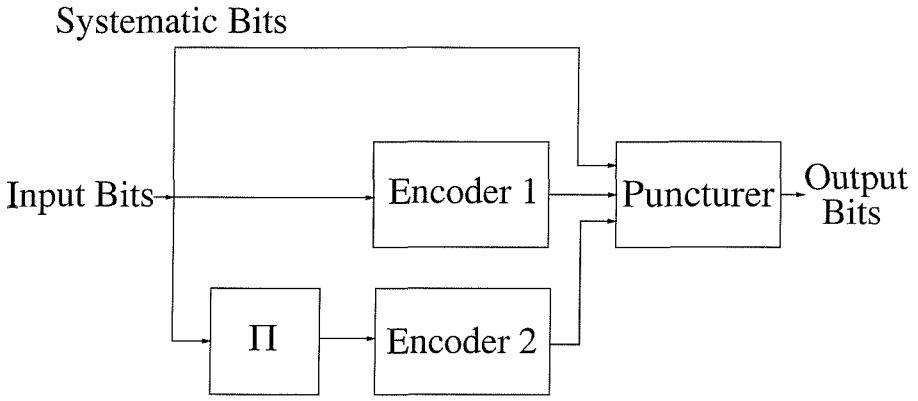


Figure 4.4: The turbo encoder

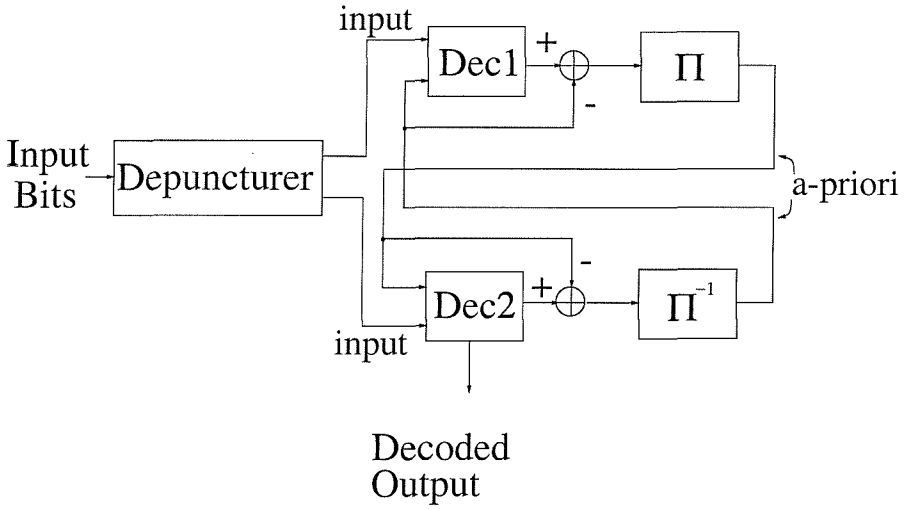


Figure 4.5: The turbo decoder

were directly fed to the output by the puncturer and the other half of the bits are encoded by both Encoder 1 and also by Encoder 2, in the latter case after interleaving. The output of the two encoders is multiplexed so that half of the bits of each encoder are passed to the turbo encoder's output. The turbo decoder is illustrated in Figure 4.5. The operation of this module is similar to that of the turbo equaliser, taking into account that in this case the equaliser module has been replaced by another convolutional decoder. Details of various turbo decoding algorithms can be found for example in [184].

Having described the most important components of the proposed turbo–PSP equaliser, we will now address the closely related issue of differential coding.

4.4 Differential Coding

Blind equalisers may converge to a tap setting, which is rotated in the complex plane, as compared to the ideal setting. Specifically, when QAM is employed, the symmetry of the modulation constellation implies that the blind equaliser may converge to four different settings, each corresponding to one of the four quadrants of the complex plane. A common solution to this problem inherent in any blind equaliser is the employment of differential encoding. Seshadri [6] observed this phenomenon for the case of a PSP equaliser. For the turbo–PSP equaliser, however, this solution cannot be used, since differential encoding and decoding of soft values would have to be performed. However, the differential encoding of soft values results in converging to zero soft values. This is due to the fact that, unlike differential decoding, during defferential encoding an erroneous decision propagates further than the next bit. The problem can be avoided by performing differential encoding in a block basis rather than in a bit basis. In other words, differential encoding can be performed by taking into account a whole block of bits in deciding on the value of a specific bit, rather than taking into account only the value of the next bit. This technique requires exponentially increased complexity and is therefore impractical. Thus, other techniques of overcoming this problem have to be used. We will describe an appropriate method below.

It can be observed that the equaliser typically converges to that specific setting, which it has been initialised closest to. If the initialisation is close to the correct setting, then the equaliser will converge and in the next iteration the equaliser will be fed with a–priori values, which will improve its new CIR estimate and its soft output values. By contrast, if the equaliser's initialisation is closer to a setting which is not the correct one, then it will start converging to this particular setting. However, the SOVA will be fed with phase–rotated values and in the next iteration the equaliser's a–priori values will drive the equaliser to instability. A way of overcoming this problem is to shift the estimated CIR in phase, so that it matches the actual CIR as closely as possible. The problem that arises is that of estimating the phase shift. This can be achieved by initially invoking the PSP equaliser without iterations and transmitting a series of pilot symbols. As we will see in the next section, a few training pilot symbols can be transmitted, resulting in a minor decrease of the bandwidth. After the selection of the right coordinate quadrant, equalisation continues as before. The number of iterations is adjusted according to an estimate of the grade of convergence provided by the equaliser.

4.5 Performance Results

In this section we present performance results for the turbo PSP–equaliser described in Section 4.3. The results are based on computer simulations. The channels assumed are static or fading, having the

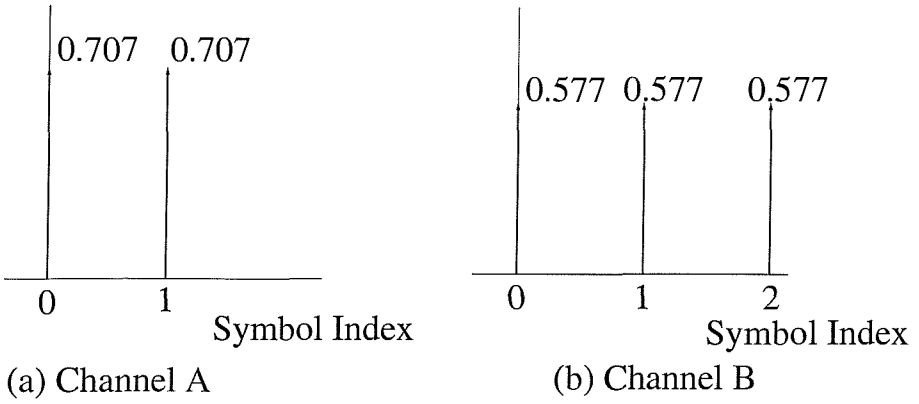


Figure 4.6: The CIRs used in our simulations

Frame length (bits)	174
Interleaver type	Bit
Interleaver block length	5x174
Carrier frequency	1900MHz
Symbol rate	2.6MBaud
Doppler speed	48Km/h
Code rate	1/2
Convolutional generator polynomial 1	17_{OCT}
Convolutional generator polynomial 2	13_{OCT}
Turbo convolutional generator polynomial 1	17_{OCT}
Turbo convolutional generator polynomial 2	17_{OCT}
Equaliser step–size	$5x10^{-3}$

Table 4.1: The turbo–PSP equaliser parameters used in the simulations.

CIRs given in Figure 4.6. Iteration or convergence control has been considered in two different ways. In Detector 1 (D1) the variance of the input LLR values of the de–interleaver was measured and then its slope was estimated by taking four samples into account. The LLR variance provides an estimate of our confidence in the LLR values, since the higher the absolute values of these LLRs, the more confident the decisions. The associated LLR variance is expected to increase iteration after iteration and reach its peak value, when the iterations can no longer improve the BER. This condition is detected by estimating the slope of the variance by utilising the least squares method and taking into account the last four values of the LLR variance in the calculation of the slope. According to D1 the iterations are curtailed when the slope becomes less than a threshold value, which was selected to be zero. According to our second detector (D2) the variance of the difference between the previous and current equaliser output LLRs is measured and compared against a fixed threshold. The iterations are curtailed, when the variance becomes less than the threshold. In our simulations D1 and D2 are evaluated both in terms of their performance and complexity reduction. The general turbo–PSP equaliser's parameters are given in Table 4.1. In Figure 4.7 the BER versus Bit SNR curves are given for both QPSK and

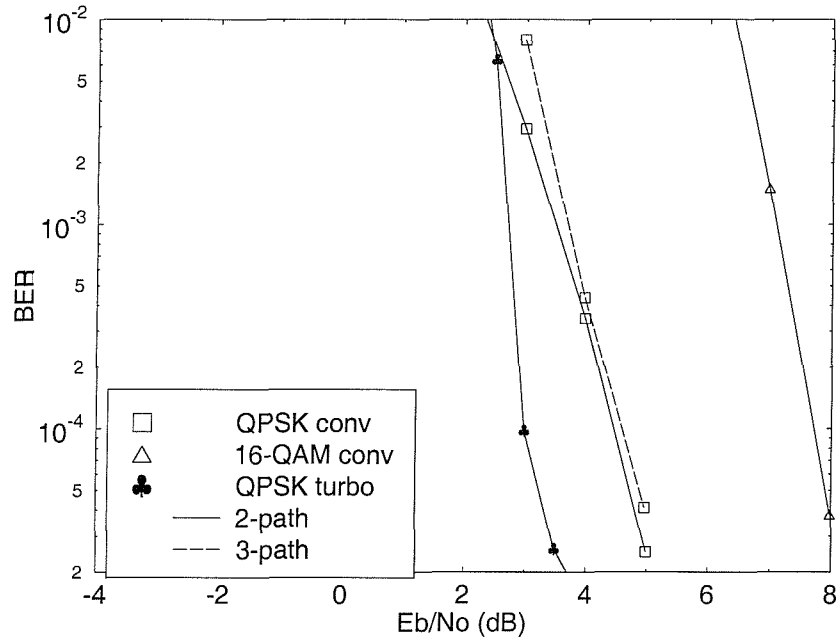


Figure 4.7: BER versus Bit SNR curves over static channels, exhibiting the impulse responses shown in Figure 4.6 and using the turbo–PSP equaliser parameters given in Table 4.1. The iteration control obeyed D1.

16–QAM and also for turbo–coded QPSK using 4 iterations, over the static channels of Figure 4.6. It is clear that using turbo coding instead of convolutional coding improves the performance, although it increases the complexity of the system. In this specific system, the complexity is eight times the complexity of the convolutional coded system but the number of iterations of the turbo equaliser also has to be taken into account. In this case the associated complexity was approximately the same for both convolutional and turbo coding.

The Bit SNR is defined as:

$$\text{Bit SNR} = \frac{E_a}{\text{IBPS} \cdot E_e}, \tag{4.19}$$

where E_a is the average QAM signal power, E_e is the average noise power and IBPS is the number of information bits per symbol, i.e. the number of convolutional coded bits per symbol. This definition assists in the realistic assessment of the benefits of using strong convolutional coding for protecting the data.

In Figure 4.8 a comparison is given between D1 and D2 for QPSK over the static channels of Figure 4.6(a). Finally, in Figure 4.9 we portrayed the associated average number of iterations for both detectors D1 and D2. Firstly, we demonstrated in Figure 4.8 that the BER is not affected by the choice of D1 or D2. This implies that both detection methods assign a sufficiently high number of iterations for correct decoding. The number of iterations required by D1 is lower than that of D2 at low SNRs and higher than that of D2 at high SNRs. This can be readily interpreted, if we consider

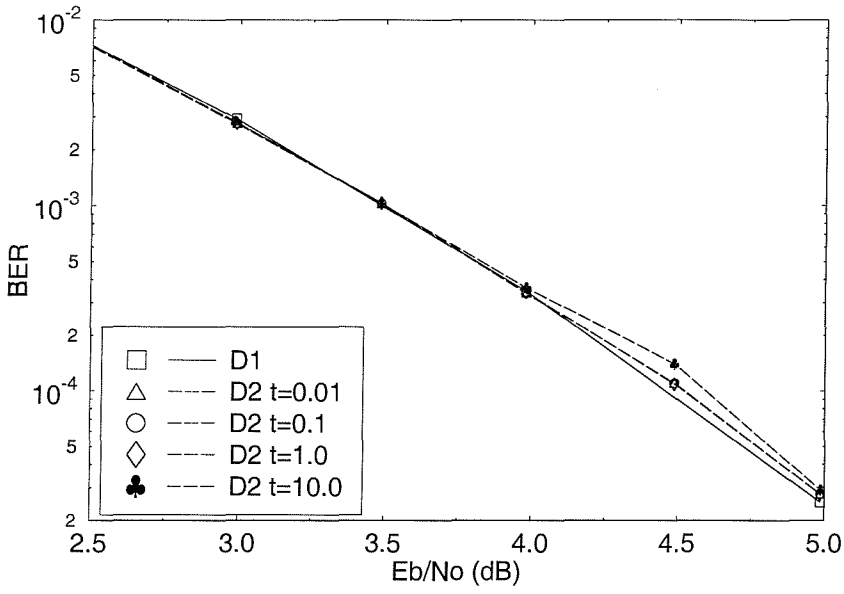


Figure 4.8: Comparison between D1 and D2 in terms of their BER versus Bit SNR performance. The channel is assumed to be static, obeying the impulse response of Figure 4.6(a) and the turbo–PSP equaliser parameters are given in Table 4.1. The variable t denotes the "threshold".

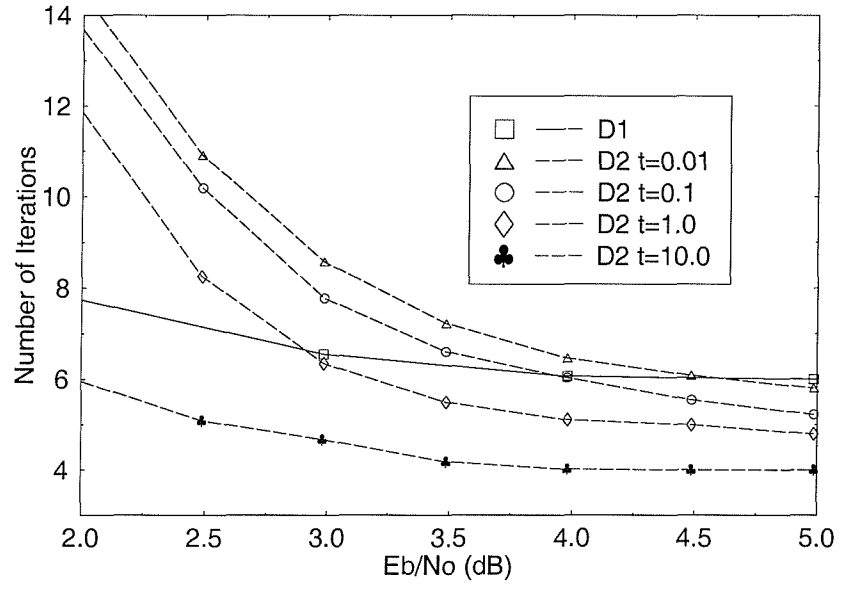


Figure 4.9: Comparison between D1 and D2 in terms of the average number of iterations versus Bit SNR curves. The channel is assumed to be static, having the impulse response shown in Figure 4.6(a) and the turbo–PSP equaliser parameters are given in Table 4.1. The variable t denotes the "threshold".

that in D1 the noise does not affect the slope as much as in D2, since we take into account three previous samples of the LLR variance in addition to the current one, which implies that we are emplying averaging. Moreover, there is a minimum required number of iterations for this detection scheme, since it contains memory and this implies a delay. This minimum required number of iterations does not depend on the SNR. In D1 the convergence detection is dominated by this minimum number of iterations. By contrast, D2 takes into account only the current value of a variance and therefore it detects convergence immediately, when this value becomes less than a threshold, independently of any previous values. At high SNRs the number of iterations required directly depends on the threshold value and convergence is typically achieved within a few iterations. At low SNRs, however, convergence is not readily obtained due to the excessive noise and hence typically a large number of iterations is necessary. In Figure 4.9 we observe that a lower threshold used for D2 requires more iterations, which was expected. We also observe in Figure 4.8 that the BER performance is good, approaching the performance of a turbo equaliser using perfect channel estimation for a sufficiently large number of iterations. This is because the PSP equaliser requires a considerable number of input symbols to converge, especially for higher–order QAM. The feedback loop of the PSP equaliser using the channel decoder's output information expedites convergence. The larger the number of iterations, the better the convergence of the PSP equaliser. Finally, in Figure 4.10 we have plotted the BER versus bit SNR curves for different but fixed numbers of iterations. The modulation schemes used is QPSK, the turbo–PSP equaliser parameters are given in Table 4.1 and the channel is the static channel of Figure 4.6(a). We can observe from Figure 4.10 that no significant performance improvement is achieved after three iterations, implying that employing three iterations would be sufficient in this particular case.

In Figure 4.11 we provide BER performance results for the *2*– and *3*–path fading channels of Figure 4.6 using the parameters given in Table 4.1. The performance is again high compared to the benchmarker using perfect CIR estimation. It is anticipated that the performance will become better when transmitting over the 3-path channel, than that over the 2–path channel, since it is less probable for three paths to fade at the same time than for two paths. A comparison similar to that over the previously studied static channels is given for D1 and D2 in Figure 4.12. Finally, in Figure 4.13 we have portrayed the associated average number of iterations for comparison. We observe again in Figure 4.12 that the associated BER performance is similar for D1 and D2, whilst the number of iterations is now clearly lower for D2, even at low SNRs. This is because in the previous static channel's case the channel's z–domain transfer function contained a zero on the unit circle, rendering convergence slower and less accurate. This resulted in a larger number of iterations required for the variance to be below the threshold value. In the fading channel's case, such channel transfer functions do not occur very frequently, hence the associated convergence is typically faster. Similarly, D1 also requires

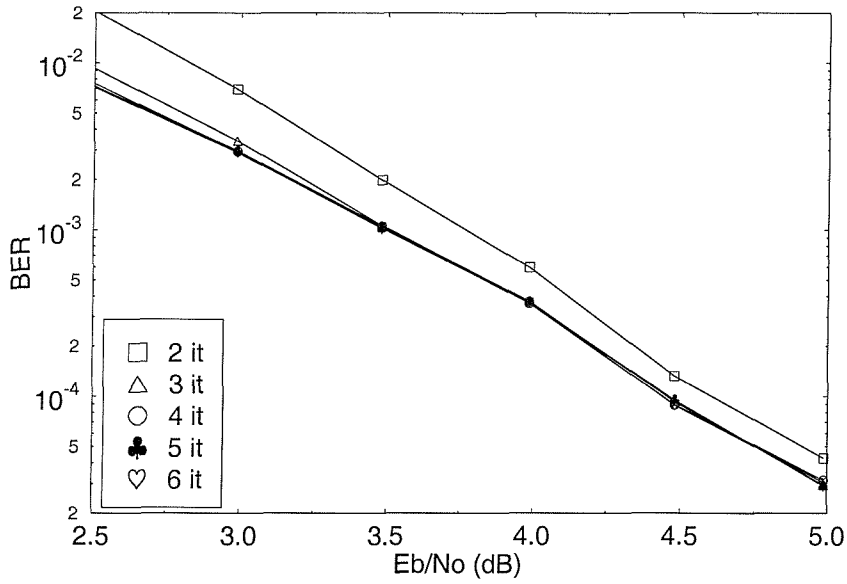


Figure 4.10: BER versus bit SNR curves for different but fixed numbers of iterations. The channel is assumed to be static, having the impulse response shown in Figure 4.6(a) and the turbo–PSP equaliser parameters are given in Table 4.1.

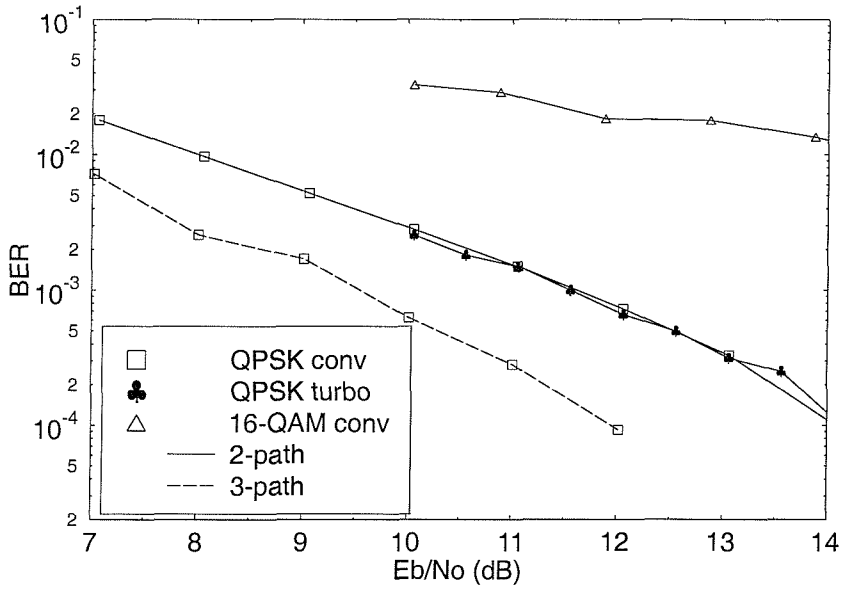


Figure 4.11: BER versus Bit SNR curves over fading channels having the impulse responses shown in Figure 4.6 and using the turbo–PSP equaliser parameters given in Table 4.1.

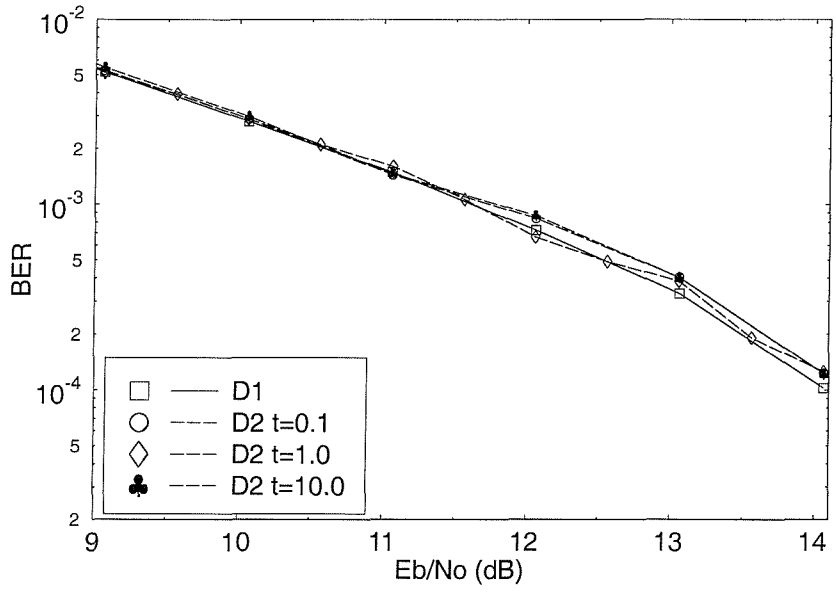


Figure 4.12: Comparison between D1 and D2 in terms of their BER versus Bit SNR curves. The channel is assumed to be Rayleigh fading, obeying the impulse response of Figure 4.6(a) and the turbo–PSP equaliser parameters are given in Table 4.1.

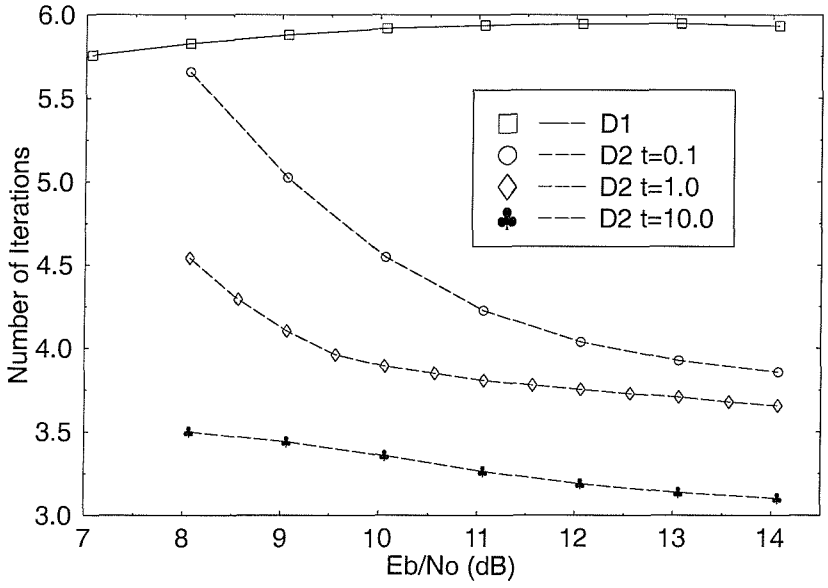


Figure 4.13: Comparison between D1 and D2 in terms of the required average number of iterations versus Bit SNR. The channel is assumed to be Rayleigh fading, using the impulse response of Figure 4.6(a) and the turbo–PSP equaliser parameters are given in Table 4.1. The variable t denotes the "threshold".

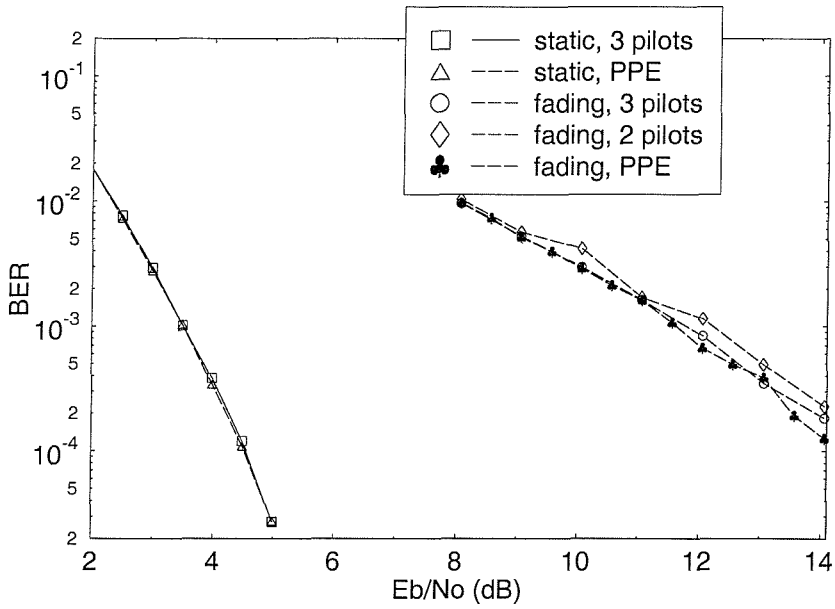


Figure 4.14: BER comparison between perfect phase estimation (PPE) and pilots based phase estimation. The channel is assumed to be static or fading, the impulse response of which is shown in Figure 4.6(a) and the turbo–PSP equaliser parameters are given in Table 4.1.

less iterations in this case for the same reason. However, the number of iterations has to exceed a minimum in order to overcome the memory effect associated with this detector. In the performance results presented so far the phase rotation problem has been neglected and the phase estimation has been assumed to be perfect. In a more realistic scenario, a few pilot symbols can be used to estimate the phase, without creating a significant overhead. For example, if we use three pilots out of 174 symbols of a transmission frame, then the overhead is around 1.7%. This corresponds to a shift of about 0.07dB, if we take it into account in the BER versus Bit SNR curves, which is almost negligible. In Figure 4.14 we have plotted the BER curves for both the static and fading channels of Figure 4.6 using both perfect phase estimation (PPE) with three pilot symbols. The performance is clearly very similar for both the static and fading channels and even for low SNR values and for twin–pilot assisted estimation. This justifies the employment of the phase estimation approach.

In the simulation results presented so far the parameter M of the M–algorithm was equal to the total number of trellis states. However, transmitting over higher–delay channels is a task, which imposes a higher complexity due to the fact that the number of states encountered is exponentially related to the number of channel paths. We will now show the effect of reducing the number of states considered on the performance and complexity of the algorithm. In Figure 4.15 we have plotted the BER against the Bit SNR for the three–path static channel of Figure 4.6 for different number of trellis states considered. As we can see from this figure, the reduction of the number of retained trellis states results in a shift of

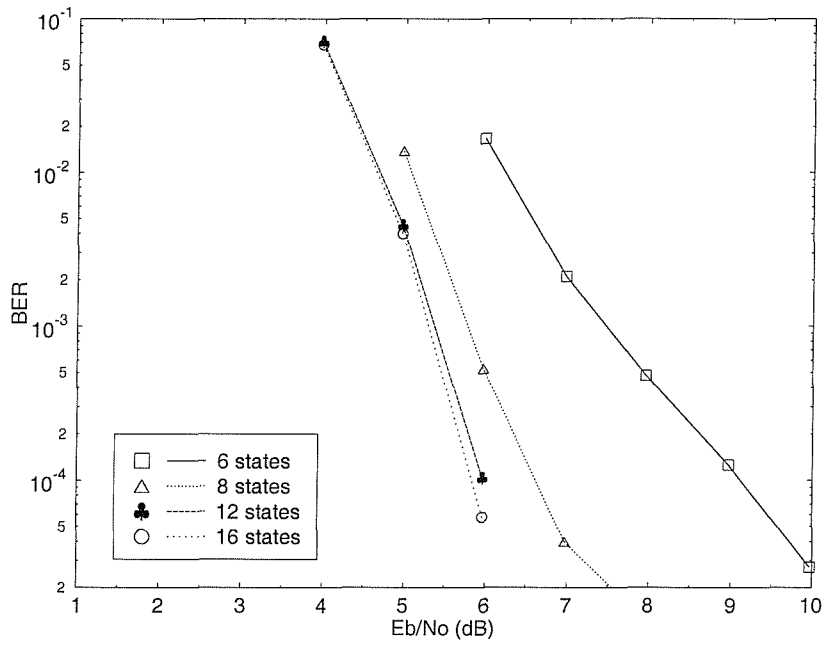


Figure 4.15: BER comparison for different number of states M, employing QPSK. The channel is assumed to be a three–tap equal–weight static channel, the impulse response of which is shown in Figure 4.6(b) and the turbo–PSP equaliser parameters are given in Table 4.1.

the BER curve towards higher SNRs, i.e. in a reduction of the performance of the system. However, this performance reduction is less than $1dB$, when we employ only half of the maximum number of states. In Figure 4.16 the same plot is given for a four–tap equal–weight channel. We can see that, in this case retaining only half of the total number of states gives a similar performance to that, when retaining a higher number of states. On the same note, when retaining 15 states, accounting for about one fourth of the total number of states, imposes only about $1dB$ SNR performance degradation at $BER = 10^{-4}$. Thus, retaining a lower number of states is adequate for this equaliser and the required proportion of states in relation to the total number of states is lower for higher–dispersion CIRs, when aiming for a given target performance.

Finally, having discussed the effect of reducing the number of states in the turbo–PSP algorithm, in Figure 4.17 we have plotted the achievable coding gain at a BER of 10^{-4} as a function of the decoding complexity for both the convolutional and turbo–coded schemes, in the latter case in conjunction with four turbo iteration loops. The coding gain is defined as the SNR gain of using channel coding as opposed to not using channel coding at a certain BER and it is measured in dB. In our estimate the complexity was deemed to depend on that of the channel decoder. Under this assumption, the turbo–coded system has a complexity which is $O(2^K \cdot N \cdot L)$ with N being the number of iterations of the turbo channel decoder and L being the number of turbo equaliser iterations, assuming that the figure

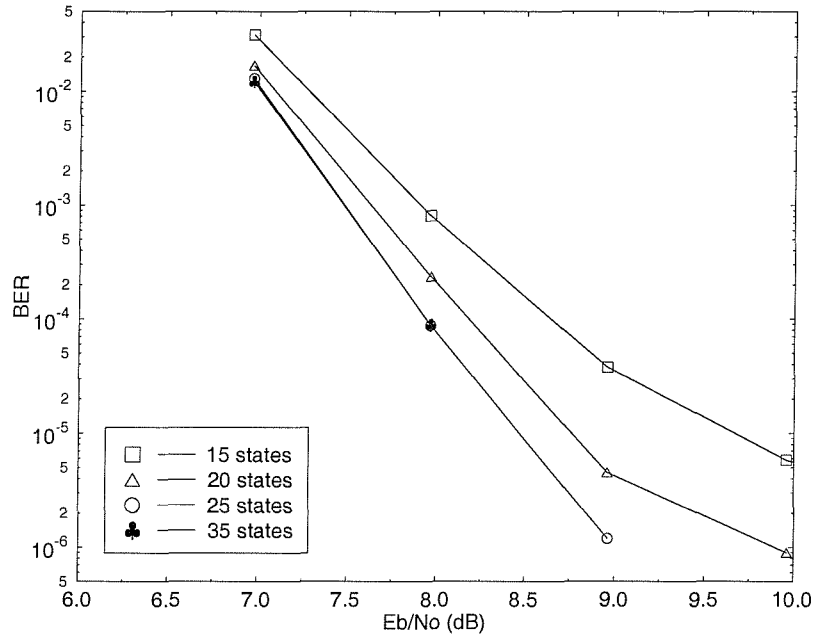


Figure 4.16: BER comparison for different number of states M, employing QPSK. The channel is assumed to be a four–tap equal–weight static channel and the turbo–PSP equaliser parameters are given in Table 4.1.

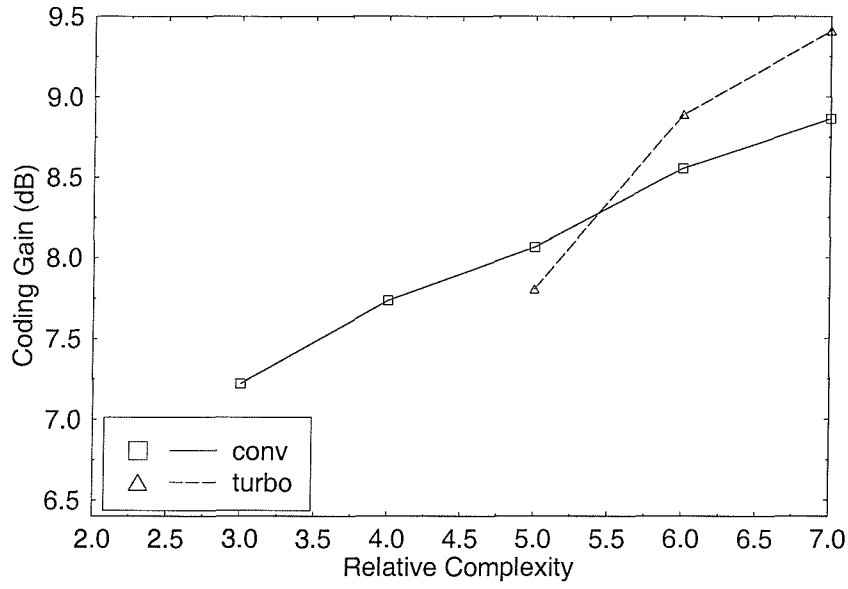


Figure 4.17: The coding gain as a function of complexity for convolutional and turbo-coded turbo-PSP schemes. The channel is assumed to be static, the impulse response of which is shown in Figure 4.6(a) and the turbo-PSP equaliser parameters are given in Table 4.1.

2^{K-1}, which is the complexity of each of the two MAP channel decoders, is multiplied by two, since there are two such constituent decoders, and also by the number of iterations, N. The specific codes used optimum generator polynomials both for the convolutional and for the turbo codes. As seen in Figure 4.17, it was found that the coding gain was higher for the convolutional coded turbo-PSP scheme at a low complexity, while this performance trend was reversed for a higher complexity.

4.6 Summary

A novel blind equaliser was proposed, jointly performing channel equalisation and channel decoding by combining turbo equalisation with PSP. We achieved this by feeding channel decoding information back to the PSP equaliser as a-priori information and repeating this process in an iterative fashion. This enhanced-reliability feedback information output by the channel decoder enhances the equaliser's performance in terms of its robustness against channel noise and also in terms of its convergence capability, provided that the soft feedback information does not contain an excessive number of errors. Clearly, this technique enhances the performance of the PSP based channel equaliser iteration by iteration. However, when the soft LLR feedback information applied by the channel decoder to the equaliser is unreliable, then the PSP equaliser may be driven to instability. As in the case of a conventional PSP based equaliser, the associated exponentially increasing complexity can be reduced

by employing the M–algorithm, that is by retaining a limited number of trellis states, when the highly dispersive CIR requires a long channel equaliser trellis, which would result in an excessive complexity. In this case, the PSP equaliser will perform worse than in the conventional open loop scenario, since its a–priori feedback will generally be more error–prone. Nevertheless, we found that this performance degradation is not significant for CIR durations of up to four symbol intervals. We also found that this performance degradation decreases when the CIR duration increases and the proportion of the total number of states retained is kept the same. Thus, taking into account that the simulated CIRs represent worst case scenario, we anticipate that this equaliser can cope with higher channel order and limited complexity.

The number of iterations used was controlled by using a convergence detection algorithm. This algorithm is not optimum and if improved, it can reduce the number of iterations and the associated complexity required.

The sign ambiguity issue, inherent in blind equalisation, could not be solved by simply employing differential coding. This is because the turbo equaliser would require a differential encoder processing soft inputs, which would require an excessive complexity. Nonetheless, this issue has been resolved by using a low number of pilot symbols. In fact, using pilot symbols for estimating the phase has proven to be a very good trade–off between the performance degradation imposed by using differential encoding and the bandwidth reduction incured by sending pilot symbols, rather than transmitting information.

Having introduced and discussed the convolutionally–coded turbo–PSP algorithm, in the next chapter we will introduce a modification of this algorithm employing coded modulation schemes.

Chapter 5

Combined Trellis–Coded Modulation and Turbo Equalisation[1]

5.1 Introduction

After having introduced the turbo–PSP concept in the previous chapter, we will now introduce a modification of this algorithm, which employs combined modulation and channel coding techniques and benefits from invoking symbol– instead of bit– based decoding. This method exploits the enhanced data protection offered by Trellis–Coded Modulation (TCM) [185] or Bit–Interleaved Coded Modulation (BICM) [186] and exhibits good performance in terms of its output Bit Error Rate (BER). Explicitly, a BER comparable to that of a trained turbo equaliser is attained at the cost of a modest complexity increase.

The chapter is organised as follows. The communications system is described in Section 5.2, while in Section 5.3 the proposed coded modulation turbo–PSP equaliser is described. Specifically, in Section 5.4 the encoder and decoder used in the coded modulation schemes are detailed. Finally, in Section 5.5 performance results are provided for fading dispersive channels.

5.2 System Description

The communications system under consideration is shown in Figure 5.1. The information bits are mapped to QAM symbols by a channel encoder, which can be a trellis coded modulation encoder [185], a turbo trellis coded modulation encoder [187] or a convolutional encoder in the case of BICM [186].

[1] This chapter is based on joint work with my colleague Michael Soon Ng, whose contribution is gratefully acknowledged [13]

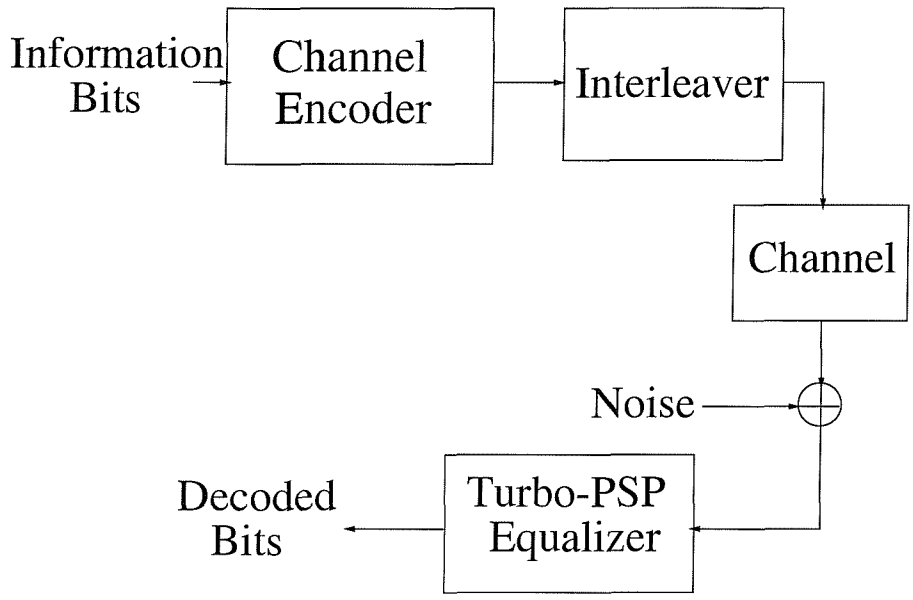


Figure 5.1: Equalised communications system

The QAM symbols are then interleaved, in order to disperse the channel's bursty errors as well as to enhance the turbo-equaliser's performance, generating the transmitted QAM symbols $a(n)$. These symbols are convolved with the Channel's Impulse Response (CIR) h_i and then the channel noise $e(n)$ is added, yielding the received symbols $y(n)$ as:

$$y(n) = \sum_{i=-L_1}^{L_2} h_i \cdot a(n-i) + e(n). \tag{5.1}$$

The restoration of the original information bits is performed by the turbo-PSP equaliser. In the next section we provide further details concerning the operation of the turbo-PSP equaliser using TCM, TTCM and BICM.

5.3 Turbo-PSP Equaliser Description

The turbo-PSP equaliser performs joint channel equalisation and coded modulation decoding using the schematic shown in Figure 5.2. The operation of this system is the same as that of the turbo-PSP equaliser using convolutional coding, as highlighted in Section 4.2. The sole difference is the replacement of the convolutional decoder by a TCM decoder. Hence, a bit-to-symbol converter is placed before the TCM decoder on order to convert the LLR values to symbol probabilities, which are necessary for facilitating TCM decoding. Similarly, a symbol-to-bit converter is employed at the output of the decoder. The calculation of the symbol probabilities from the probabilities of the input

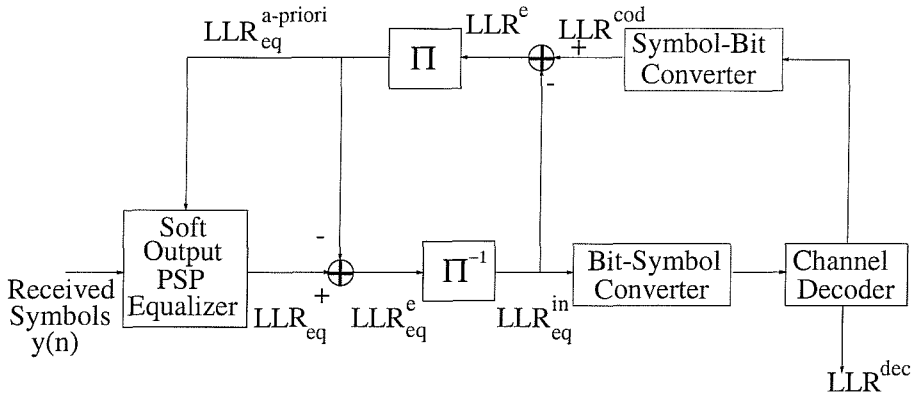


Figure 5.2: The turbo–PSP equaliser

bits is based on the following formula:

$$Prob(Symbol = A_i) = \Pi_j Prob(Bit_j(Symbol) = Bit_j(A_i)) \qquad i = 0, \cdots, Q, \quad j = 0, \cdots, K, \tag{5.2}$$

where Q is the number of symbols in the modulation constellation used, K is the number of bits per symbol, A_i represent the constellation symbols and the function $Bit_j(A_i)$ returns the value of the j–th bit of symbol A_i. The assumption implicitly stipulated here is that the bits are independent of each other. This, however, is not a valid assumption, since channel coding has deliberately imposed a certain amount of correlation or interdependence on the bits, in order to exploit this redunduncy for correcting transmission errors. Similarly to the bit–to–symbol conversion of Equation (5.2), the symbol–to–bit conversion is based on the following formula:

$$Prob(Bit_j = Bit) = \sum_i Prob(Symbol_i; Bit_j(Symbol_i) = Bit), \tag{5.3}$$

This formula is accurate, in the sense that it does not require any assumptions concerning the correlation of the input symbols. However, any correlation amongst the bits of a block would have already been lost at the bit–to–symbol converter of Figure 5.2 and hence the input of the bit–to–symbol converter is uncorrelated.

Having described the basic functions of the component modules of the turbo–PSP equaliser in Figure 5.2, we will now proceed to highlighting the operation of these components. The PSP equaliser is the same module as the one used in Section 4.3.1 of the previous chapter. Therefore, we will not elaborate on it further in this chapter. In the next section we describe the operation of the TCM encoder and decoder.

5.4 Coded Modulation Schemes

Trellis Coded Modulation (TCM) [185] was proposed originally for Gaussian channels and it was later further developed for applications in mobile communications [188, 189]. Turbo Trellis Coded Modulation (TTCM) [187] is a more recent joint coding and modulation scheme that has a structure similar to that of the family of power efficient binary turbo codes [135], but employs TCM schemes as component codes. Both TCM and TTCM use symbol based interleavers and Set–Partitioning (SP) based signal labeling. Another coded modulation scheme distinguishing itself by utilising bit–based interleaving in conjunction with Gray signal constellation labeling is referred to as Bit–Interleaved Coded Modulation (BICM) [186]. The number of parallel bit–interleavers equals the number of coded bits in a symbol for the BICM scheme proposed in [186]. Recently iterative joint decoding and demodulation assisted BICM (BICM–ID) was also proposed in [190,191], in an effort to further improve the system's performance, which uses SP based signal labeling.

5.4.1 TCM Turbo–PSP

The TCM decoder operates similarly to the corresponding module of a trained turbo equaliser [135, 182, 183]. A non–binary MAP decoder is invoked for the channel decoder module. Given the LLR value $LLR(Bit)$ of a binary bit, we can calculate the probability of $Bit = +1$ or $Bit = -1$ as follows. Remembering that $Prob(Bit = -1) = 1 - Prob(Bit = +1)$, and taking the exponent of both sides in Equation 4.2 we can write:

$$e^{LLR(Bit)} = \frac{Prob(Bit = +1)}{1 - Prob(Bit = +1)}. \tag{5.4}$$

Hence we have:

$$Prob(Bit = 1) = \frac{e^{LLR(Bit)}}{1 + e^{LLR(Bit)}}, \tag{5.5}$$

and,

$$Prob(Bit = -1) = \frac{1}{1 + e^{LLR(Bit)}}. \tag{5.6}$$

Then, assuming that the bits of a symbol are independent of each other, the probability of a *Symbol* which is represented by the bits $Bit^1, \ldots, Bit^n$, can be calculated as:

$$Prob(Symbol) = \prod_{i=1}^{i=n} Prob(Bit^i), \tag{5.7}$$

where we have $Symbol \in (0, \ldots, 2^n - 1)$ for the 2^n–array modulation scheme used. The probability of the transition from states s' to state s, commonly defined as $\gamma(s', s)$, used in the non–binary MAP

decoder [135], can be calculated as:

$$\gamma(s^{'}, s) = \eta_{Symbol} \cdot \prod_{i=1}^{i=n} Prob(Bit^i), \tag{5.8}$$

where *Symbol* is the trellis transition branch label from state $s^{'}$ to state s and η_{Symbol} is the "a priori" information of the *Symbol*. Then the α and β values can be obtained using Equations 4.5 and 4.6, as for the PSP equaliser.

The number of transitions emerging from a specific trellis state is equal to 2^k, where k is the number of information bits per n–bit modulation symbol. The coding rate used is $R = \frac{k}{n}$, where $k = n - 1$. Therefore the log–MAP decoder used is non–binary when $k > 1$. By contrast, if $k = 1$, then the number of transitions emerging from a trellis state is equal to $2^1 = 2$, i.e. a binary MAP decoder is used. The a–posteriori probability (APP) of the information symbol u_t, $u_t \in (0, \ldots, 2^k - 1)$ at time instant t can be computed from [135]:

$$APP(u_t) = \sum_{s^{'} \to s, u_t} \alpha_{k-1}(s^{'}) \cdot \gamma_k(s^{'}, s) \cdot \beta_k(s). \tag{5.9}$$

The final decoded information symbol at instant t is the hard decision based symbol generated from these APP values. However, we have to feed back the LLR values of all the n coded bits, rather than just those of the k information bits, to the PSP equaliser, after improving their reliability by the channel decoder. The APP of the coded symbol x_t, $x_t \in (0, \ldots, 2^n - 1)$ at time instant t can be computed from:

$$APP(x_t) = \sum_{s^{'} \to s, x_t} \alpha_{k-1}(s^{'}) \cdot \gamma_k(s^{'}, s) \cdot \beta_k(s), \tag{5.10}$$

while Equation (5.9) formulated the APP of the original encoded information symbol. The probability of bit i to be 1 in a coded symbol x is calculated from:

$$Prob(Bit^i = 1) = \sum_{x=0}^{x=2^n-1} APP(x^i = 1), \tag{5.11}$$

where x^i denotes the binary value at bit position i of the symbol x, $x^i \in (0, 1)$ and in verbal terms the probability of $Bit^i = 1$ is given by the sum of the probabilities of all symbols from the set of $2^n - 1$ number of phasors, which have a binary 1 at bit position i. Similar procedure is carried out for determining $Prob(Bit^i = 0)$ and finally the LLR of the bits can be computed from Equation 4.2.

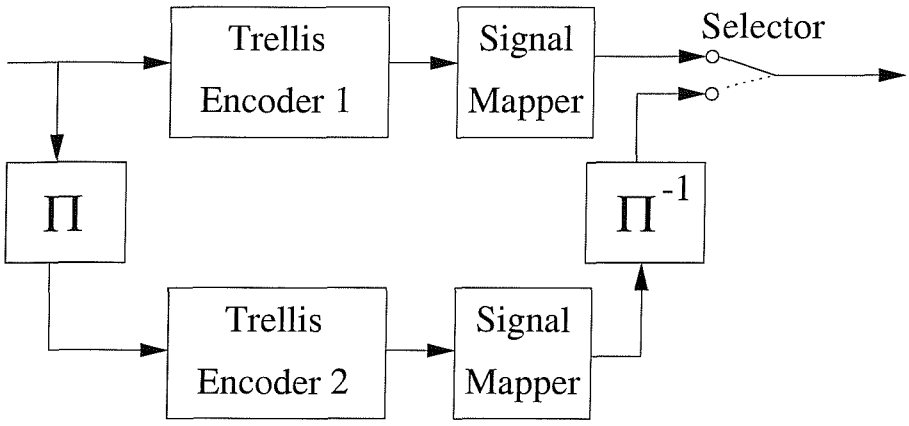


Figure 5.3: Turbo trellis–coded modulation encoder.

5.4.2 TTCM Turbo–PSP

An extension of the turbo–PSP equaliser employing Turbo Trellis–Coded Modulation (TTCM) has also been considered. TTCM was proposed in [187], which avoids the disadvantage of an effective throughput loss, which would be incurred when applying a direct parallel concatenation of two TCM schemes. Specifically, the effective throughput loss is avoided by puncturing the parity information in a particular manner, so that all infomation bits are sent only once, while the parity bits are provided alternatively by the two constituent TCM encoders. The TTCM encoder is shown in Figure 5.3, which comprises two identical TCM encoders linked by the symbol interleaver Π. The TTCM decoder structure shown in Figure 5.4 is similar to that of binary turbo codes, except for the difference in the nature of the information passed from one decoder to the other. Each decoder alternately processesq its corresponding encoder's channel impaired output symbol, and then the other encoder's channel impaired output symbol. The information bits, i.e. the systematic bits, are constituted by the corresponding systematic TCM encoder output bits received over the channel in both cases. The systematic information and the parity information are transmitted together in the same symbol. Hence, the systematic information component cannot be separated from the extrinsic information, since the noise that affects the parity component of a TTCM symbol also affects the systematic information component. The output of each symbol–based MAP decoder of Figure 5.4 can be split into two components [135]:

1. the *a priori* component, and

2. the amalgamated (extrinsic systematic) [*e*&*s*] component.

Each decoder of Figure 5.4 has to pass only the [*e*&*s*] component to the other decoder, which is written in parentheses in order to emphasize the inseparability of the extrinsic and systematic components.

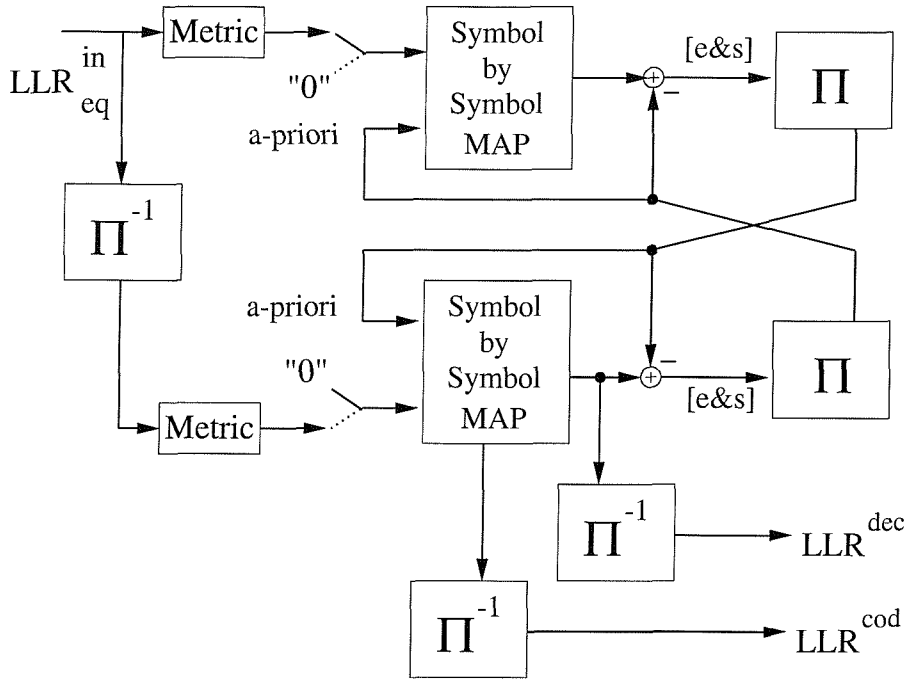


Figure 5.4: TTCM decoder.

The reason the a–priori component is subtracted from the output of the symbol–based MAP decoder of Figure 5.4 is that this information component was generated by the other decoder and hence it must not be fed back to it. Otherwise the probability estimates of the two decoders become dependent on each other in enhancing the decoder's decision reliability. The LLR_{eq}^{in} output of the equaliser is forwarded to the "metric" calculation block of Figure 5.4, in order to generate a set of 2^n symbol reliabilities. The selectors, in front of the Symbol by Symbol MAP decoder of Figure 5.4, select the current symbol's reliabilities from the "metric" calculation block, if the current received symbol corresponds to that component decoder, otherwise depuncturing will be applied, where the reliabilities of the symbols are set to a fixed value corresponding to no a–priori information, which is represented by "0" in the logarithmic domain). The "metric" calculation block provides the decoder with the parity and systematic $[p\&s]$ information, and the second input to the symbol by symbol MAP decoder of Figure 5.4 is the *a priori* information acquired from the other decoder. The MAP decoder then provides the *aposteriori* information constituted by the (*a priori* + $[e\&s]$) components as the output. Then the *a priori* information is subtracted from the *a posteriori* information, again so that information is not used more than once in the other decoder. The resulting $[e\&s]$ information is interleaved (or de–interleaved) in order to create the *a priori* input of the other decoder. This decoding process will continue iteratively, in order to generate an improved version of the set of symbol reliabilities for the other decoder. One iteration comprises the decoding of the received symbols by both of the component decoders. Finally, the *aposteriori* information of the lower component decoder of Figure 5.4 will be de–interleaved, in order to extract $(n-1)$ decoded information bits per symbol. On the

other hand, the *aposteriori* information of the n coded bits is de-interleaved, in order to provide the LLR^{cod} signal of Figure 5.2 to the equaliser input.

5.4.3 BICM Turbo–PSP

The major difference between BICM [186] in comparison to TCM [185] is that bit based interleaving is used and non–systematic convolutional codes are utilised for protecting the information. The decoding of BICM is similar to that of TCM, as ot was highlighted in Section 5.4.1.

These two coded modulation schemes exhibit a similar coding rate, depending on the modulation mode used. Specifically, i.e rate–1/2 code is used for QPSK and a rate–2/3 code is employed for 8–PSK. In our investigations, Gray signal constellation labelling is invoked, since we found that the performance of the combined system was optimum for Gray signal labelling. This is because the equaliser works better in Gray labelling and has a greater influence in performance term compared to the coded modulations.

Having described the most important components of the proposed coded modulation based turbo–PSP equaliser, we will now provide performance results characterising the various schemes, which were obtained by means of computer simulations.

5.5 Performance Results

In this section we present performance results for the turbo PSP–equaliser described in Section 5.3. The results are based on computer simulations. The channels assumed are fading, having the CIRs given in Figure 4.6. The convergence control is assumed to be D2, as it was defined in Section 4.5 of the previous chapter. The general turbo–PSP equaliser parameters are given in Table 5.1. In Figures 5.5–5.6 the BER versus Bit SNR curves are given for QPSK and 8–PSK, over the fading channels of Figure 4.6. The Bit SNR was defined in Equation (4.19) of Chapter 4. As it is expected, the performance recorded for transmission over the fading 3–tap channel is better than that over the fading 2–tap channel. This is because in this case the intersymbol interference is reduced, since the probability of two paths fading simultaneously is higher than that of three paths fading simultaneously. We also observe that TCM is the best performer when QPSK is employed, while it performs slightly worse than TTCM in conjunction with 8-PSK. BICM exhibits by far the worst performance, especially when 8-PSK modulation is employed. The important trend we inferred from our simulations is the inferiority of BICM for transmissions over fading channels. Essentially BICM is a binary version of TCM, which is related to the turbo–PSP algorithm of Chapter 4. While TCM uses set–partitioning for maximising

Frame length (bits)	174
Interleaver block length	5x174
Carrier frequency	1900MHz
Symbol rate	2.6MBaud
Doppler speed	48Km/h
Equaliser step–size	$5x10^{-3}$
TCM memory	6
TTCM memory	3
BICM memory	6
TTCM number of iterations	4

Table 5.1: The turbo–PSP equaliser parameters used in the simulations. Observe in the table that for the shake of a fair comparison between TCM, TTCM and BICM, the code memory was 6, 3 and 6 respectively, since TTCM used four iterations, which resulted in a similar complexity for all three schemes.

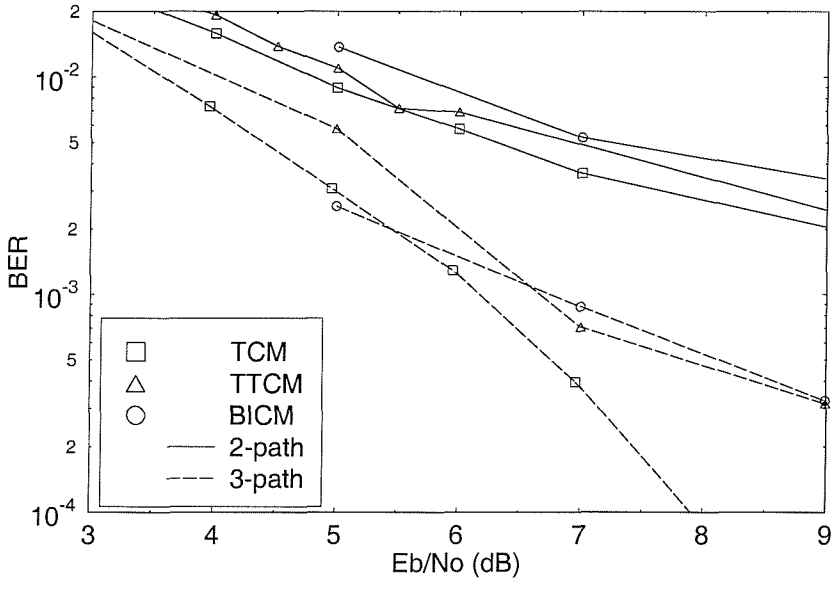


Figure 5.5: BER versus Bit SNR curves over **fading channels**, exhibiting the impulse response shown in Figure 4.6 and using the turbo–PSP equaliser parameters given in Table 5.1. The modulation was **QPSK**, the coding rate was R=1/2 and the iteration control obeyed D2 of Section 4.5.

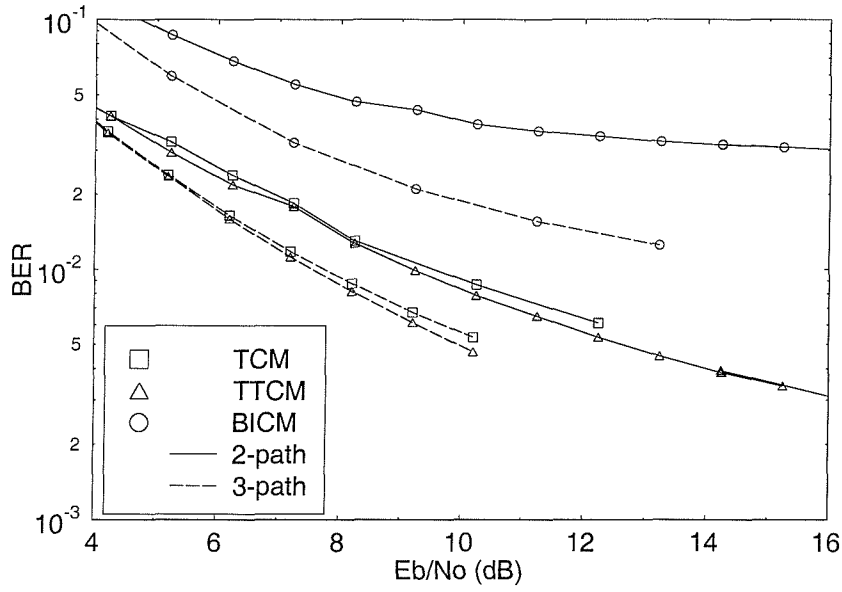


Figure 5.6: BER versus Bit SNR curves over **fading channels**, exhibiting the impulse response shown in Figure 4.6 and using the turbo–PSP equaliser parameters given in Table 5.1. The modulation was **8–PSK**, the coding rate was R=2/3 and the iteration control obeyed D2 of Section 4.5.

the Euclidean distance of the constellation points represented by the encoded bits, BICM is concerned only with bits, not with symbols. However, the PSP equaliser exhibits a degraded performance, when using no Gray coding, since then the neighbouring symbols can have more than one different bits. With the aid of simulations we found that the PSP equaliser's performance degradation encountered due to using no Gray coding was higher, than the performance degradation incurred when using Gray–coded TCM, rather than classic set-partitioning based TCM. Thus in our simulations Gray-coded modulation was used in conjunction with TCM, TTCM and BICM. In fact a TCM turbo–PSP equaliser would perform better, if the bit probabilities represented by the LLRs were replaced by symbol probabilities. However, this would dramatically increase the associated complexity, since it would imply that at each symbol instant the turbo–PSP scheme would have to process an increased number of values, which is equal to the number of symbols in the QAM constellation, rather than the number of bits per symbol. Therefore, if K is the number of bits per symbol, we would have a factor of $2^K/K$ more LLR values to process. In order to avoid this complexity problem, we have used bit LLR values, just as in the case of the turbo–PSP scheme of Chapter 4. As we have mentioned before, this is not optimum in conjunction with symbol–based coding techniques, such as TCM and TTCM, because we assumed that the probability of a symbol is given by the product of the probabibilities of its constituent bits. This assumption ignores the correlation amongst these bits, as it was highlighted in Section 5.2.

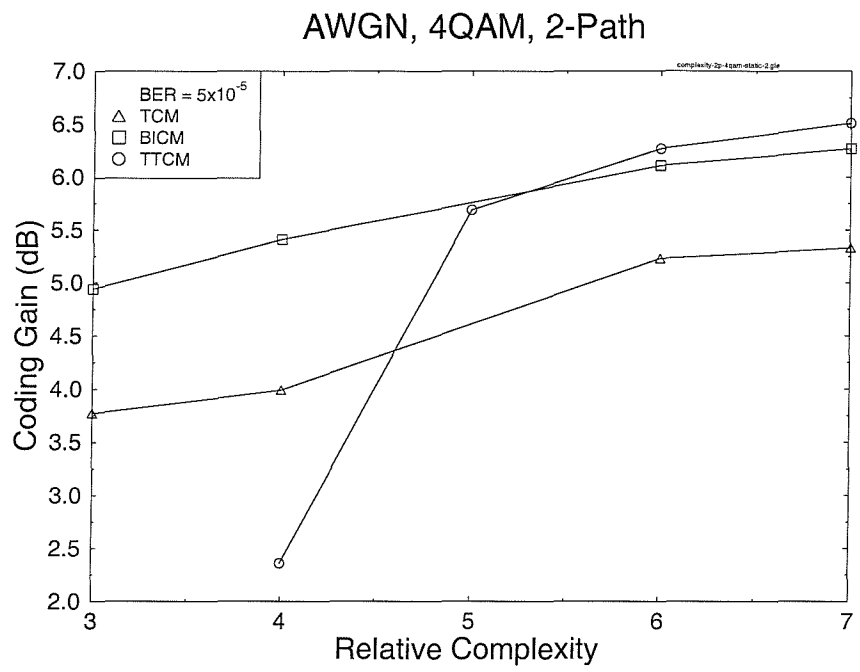


Figure 5.7: The coding gain of TCM, TTCM and BICM at a BER of $5 \cdot 10^{-5}$ versus complexity for transmissions over the 2–path channel of Figure 4.6, employing QPSK.

In Figure 5.7 we have plotted the coding gain of the various techniques at a BER of $5 \cdot 10^{-5}$ for QPSK for transmissions over the static 2–path channel of Figure 4.6 versus their relative complexity. It has been assumed that the complexity of the PSP equaliser is approximately the same for all the algorithms and therefore the only factor that affects the total complexity is the complexity of the channel decoder. We observe from Figure 5.7 that the coding gain is the highest for BICM at a lower decoding complexity. When the complexity is increased, then the coding gain of the TTCM becomes marginally better. This happens at a coding gain of approximetaly 5.7dB. The above observation is in accordance with our previous remarks. TTCM performs better for the low–ISI scenario, which is achieved when higher complexity is affordable and the difference in performance is higher, when the affordable complexity increases, since as the complexity increases, the advantageous characteristics of turbo decoding become more dominant.

5.6 Summary

In this chapter a novel blind equaliser was proposed, extending the turbo–PSP algorithm of the previous chapter by replacing the separate convolutional channel coding scheme with coded modulation techniques. The idea originated from the fact that by combining modulation and channel coding the turbo–PSP equaliser would become more efficient, since it is a symbol–based detection technique. However, three main calamities exist in the context of this technique. On the one hand, when the

symbol probabilities replace the bit LLR values, then the complexity and storage requirements increase dramatically, especially in the context of higher–order QAM schemes. Thus the turbo equaliser was modified for using bit–based LLR values for communicatiing amongst the detection blocks of Figure 5.2. This gave rise to the second main problem. Specifically, the conversion of the bit LLR values to symbol probablities was based on the assumption that the bits of a symbol are independent even after channel encoding. However, this assumption inposes a performance degradation on the TCM and TTCM decoders, since the symbol probabilities became somewhat inaccurate, although this was partially compensated by the iterative TTCM scheme. Finally, the employment of Gray–coded modulation, which was preferable in terms of the overall system's performance, gave rise to a performance degradation of the TCM and TTCM decoders, since these schemes were optimised for set–partitioning based mapping of the bits to the symbols.

Nevertheless, in Figures 5.5 and 5.6 we found that the TCM and TTCM schemes outperformed the BICM arrangement, which was similar to the convolutional coding, used in Chapter 4 in the context of QPSK and 8-PSK and fading channels. TCM performs better in conjunction with QPSK, while TTCM performrs better when the modulation employed is 8-PSK.

Having presented a range of coded modulation based turbo–PSP schemes, a summary of our findings will conclude this treatise in the next chapter.

Chapter 6

Conclusions and Further Work

In this treatise the potential of reducing or eliminating the communications overhead required for transmitting the equaliser training information was studied using blind equaliser. These equalisers exploit the various statistical characteristics of the received signal and the a–priori known statistical characteristics of the transmitted signal for removing the channel–induced dispersion from the received signal without employing explicit training. Over the past decades significant research efforts have been focused on this issue and different techniques have been proposed. Some of these techniques have been investigated in this treatise and their performance has been quantified. Specifically, it was confirmed that the PSP technique of Section 1.3.9 offers a substantial performance advantage compared to the family of Bussgang techniques, highlighted in Section 1.3.5, in terms of convergence speed, convergence precision and robustness to noise. On the other hand, PSP technique is significantly more complex, than the Bussgang techniques. The M–algorithm based approximation of Section 1.3.9 can be used for reducing the complexity imposed by the PSP, by reducing the number of trellis states retained, while sustaining an adequate level of performance. Additionally, we found that in scenarios, where the CIR contains paths having negligible tap value, we can decrease the complexity by assuming that these paths are zero and therefore do not contribute to the ISI. This is equivalent to invoking a reduced–state PSP, because some of the states of the full–state PSP are effectively obliterated. As argued in Section 1.3.9.1, the task in this case is to decide, which of the CIR taps, if any, can be eliminated. The solution proposed is to inspect the full–state PSP and reduce unnecessary taps when their value is below a threshold value and reinstate them one by one again once the error detection codecs of the system decided that the ISI was not sufficiently reduced. However, an algorithm which would decide on the locations of the channel taps to be removed or added is an open research isue.

The application of range of blind equalisers was studied in terms of a satellite based digital video broadcast (DVB) system. It was confirmed in Section 2.6 that for this particular application the

performance of PSP was vastly superior compared to the set of Bussgang equalisers studied and also that by adding turbo–coding, we can significantly improve the achievable performance. More specifically, since the employment of turbo–coding resulted in a 5dB power budget gain, it was possible to invoke a higher throughput, but less error–resilient modulation.

Following these investigations we focussed our attention on designing equalisers, which are suitable for transmissions over channels having equal–weight CIR taps. The equalisers designed for such channels have a large number of non–negligible taps. This constraint gave rise to the idea that an equaliser having a feedback section could equalise both the pre–cursor as well as the post–cursor parts of the ISI. More importantly, the feedback section of this equaliser would be capable of mimicking the CIR, so that when the equaliser has a high number of non–negligible magnitude taps, then this feedback section can invert the channel effects without the aid of the feedforward section. Thus, the number of taps required would be reduced. This idea led us to the design of the DFE–CMA equaliser, which extends the conventional CMA to a soft–decision feedback assisted equaliser. This scheme was capable of creating a link between an equaliser designed for inverting the CIR and an equaliser designed for mimicking it. Our studies showed that the undesirable local minima associated with the conventional CMA do not exist in the context of the DFE–CMA, since the feedback section does not allow such points to form stationary points. Our simulation results confirmed the superiority of this algorithm over the conventional CMA when encountering equal–weight CIRs. An optimisation technique, which would identify whether a longer feedforward or feedback section is more suitable for a specific propagation environment in the context of the DFE–CMA, constitutes an interesting further research issue, which would render this system directly applicable to fading channel scenarios.

Following our quest for a rapidly converging blind equaliser having a superior performance, such as that exhibited by PSP, the idea of employing feedback, used in the DFE–CMA in the form of feeding equalised channel symbols back to the equaliser, was extended to replacing this feedback information by channel decoding assisted information invoking the concept of iterative turbo equalisation. In this respect, blocks of soft bit probability values are passed back and forth from the PSP equaliser to the channel decoder and vice versa. Accordingly, a novel algorithm combining channel equalisation with channel decoding was proposed, namely the concept of the turbo–PSP scheme, exploiting the channel decoding information for providing increased–confidence soft valued feedback to a PSP equaliser in an iterative fashion. The idea goes back to the originators of PSP [23], who proposed its employment in conjunction with channel coding, such that channel decoding and equalisation are performed simultaneously. In this algorithm, the decision concerning the transition from one trellis state to another depends on a metric produced by channel decoding, rather than by channel estimation. The proposed algorithm has the advantage that channel decoding and channel equalisation are separated, hence they

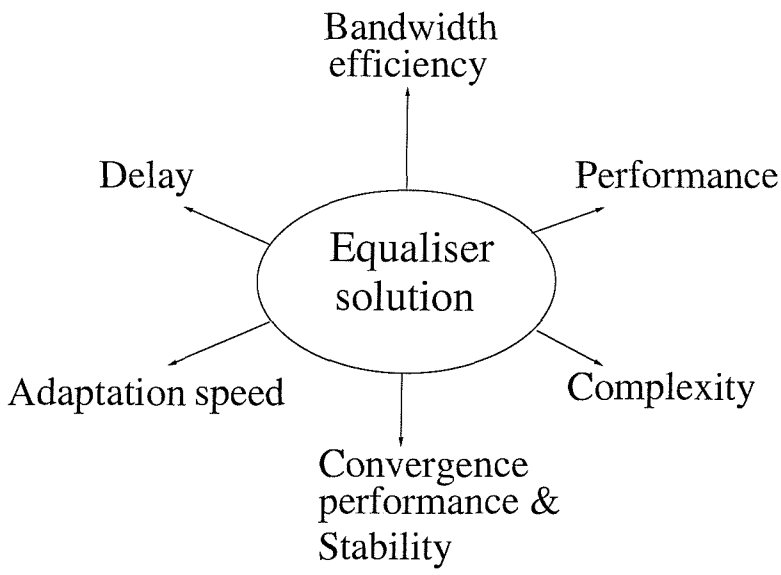


Figure 6.1: Contradictory design factors in the design of blind equalisers

can operate iteratively. Moreover, turbo channel coding can also be used, enhancing the performance in mildly dispersive scenarios. The issues discussed previously in the context of PSP are also valid in this case. However, because of the a–priori feedback information provided for the equaliser, the performance is more sensitive to decision errors fed back by the equaliser to the channel decoder. Thus in the context of this algorithm good initialisation is essential, in order to avoid the equaliser's instability. In the context of the turbo–PSP equaliser, the sign ambiguity problem, which is inherent in blind equalisation, has been solved by transmitting a low number of known pilot symbols carrying the phase reference information. It was found that using only a low number of pilot symbols was sufficient for maintaining a performance similar to that of the scheme using perfect phase estimation. In general, when employing this technique is a viable option, it is preferable to using differential encoding, since the amount of extra overhead it requires is minimal and can be sonsidered as a modest investment, compared to the 3dB SNR loss, which is the approximate performance degradation, when differential encoding is used. In the case of employing turbo–PSP, this technique was the only available option, since differential encoding could not be employed in the context of the soft LLR values used.

The blind equaliser design considerations, which were taken into account in the course of this work were sumarised in Figure 1.7 of Chapter 1, which is repeated here as Figure 6.1 for convenience. Considering the equaliser's convergence speed, convergence accuracy and robustness to noise, the PSP equaliser is the best performer from the set of equalisers evaluated in this treatise, as we observed from the BER versus SNR plots of Chapter 2. Additionally, in the context of the performance results plotted in Chapters 4 and 5, as we observed from Figures 4.7–4.17 and 5.5–5.7, we showed that the turbo–PSP scheme provides an enhanced performance in terms of convergence accuracy and robustness to noise,

owing to the data protection offered by channel coding. This improvement is more pronounced, when coded modulation schemes are employed in the context of transmissions over fading channels, while for transmissions over static channels convolutional coding appears to have the edge. By employing iterative decoding such as in turbo convolutional decoding or in iterative turbo TCM detection, the equaliser exhibits a better performance, than the non–iterative channel decoding turbo–PSP schemes, such as convolutional or TCM, for transmissions over static channels. However, for transmissions over fading channels iterative decoding performs worse, than its non–iterative counterpart, when the affordable complexity limits the number of turbo iterations and hence the turbo decoder's performance.

The Bussgang techniques evaluated in Chapter 2 of this treatise did not exhibit the same attractive performance characteristics, as the PSP based techniques. However, their complexity was significantly lower. These characteristics render them attractive for transmissions over channels exhibiting mild and slowly variant distortion, as well as for transmissions using higher–order modulation schemes, since unlike in the case of PSP techniques, their complexity does not depend on the modulation constellation.

Finally, all the blind algorithms studied in this treatise are bandwidth efficient in comparison to their trained counterparts in the sense of utilising all the available bandwidth for transmitting information, since they refrain from using an explicit training sequence. However, using differential decoding for removing the phase ambiguity inevitably results in a performance degradation of 3dB, which compares unfavourably with the 22.4% of training–related transmission overhead that we could save in the case of GSM transmissions, for example, by eliminating the training sequence. We found evidence that in the case of turbo–PSP, initialising the channel estimate using the PSP equaliser alone, i.e. non–iteratively, was adequate for removing the sign–ambiguity, when a low number of known pilot symbols were employed. The same technique can be applied to any blind equalisers, since this type of phase–ambiguity removal does not require any assumption concerning the type of the equaliser, other than its ability to converge to a sign–reversed equilibrium. Thus, we can eliminate the differential encoding and yet maintain a good performance by using a low number of known pilot symbols, which was found to account for about 1.7% of the total throughput in the case of turbo–PSP for the transmission scenarios of Section 4.5. This is equivalent to a mere 0.07 dB SNR performance degradation, which constitutes a good trade–off between the achievable performance and the required bandwidth.

Again, blind equalisation offers certain benefits in comparison to trained equalisation. However, further research is required in a range of areas, since some of the properties of even the best–understood blind equalisation algorithms, such as the CMA are yet to be clarified. This is particularly so in the context of fading channels.

Potential areas of further research related to this study can be summarised as follows:

- Investigation of the Bussgang equalisers' error surface characteristics, with the aim of providing better initialisation strategies, which are capable of configuring the equaliser in initial states that are far from points of undesirable equilibria.
- Design of Bussgang equalisers having automatically controlled parameters.
- Study of the DFE–CMA in conjunction with different feedback sections, which can improve the achievable performance.
- Study of the convergence properties of the turbo–PSP.
- Contrive an algorithm for detecting the optimum number of trellis states to be retained.
- Theoretical evaluation of the optimum algorithm designed for controlling the number of iterations of the turbo–PSP, according to the prevalent channel condition.
- Application of channel decoding information for assisting the operation of Bussgang equalisers as implicit training, in order to obtain a low–complexity iterative algorithm.
- Evaluation of the possibility of replacing the convolutional coding by block coding in the turbo–PSP.
- Application of the equalisers studied in multiuser detectors.

In this treatise various combinations of blind equalisers and separate channel codecs or coded modulation schemes have been proposed, although their research is by no means a closed chapter. These blind techniques have also found a plethora of applications in the context of blind mmultiuser detectors and this area poses a range of further research challenges.

Appendix A

Complexity Estimates of the Equaliser Algorithms

In this appendix, the complexity of each of the equalisers studied in Chapters 1–5 is estimated in terms of the number of real multiplications and additions required at each symbol interval. Additionally, an estimate of the associated storage requirements is given.
First of all we provide an overview of the number of computations required by the addition/substraction and multiplication/division of complex numbers :

- Real x Complex

$$a \cdot (c + j \cdot d) = a \cdot c + j \cdot a \cdot d$$

  Multiplications : 2
  Additions : 0

- Real + Complex

$$a + (c + j \cdot d) = (a + c) + j \cdot d$$

  Multiplications : 0
  Additions : 1

- Complex x Complex

$$(a + j \cdot b) \cdot (c + j \cdot d) = a \cdot c - b \cdot d + j \cdot (a \cdot d + b \cdot c)$$

Multiplications : 4

Additions : 2

- Complex + Complex

$$(a + j \cdot b) + (c + j \cdot d) = (a + c) + j \cdot (b + d)$$

Multiplications : 0

Additions : 2

- Real / Complex

$$a/(c + j \cdot d) = \frac{a \cdot c}{c^2 + d^2} - j \cdot \frac{a \cdot d}{c^2 + d^2}$$

Multiplications : 6

Additions : 1

- Complex / Complex

$$(a + j \cdot b)/(c + j \cdot d) = \frac{a \cdot c + b \cdot d}{c^2 + d^2} + j \cdot \frac{b \cdot c - a \cdot d}{c^2 + d^2}$$

Multiplications : 8

Additions : 3

We can now count the number of additions and multiplications involved in each of the Bussgang algorithms of Chapter 1. In all these algorithms the extraction of $z(n)$ from $\mathbf{y}(n)$ and $\mathbf{c}$ from the relationship $z(n) = \mathbf{c}^T\mathbf{y}(n)$ needs $4 \cdot (2N + 1)$ multiplications and $2 \cdot (2N + 1) + 2 \cdot (2N) = 8N + 2$ additions. Additionally, they require $2 \cdot (2N + 1) = 4N + 2$ memory cells for storing the vector $\mathbf{c}$ and another $2 \cdot (2N + 1) = 4N + 2$ memory cells for the vector $\mathbf{y}(n)$. The associated number of operations have to be added to the number of operations required for the equaliser tap vector update of each separate algorithm, in order to provide the final complexity estimates of Table 1.1.

Appendix B

Extraction of the RLS Algorithm Update Equations

From Equation (1.29), which is repeated here for convenience, we have:

$$\mathbf{\Phi}(n) = \sum_{i=1}^{n} w^{n-i} \cdot \mathbf{y}(i) \cdot \mathbf{y}^H(i),$$

which can be rearranged as:

$$\mathbf{\Phi}(n) = w \left[\sum_{i=1}^{n-1} w^{n-1-i} \mathbf{y}(i)\mathbf{y}^H(i) \right] + \mathbf{y}(n)\mathbf{y}^H(n), \tag{B.1}$$

yielding:

$$\mathbf{\Phi}(n) = w \cdot \mathbf{\Phi}(n-1) + \mathbf{y}(n)\mathbf{y}^H(n). \tag{B.2}$$

In the same way, from Equation (1.31) we find that:

$$\mathbf{g}(n) = w \cdot \mathbf{g}(n-1) + \mathbf{y}(n)a^*(n). \tag{B.3}$$

Here we will make use of the *Matrix Inversion Lemma* of [14] or [15]. This Lemma states that if $\mathbf{A}$ and $\mathbf{B}$ are two positive–definite MxM matrices, while $\mathbf{D}$ is a positive–definite NxN matrix and $\mathbf{C}$ is a positive–definite MxN matrix, related by:

$$\mathbf{A} = \mathbf{B}^{-1} + \mathbf{C}\mathbf{D}^{-1}\mathbf{C}^H, \tag{B.4}$$

then the inverse of $\mathbf{A}$ can be expressed as:

$$\mathbf{A}^{-1} = \mathbf{B} - \mathbf{BC}(\mathbf{D} + \mathbf{C}^H\mathbf{BC})^{-1}\mathbf{C}^H\mathbf{B}. \tag{B.5}$$

Upon considering Equation (B.2) we have:

$$\mathbf{A} = \mathbf{\Phi}(n), \tag{B.6}$$

$$\mathbf{B} = w^{-1}\mathbf{\Phi}^{-1}(n-1), \tag{B.7}$$

$$\mathbf{C} = \mathbf{y}(n), \tag{B.8}$$

$$\mathbf{D} = 1. \tag{B.9}$$

If the matrix $\mathbf{\Phi}(n)$ is positive-definite, i.e. not singular, then we can invert it using the matrix inversion lemma of Equation (B.5) as follows:

$$\mathbf{\Phi}^{-1}(n) = w^{-1}\left(\mathbf{\Phi}^{-1}(n-1) - w^{-1}\frac{\mathbf{\Phi}^{-1}(n-1)\mathbf{y}(n)\mathbf{y}^H(n)\mathbf{\Phi}^{-1}(n-1)}{1+\mathbf{y}^H(n)\mathbf{\Phi}^{-1}(n-1)\mathbf{y}(n)}\right) \tag{B.10}$$

Setting

$$\mathbf{k}(n) = \frac{w^{-1}\mathbf{\Phi}^{-1}(n-1)\mathbf{y}(n)}{1+w^{-1}\mathbf{y}^H(n)\mathbf{\Phi}^{-1}(n-1)\mathbf{y}(n)} \tag{B.11}$$

Equation (B.10) becomes

$$\mathbf{\Phi}^{-1}(n) = w^{-1}\mathbf{\Phi}^{-1}(n-1) - w^{-1}\mathbf{k}(n)\mathbf{y}^H(n)\mathbf{\Phi}^{-1}(n-1). \tag{B.12}$$

Equation (1.32) gives the equaliser coefficient update algorithm, which is repeated here for convenience:

$$\hat{\mathbf{c}}^{(n)} = \mathbf{\Phi}^{-1}(n)\cdot\mathbf{g}(n).$$

Substituting $\mathbf{\Phi}^{-1}(n)$ from Equation (B.12) we obtain:

$$\hat{\mathbf{c}}^{(n)} = w^{-1}\mathbf{\Phi}^{-1}(n-1)\cdot\mathbf{g}(n) - w^{-1}\mathbf{k}(n)\mathbf{y}^H(n)\mathbf{\Phi}^{-1}(n-1)\cdot\mathbf{g}(n). \tag{B.13}$$

Now upon substituting Equation (B.3) into Equation (B.13) we arrive at:

$$\hat{\mathbf{c}}^{(n)} = \hat{\mathbf{c}}^{(n-1)} - \mathbf{k}(n)\left[a^*(n) - \mathbf{y}^H(n)\hat{\mathbf{c}}^{(n-1)}\right]. \tag{B.14}$$

Finally, upon setting

$$\xi(n) = a(n) - \left(\hat{\mathbf{c}}^{(n-1)}\right)^H\mathbf{y}(n) \tag{B.15}$$

we obtain:

$$\hat{\mathbf{c}}^{(n)} = \hat{\mathbf{c}}^{(n-1)} + \mathbf{k}(n)\xi^*(n). \tag{B.16}$$

We can now see that the recursive formula of Equations (B.16), (B.15), (B.11) and (B.12) becomes identical to Equations (1.33)–(1.36).

Appendix C

Vector and Complex Analysis

In this Appendix we give the definitions of differentiation with respect to complex vectors along with some of the associated properties. Based on these definitions, we then explain how we can minimise the cost–functions encountered in our theoretical equaliser–related considerations.

First of all we define the derivative of a complex function with respect to a complex number $z = x + jy$ as [15]:

$$\frac{\partial}{\partial z} = \frac{1}{2}\left(\frac{\partial}{\partial x} - j\frac{\partial}{\partial y}\right) \tag{C.1}$$

and

$$\frac{\partial}{\partial z^*} = \frac{1}{2}\left(\frac{\partial}{\partial x} + j\frac{\partial}{\partial y}\right) \tag{C.2}$$

From this definition we observe immediately that

$$\frac{\partial z}{\partial z} = 1 \tag{C.3}$$

$$\frac{\partial z^*}{\partial z} = 0 \tag{C.4}$$

$$\frac{\partial z}{\partial z^*} = 0 \tag{C.5}$$

$$\frac{\partial z^*}{\partial z^*} = 1. \tag{C.6}$$

We can now define the derivative of a complex function, or a vector of complex functions, with respect to a complex vector $\mathbf{z} = \mathbf{x} + j\mathbf{y}$ as [15] :

$$\frac{\partial}{\partial \mathbf{z}} = \frac{1}{2}\begin{bmatrix} \frac{\partial}{\partial x_1} - j\frac{\partial}{\partial y_1} \\ \frac{\partial}{\partial x_2} - j\frac{\partial}{\partial y_2} \\ \vdots \\ \frac{\partial}{\partial x_N} - j\frac{\partial}{\partial y_N} \end{bmatrix} \tag{C.7}$$

and

$$\frac{\partial}{\partial \mathbf{z}^*} = \frac{1}{2}\begin{bmatrix} \frac{\partial}{\partial x_1} + j\frac{\partial}{\partial y_1} \\ \frac{\partial}{\partial x_2} + j\frac{\partial}{\partial y_2} \\ \vdots \\ \frac{\partial}{\partial x_N} + j\frac{\partial}{\partial y_N} \end{bmatrix}, \tag{C.8}$$

where $x_1 \dots x_N$ and $y_1 \dots y_N$ are the real and imaginary vector elements, respectively. As before, we can observe that:

$$\frac{\partial \mathbf{z}}{\partial \mathbf{z}} = \mathbf{I} \tag{C.9}$$

$$\frac{\partial \mathbf{z}^*}{\partial \mathbf{z}} = \mathbf{O} \tag{C.10}$$

$$\frac{\partial \mathbf{z}}{\partial \mathbf{z}^*} = \mathbf{O} \tag{C.11}$$

$$\frac{\partial \mathbf{z}^*}{\partial \mathbf{z}^*} = \mathbf{I}, \tag{C.12}$$

where $\mathbf{I}$ and $\mathbf{O}$ are the identity and null matrices of order N respectively. We can now portray an important property, which we use for the minimisation of all of our cost–functions:

$$\frac{\partial \mathbf{a}^T \cdot \mathbf{b}}{\partial \mathbf{z}} = \frac{\partial \mathbf{b}}{\partial \mathbf{z}} \cdot \mathbf{a} + \frac{\partial \mathbf{a}}{\partial \mathbf{z}} \cdot \mathbf{b}, \tag{C.13}$$

where $\mathbf{a}, \mathbf{b}$ can be any M–diamensional vector. This property can be verified by a simple substitution of the definitions in Equations (C.1)–(C.13), applying the derivative product property of the complex numbers:

$$\frac{\partial a \cdot b}{\partial x} = a \cdot \frac{\partial b}{\partial x} + b \cdot \frac{\partial b}{\partial x} \tag{C.14}$$

We will now formulate the definition of the gradient vector [15] as:

$$\nabla_{\mathbf{z}} = \begin{bmatrix} \frac{\partial}{\partial x_1} + j\frac{\partial}{\partial y_1} \\ \frac{\partial}{\partial x_2} + j\frac{\partial}{\partial y_2} \\ \vdots \\ \frac{\partial}{\partial x_N} + j\frac{\partial}{\partial y_N} \end{bmatrix} \tag{C.15}$$

and

$$\nabla_{\mathbf{z}^*} = \begin{bmatrix} \frac{\partial}{\partial x_1} - j\frac{\partial}{\partial y_1} \\ \frac{\partial}{\partial x_2} - j\frac{\partial}{\partial y_2} \\ \vdots \\ \frac{\partial}{\partial x_N} - j\frac{\partial}{\partial y_N} \end{bmatrix}. \tag{C.16}$$

It is clear that the properties of the derivatives with respect to the vector $\mathbf{z}$ are still valid, only now $\mathbf{z}$ and $\mathbf{z}^*$ are mutually exchanged. In mathematical terms, the relationship between the two definitions in Equations (C.1) and (C.7) is given as:

$$\nabla_{\mathbf{z}} = 2\frac{\partial}{\partial \mathbf{z}^*}. \tag{C.17}$$

C.1 Gradient Vector Properties

The specific properties of the gradient vector to be used in our discourse are:

$$\nabla_{\mathbf{z}}\mathbf{z} = \mathbf{O}, \tag{C.18}$$

$$\nabla_{\mathbf{z}}\mathbf{z}^* = \mathbf{I}, \tag{C.19}$$

$$\nabla_{\mathbf{z}^*}\mathbf{z} = \mathbf{I}, \tag{C.20}$$

$$\nabla_{\mathbf{z}^*}\mathbf{z}^* = \mathbf{O}, \tag{C.21}$$

$$\nabla_{\mathbf{z}}(\mathbf{a}^T \cdot \mathbf{b}) = \nabla_{\mathbf{z}}\mathbf{b} \cdot \mathbf{a} + \nabla_{\mathbf{z}}\mathbf{a} \cdot \mathbf{b}. \tag{C.22}$$

We will also prove the validity of a few more properties, which we will need in order to derive the analytical expressions of the cost–functions' derivatives with respect to the equaliser coefficient vector:

$$\nabla_{\mathbf{z}}(f(\mathbf{a}))^k = \begin{bmatrix} \frac{\partial}{\partial x_1} + j\frac{\partial}{\partial y_1} \\ \frac{\partial}{\partial x_2} + j\frac{\partial}{\partial y_2} \\ \vdots \\ \frac{\partial}{\partial x_N} + j\frac{\partial}{\partial y_N} \end{bmatrix} (f(\mathbf{a}))^k = \begin{bmatrix} \frac{\partial (f(\mathbf{a}))^k}{\partial x_1} + j\frac{\partial (f(\mathbf{a}))^k}{\partial y_1} \\ \frac{\partial (f(\mathbf{a}))^k}{\partial x_2} + j\frac{\partial (f(\mathbf{a}))^k}{\partial y_2} \\ \vdots \\ \frac{\partial (f(\mathbf{a}))^k}{\partial x_N} + j\frac{\partial}{\partial y_N} \end{bmatrix} =$$

$$= k \cdot (f(\mathbf{a}))^{k-1} \begin{bmatrix} \frac{\partial f(\mathbf{a})}{\partial x_1} + j\frac{\partial f(\mathbf{a})}{\partial y_1} \\ \frac{\partial f(\mathbf{a})}{\partial x_2} + j\frac{\partial f(\mathbf{a})}{\partial y_2} \\ \vdots \\ \frac{\partial f(\mathbf{a})}{\partial x_N} + j\frac{\partial f(\mathbf{a})}{\partial y_N} \end{bmatrix} = k \cdot (f(\mathbf{a}))^{k-1} \nabla_{\mathbf{z}} f(\mathbf{a}), \tag{C.23}$$

$$\begin{aligned} \nabla_{\mathbf{z}} |\mathbf{a}^T \cdot \mathbf{b}| &= \nabla_{\mathbf{z}} \sqrt{|\mathbf{a}^T \cdot \mathbf{b}|^2} \\ &\overset{(C.23)}{=} \frac{1}{2|\mathbf{a}^T \cdot \mathbf{b}|} \nabla_{\mathbf{z}} |\mathbf{a}^T \cdot \mathbf{b}|^2 \\ &= \frac{1}{2|\mathbf{a}^T \cdot \mathbf{b}|} \nabla_{\mathbf{z}} (\mathbf{a}^T \cdot \mathbf{b})(\mathbf{a}^T \cdot \mathbf{b})^* \\ &\overset{(C.22)}{=} \frac{1}{2|\mathbf{a}^T \cdot \mathbf{b}|} \left((\mathbf{a}^T \cdot \mathbf{b}) \nabla_{\mathbf{z}} (\mathbf{a}^H \cdot \mathbf{b}^*) + (\mathbf{a}^H \cdot \mathbf{b}^*) \nabla_{\mathbf{z}} (\mathbf{a}^T \cdot \mathbf{b}) \right) \\ &= \frac{1}{2|\mathbf{a}^T \cdot \mathbf{b}|} \left((\mathbf{a}^T \cdot \mathbf{b}) \nabla_{\mathbf{z}} (\mathbf{a}^H \cdot \mathbf{b}^*) + (\mathbf{a}^H \cdot \mathbf{b}^*) \nabla_{\mathbf{z}} (\mathbf{a}^T \cdot \mathbf{b}) \right) \\ &= \frac{(\mathbf{a}^T \cdot \mathbf{b}) \left(\nabla_{\mathbf{z}} \mathbf{b}^* \cdot \mathbf{a}^* + \nabla_{\mathbf{z}} \mathbf{a}^* \cdot \mathbf{b}^* \right) + (\mathbf{a}^H \cdot \mathbf{b}^*) \left(\nabla_{\mathbf{z}} \mathbf{a} \cdot \mathbf{b} + \nabla_{\mathbf{z}} \mathbf{b} \cdot \mathbf{a} \right)}{2|\mathbf{a}^T \cdot \mathbf{b}|}. \end{aligned} \tag{C.24}$$

These properties are useful for the minimisation of the blind equaliser cost–functions.

In the next sections we will deal with the individual algorithms separately and we will outline the cost–function minimisation procedure for each one of them. Three useful relationships which we will use in the next sections accrue from the direct application of Equations (C.23) and (C.24), yielding:

$$\begin{aligned} \nabla_{\mathbf{c}} |z(n)| &= \nabla_{\mathbf{c}} \sqrt{|z(n)|^2} \\ &= \frac{1}{2\sqrt{|z(n)|^2}} \nabla_{\mathbf{c}} |z(n)|^2 \\ &= \frac{1}{2|z(n)|} \nabla_{\mathbf{c}} (\mathbf{c}^T \mathbf{y}(n))(\mathbf{c}^H \mathbf{y}^*(n)) \\ &= \frac{1}{2|z(n)|} \mathbf{c}^T \mathbf{y}(n) \mathbf{y}^*(n) \\ &= \frac{z(n) \cdot \mathbf{y}^*(n)}{2|z(n)|}, \end{aligned} \tag{C.25}$$

$$\begin{aligned}\nabla_{\mathbf{c}} Re\{z(n)\} &= \nabla_{\mathbf{c}} \frac{z(n)+z^*(n)}{2} \\ &= \nabla_{\mathbf{c}} \frac{\mathbf{c}^T\mathbf{y}(n)+\mathbf{c}^H\mathbf{y}^*(n)}{2} \\ &\overset{(C.18)-(C.22)}{=} \frac{1}{2}\cdot\mathbf{y}^*(n),\end{aligned} \tag{C.26}$$

$$\begin{aligned}\nabla_{\mathbf{c}} Re\{z(n)\} &= \nabla_{\mathbf{c}} \frac{z(n)-z^*(n)}{2j} \\ &= \nabla_{\mathbf{c}} \frac{\mathbf{c}^T\mathbf{y}(n)-\mathbf{c}^H\mathbf{y}^*(n)}{2j} \\ &\overset{(C.18)-(C.22)}{=} \frac{j}{2}\cdot\mathbf{y}^*(n),\end{aligned} \tag{C.27}$$

where $z(n) = \mathbf{c}^T\mathbf{y}(n)$ is the equalised signal, expressed as the product of two vectors, namely that of the equaliser tap vector $\mathbf{c}$ and the received signal $\mathbf{y}(n)$, which effectively forms the convolution of the received signal with the impulse response of the equaliser.

C.2 Differentiation Method: An Illustrative Example

During the minimisation of the cost–functions we will only make use of the gradient vector of Equation (C.7) but not of the vector derivative of Equation (C.1). The reason that we use the specific formula of Equation (C.15) for the definition of the derivative of the cost–function with respect to a vector is that, in our analysis we aim for determining the derivatives of the cost–functions, so that, the steepest descent coefficient update algorithm of Equation (1.16) leads to the correct point of equilibrium. By defining the derivative of our cost–function as:

$$\frac{\partial J(n)}{\partial c_i} = \frac{\partial J(n)}{\partial Re\{c_i\}} + j\cdot\frac{\partial J(n)}{\partial Im\{c_i\}}$$

we make use of the iterative process :

$$x_{n+1} = x_n - \lambda\frac{\partial J}{\partial x} \tag{C.28}$$

in which we are searching for the minimum of the cost–function J with respect to the real variable x. We note that the minus sign in front of λ indicates that we are looking for a minimum. By contrast, a plus sign would generally direct the iterative process towards the function's maxima. Therefore, the real part of the equaliser coefficients in Equation (1.16) is updated using the derivative of $J(n)$, as if it was a function of a real variable. The same is valid for the imaginary part. If, instead of a plus sign, we had a minus sign in front of the imaginary part, then this would lead to incorrect coefficient

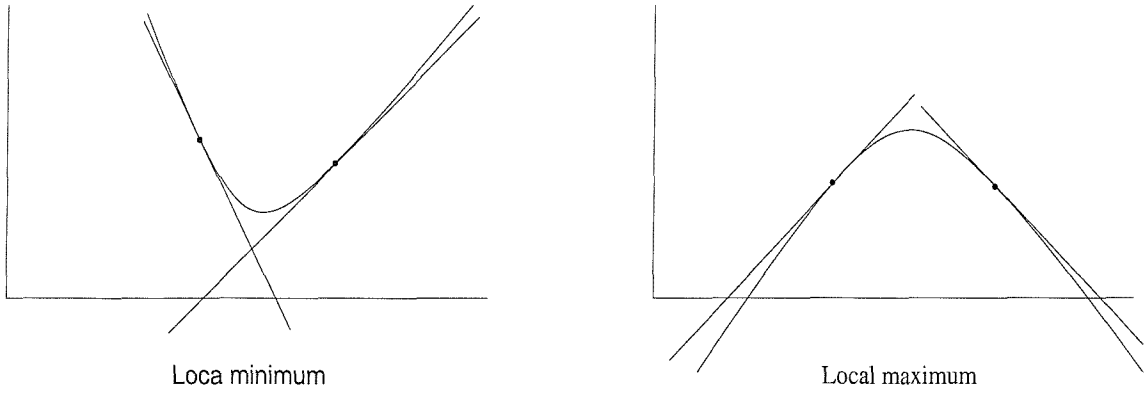


Figure C.1: Illustration of the steepest descent technique for a function of one variable: (a) in the region of a local minimum and (b) in the region of a local maximum.

adjustment, since in this case the algorithm would be approaching the maxima of $J(n)$. This explains the motivation behind employing the gradient vector for the blind equaliser's coefficient adjustment. A graphical illustration of this is given in Figure C.1 for the simple case of a function depending on single variable. In Figure C.1(a), in the region of a local minimum, the algorithm performs subtraction of the derivative of the function, with respect to the independent variable, from the estimated value of the local minimum. Accordingly, when the estimate is at the left of the minimum, the update value is positive, since the derivative is negative, and the new estimated value is closer to the minimum. The opposite is true, when the estimated value is to the right of the minimum, so that the estimated value, again, becomes closer to the minimum. By contrast, in Figure C.1(b), in the vicinity of a local maximum, the algorithm again performs subtraction of the derivative of the function, with respect to the independent variable, from the estimated value of the local maximum. Correspondingly, when the estimate is at the left of the maximum, the update value is negative, since the derivative is positive, and hence the new estimated value becomes closer to the maximum. Again, the opposite is true, when the estimated value is to the right of the maximum, so that the estimated value again becomes closer to the maximum. The extension of this procedure to the case of many variables is straightforward.

Appendix D

Cost–Function Minimisations

In this appendix we derive the update equations for the Bussgang algorithms of Chapter 1 by taking the gradient of their cost–function with respect to the equaliser vector. This is carried out by applying the properties of the complex analysis summarised in Appendix C and by performing the necessary calculations.

D.1 Sato's Cost–Function Minimisation

Sato's cost–function was defined in Equation (1.66) as:

$$J^S(n) = E\left[\frac{1}{2}\cdot|z(n)|^2 - \gamma\cdot(|Re\{z(n)\}| + |Im\{z(n)\}| - \gamma)\right]$$

The equation represents the statistical average value of the stochastic variables' expression. In order to find the local minima of the cost function we have to differentiate the expression with respect to the equaliser tap vector $\mathbf{c}$. In doing so, the expectation may exchange its precedence with differentiation. In this case, we can omit the expectation for simplicity. The expectation is still there, however as far as the tap–update algorithm is concerned, it is not taken into account since this algorithm relies on instant rather than average values. If we do so, then the cost–function's gradient vector, with respect to $\mathbf{c}$, is given by:

$$\begin{aligned}
\nabla_{\mathbf{c}} J^S(n) &= \nabla_{\mathbf{c}}(\frac{1}{2}\cdot|z(n)|^2 - \gamma\cdot(|Re\{z(n)\}| + |Im\{z(n)\}|)) \\
&= |z(n)|\nabla_{\mathbf{c}}|z(n)| - \\
&\quad \gamma\left(sgn(Re\{z(n)\})\cdot\nabla Re\{z(n)\} + sgn(Im\{z(n)\})\cdot\nabla Im\{z(n)\}\right) \\
&\overset{(C.25),(C.26),(C.27)}{=} |z(n)|\frac{z(n)\cdot\mathbf{y}^*(n)}{2|z(n)|} -
\end{aligned}$$

$$\begin{aligned} & \gamma \cdot \left(\frac{1}{2} \cdot sgn(Re\{z(n)\}) \mathbf{y}^*(n) + \frac{j}{2} \cdot sgn(Im\{z(n)\}) \mathbf{y}^*(n) \right) \\ = \quad & \frac{1}{2} \cdot z(n) \mathbf{y}^*(n) - \frac{\gamma}{2} \cdot (sgn(Re\{z(n)\}) + j \cdot sgn(Im\{z(n)\})) \mathbf{y}^* \\ = \quad & \frac{1}{2} \cdot (z(n) - \gamma \cdot csgn(z(n))) \mathbf{y}^*(n), \end{aligned} \tag{D.1}$$

where $sgn(re^{j\phi}) = e^{j\phi}$ is the complex sign function and $csgn(x+jy) = sgn(x)+j \cdot sgn(y)$. Furthermore we have applied the following property of the gradient of a real function:

$$\nabla_{\mathbf{x}} |f(\mathbf{x})| = sgn(f(\mathbf{x})) \cdot \nabla_{\mathbf{x}} f(\mathbf{x}). \tag{D.2}$$

Recall that Equation (D.1) is exactly the equaliser coefficient update term of Equation (1.69). This relationship can be justified by considering that the $csgn()$ function defined in Equation (1.68), which was also reepeated above, separates the real and imaginary parts of the equalised signal $z(n) = x+j \cdot y$ in Equation (1.70) such that they can be equalised independently. We will now determine the constant parameter γ following the approach of Godard [4]. Our goal is to set the mean value of the error term $\epsilon^{Sato}(n)$ in Equation (1.69) to zero by appropriately choosing γ. This implies that we have:

$$E[\mathbf{y}^*(n) z(n) - \gamma \cdot \mathbf{y}^*(n) csgn(z(n))] = 0. \tag{D.3}$$

We recall that each of the received signal vector's elements $y(n)$ is given by the convolution of the transmitted sequence with the channel's impulse response [1] $\{h_n\}$, as follows:

$$y(n) = \sum_k a(k) \cdot h_{n-k}. \tag{D.4}$$

Therefore, Equation (D.3) can be rewritten as:

$$E[z(n) \sum_k a^*(k) \cdot h^*_{n-k}] = \gamma \cdot E[(sgn(Re\{z(n)\}) + j \cdot sgn(Im\{z(n)\})) \sum_k a^*(k) \cdot h^*_{n-k}]. \tag{D.5}$$

In the state where equalisation is perfect, which is the scenario we are trying to accomplish, the equalised signal is equal to the transmitted signal, i.e. $z(n) = a(n)$. This is the desired state that the equaliser reaches after its convergence. With the equaliser at this state, taking into account that the transmitted sequence is i.i.d., this equation reduces to:

$$E[a(n) \cdot a^*(n) \cdot h^*_o] = \gamma \cdot E[(sgn(Re\{a(n)\}) + j \cdot sgn(Im\{a(n)\})) \cdot a^*(n) \cdot h^*_o]. \tag{D.6}$$

[1] Transmitter and receiver filtering is ignored here, since we can consider their influence on the received signal to be included in the channel's impulse response

Noting that h_o is a constant, after some manipulations this relationship finally reduces to:

$$E[|a(n)|^2] = \gamma \cdot E[|Re\{a(n)\}| + |Im\{a(n)\}|]. \tag{D.7}$$

For the practical scenario of symmetric constellations, i.e. when we have $E[|Re\{a(n)\}|] = E[|Im\{a(n)\}|]$, according to Equation (D.7) the parameter γ is set to:

$$\gamma = \frac{E[|a(n)|^2]}{E[|Re\{a(n)\}| + |Im\{a(n)\}|]} = \frac{E[(Re\{a(n)\})^2]}{E[|Re\{a(n)\}|]} = \frac{E[(Im\{a(n)\})^2]}{E[|Im\{a(n)\}|]}. \tag{D.8}$$

This equation is similar to the relevant CMA formula of Equation (1.77) for $p = 1$. The difference is that while the CMA's constant of $R_2 = \frac{E[|a(n)|^4]}{E[|a(n)|^2]}$ is a function of the transmitted signal magnitudes, the Sato parameter γ in Equation (D.8) is a function of the real (or the imaginary) part of the transmitted signal. This difference is reflected in the cost–functions of the two algorithms, which are given in Equations (1.74) and (1.66) respectively, where we can observe that the CMA equalises the magnitude of the received signal, while Sato's algorithm equalises the real and imaginary parts of it.

D.2 CMA Cost–Function Minimisation

The CMA's cost–function was defined in Equation (1.74), which is repeated here for convenience:

$$D^{(p,q)}(n) = \frac{1}{pq} E[||z(n)|^p - R_p|^q]$$

Again, if we omit the expectation, as we did in the previous sectionn, then its gradient vector, with respect to $\mathbf{c}$ is given by:

$$\begin{aligned}
\nabla_{\mathbf{c}} D^{(p,q)}(n) &= \frac{1}{pq} \nabla_{\mathbf{c}} ||z(n)|^p - R_p|^q \\
&\overset{(C.23)}{=} \frac{1}{p} \cdot ||z(n)|^p - R_p|^{q-1} \nabla_{\mathbf{c}} ||z(n)|^p - R_p| \\
&= \frac{1}{p} \cdot ||z(n)|^p - R_p|^{q-1} \cdot sgn(|z(n)|^p - R_p) \nabla_{\mathbf{c}} |\mathbf{c}^T \mathbf{y}(n)|^p \\
&\overset{(C.23)}{=} (|z(n)|^p - R_p)^{q-1} |z(n)|^{p-1} \nabla_{\mathbf{c}} |z(n)| \\
&= (|z(n)|^p - R_p)^{q-1} |z(n)|^{p-1} \nabla_{\mathbf{c}} |\mathbf{c}^T \mathbf{y}(n)| \\
&\overset{(C.22)}{=} (|z(n)|^p - R_p)^{q-1} |z(n)|^{p-1} \frac{(\mathbf{c}^T \mathbf{y}(n)) \mathbf{y}^*(n)}{2|\mathbf{c}^T \mathbf{y}(n)|} \\
&= \frac{1}{2} (|z(n)|^p - R_p)^{q-1} |z(n)|^{p-2} \cdot z(n) \cdot \mathbf{y}^*(n),
\end{aligned} \tag{D.9}$$

where $sgn()$ denotes the real sign function, since its argument only takes real values. For $q = 2$ this relationship becomes:

$$\nabla_{\mathbf{c}} D^{(p,2)}(n) = \frac{1}{2}(|z(n)|^p - R_p) \cdot |z(n)|^{p-2} \cdot z(n) \cdot \mathbf{y}^*(n). \tag{D.10}$$

Finally, for $p = 1$ we obtain Sato's correction term of Equation (1.69),

$$\nabla_{\mathbf{c}} D^{(1,2)}(n) = \frac{1}{2}(z(n) - R_p \cdot sgn(z(n))) \cdot \mathbf{y}^*(n),$$

while for $p = 2$ we obtain the classical CMA correction term of Equation (1.76) in the following form:

$$\nabla_{\mathbf{c}} D^{(p,2)}(n) = \frac{1}{2}(|z(n)|^2 - R_p) \cdot z(n) \cdot \mathbf{y}^*(n). \tag{D.11}$$

Again, these are exactly the equaliser coefficients update terms of Equations (1.69) and (1.76), respectively.

D.3 Signed–CMA Cost–Function Minimisation

The Signed–CMA algorithm's cost–function is defined as (1.78):

$$J^{S-CMA}(n) = E[||Re\{z(n)\}| + |Im\{z(n)\}| - R_S|].$$

If we omit the expectation, then the corresponding gradient vector with respect to $\mathbf{c}$, is expressed as:

$$\begin{aligned}
\nabla_{\mathbf{c}} J^{S-CMA}(n) &= \nabla_{\mathbf{c}} \, ||Re\{z(n)\}| + |Im\{z(n)\}| - R_S| \\
&= sgn(|Re\{z(n)\}| + |Im\{z(n)\}| - R_S) \cdot \nabla_{\mathbf{c}} \left(|Re\{z(n)\}| + |Im\{z(n)\}| - R_S\right) \\
&= sgn(|Re\{z(n)\}| + |Im\{z(n)\}| - R_S) \cdot (\nabla_{\mathbf{c}}|Re\{z(n)\}| + \nabla_{\mathbf{c}}|Im\{z(n)\}|) \\
&= sgn(|Re\{z(n)\}| + |Im\{z(n)\}| - R_S) \cdot \\
&\quad (sgn(Re\{z(n)\}) \cdot \nabla_{\mathbf{c}} Re\{z(n)\} + sgn(Im\{z(n)\}) \cdot \nabla_{\mathbf{c}} Im\{z(n)\}) \\
&= sgn(|Re\{z(n)\}| + |Im\{z(n)\}| - R_S) \cdot \\
&\quad \left(sgn(Re\{z(n)\}) \cdot \nabla_{\mathbf{c}} \frac{z(n) + z^*(n)}{2} + sgn(Im\{z(n)\}) \cdot \nabla_{\mathbf{c}} \frac{z(n) - z^*(n)}{2j}\right) \\
&= \frac{1}{2} sgn(|Re\{z(n)\}| + |Im\{z(n)\}| - R_S) \cdot \\
&\quad \Big(sgn(Re\{z(n)\}) \cdot \nabla_{\mathbf{c}} (\mathbf{c}^T \mathbf{y}(n) + \mathbf{c}^H \mathbf{y}^*(n)) + \\
&\quad + j \cdot sgn(Im\{z(n)\}) \cdot \nabla_{\mathbf{c}} (\mathbf{c}^T \mathbf{y}(n) + \mathbf{c}^H \mathbf{y}^*(n))\Big) \\
&= \frac{1}{2} sgn(|Re\{z(n)\}| + |Im\{z(n)\}| - R_S) \cdot (sgn(Re\{z(n)\}) \cdot \mathbf{y}^*(n) +
\end{aligned}$$

$$
\begin{aligned}
& j \cdot sgn(Im\{z(n)\}) \cdot \mathbf{y}^*(n)) \\
= \; & \frac{1}{2} sgn(|Re\{z(n)\}| + |Im\{z(n)\}| - R_S) \cdot csgn(z(n)) \cdot \mathbf{y}^*(n), \qquad \text{(D.12)}
\end{aligned}
$$

where $sgn()$ denotes the real sign function, since its argument only takes real values, while we have $csgn(z) = sgn(Re\{z\}) + jsgn(Im\{z\})$, as defined in Equation (1.68). Equation (D.12) is exactly the equaliser coefficient update term of Equation (1.79).

D.4 MCMA Cost–Function Minimisation

The MCMA's cost–function was defined in Equation (1.81) as:

$$J^{MCMA}(n) = E[((Re\{z(n)\})^2 - R_{p,R})^2 + ((Im\{z(n)\})^2 - R_{p,I})^2].$$

If we omit the expectation, then the corresponding gradient vector, with respect to $\mathbf{c}$ can be expressed as:

$$
\begin{aligned}
\nabla_{\mathbf{c}} J^{MCMA}(n) &= \nabla_{\mathbf{c}} \left(((Re\{z(n)\})^2 - R_{p,R})^2 + ((Im\{z(n)\})^2 - R_{p,I})^2 \right) \\
&= \nabla_{\mathbf{c}} ((Re\{z(n)\})^2 - R_{p,R})^2 + \nabla_{\mathbf{c}} ((Im\{z(n)\})^2 - R_{p,I})^2 \\
&= 2 \cdot ((Re\{z(n)\})^2 - R_{p,R}) \nabla_{\mathbf{c}} ((Re\{z(n)\})^2 - R_{p,R}) + \\
&\quad 2 \cdot ((Im\{z(n)\})^2 - R_{p,I}) \nabla_{\mathbf{c}} ((Im\{z(n)\})^2 - R_{p,I}) \\
&= 4 \cdot ((Re\{z(n)\})^2 - R_{p,R}) \cdot Re\{z(n)\} \nabla_{\mathbf{c}} Re\{z(n)\} + \\
&\quad 4 \cdot ((Im\{z(n)\})^2 - R_{p,I}) \cdot Im\{z(n)\} \nabla_{\mathbf{c}} Im\{z(n)\} \\
&= 4 \cdot ((Re\{z(n)\})^2 - R_{p,R}) \cdot Re\{z(n)\} \nabla_{\mathbf{c}} \frac{z(n) + z^*(n)}{2} + \\
&\quad 4 \cdot ((Im\{z(n)\})^2 - R_{p,I}) \cdot Im\{z(n)\} \nabla_{\mathbf{c}} \frac{z(n) - z^*(n)}{2j} \\
&= 2 \cdot ((Re\{z(n)\})^2 - R_{p,R}) \cdot Re\{z(n)\} \nabla_{\mathbf{c}} (\mathbf{c}^T \mathbf{y}(n) + \mathbf{c}^H \mathbf{y}^*(n)) - \\
&\quad 2j \cdot ((Im\{z(n)\})^2 - R_{p,I}) \cdot Im\{z(n)\} \nabla_{\mathbf{c}} (\mathbf{c}^T \mathbf{y}(n) - \mathbf{c}^H \mathbf{y}^*(n)) \\
&= 2 \cdot ((Re\{z(n)\})^2 - R_{p,R}) \cdot Re\{z(n)\} \cdot \mathbf{y}^*(n) + \\
&\quad 2j \cdot ((Im\{z(n)\})^2 - R_{p,I}) \cdot Im\{z(n)\} \cdot \mathbf{y}^*(n) \\
&= 2 \cdot (((Re\{z(n)\})^2 - R_{p,R}) \cdot Re\{z(n)\} + \\
&\quad j \cdot ((Im\{z(n)\})^2 - R_{p,I}) \cdot Im\{z(n)\}) \cdot \mathbf{y}^*(n), \qquad \text{(D.13)}
\end{aligned}
$$

where $sgn()$ denotes the real sign function, since its argument only takes real values, while $csgn(z) = sgn(Re\{z\}) + jsgn(Im\{z\})$, as defined in Equation (1.68). Equation (D.13) corresponds to the

equaliser coefficient update term of Equation (1.79).

Appendix E

Convergence Points Estimation

In this appendix we estimate the convergence points of the Bussgang algorithms of Chapter 1, based on the example of the CMA assisted blind equaliser, by taking the gradient of the corresponding cost–function with respect to the total system's transfer function vector $\mathbf{t}$ and averaging the result over the possible values of the transmitted signal, which is assumed to consist of a set of independent identically distributed points of a QAM constellation. As we will see, it is essential that this property holds for these equalisers. Naturally, the vector and complex analysis properties of Appendix C are also used.

E.1 The Constant Modulus Algorithm's Convergence Points

In this section we derive the points of equilibrium for the constant modulus algorithm. The approach we follow is similar to Wesolowksy's [102]. First we recall Equation (D.11), which gives the gradient of the cost–function for the CMA with respect to the equaliser's tap vector $\mathbf{c}$, yielding:

$$\nabla_{\mathbf{c}} J(n) = \frac{1}{2}(|z(n)|^2 - R_2) \cdot z(n) \cdot \mathbf{y}^*(n).$$

The points where this function becomes zero are the possible local minima and we will attempt to find these points. In order to achieve this, we rewrite the gradient of the cost–function of Equation (1.74) with respect to the equaliser tap vector as:

$$\begin{aligned}
\nabla_{\mathbf{c}} J(n) &= 0 \Leftrightarrow \\
\frac{1}{2}(|z(n)|^2 - R_2) \cdot z(n) \cdot \mathbf{y}^*(n) &= 0 \Rightarrow \\
z(n) \cdot z^*(n) \cdot z(n) \cdot \mathbf{y}^*(n) &= R_2 \cdot z(n) \cdot \mathbf{y}^*(n) \Rightarrow \\
(\mathbf{t}^T \cdot \mathbf{a}(n)) \cdot (\mathbf{t}^H \cdot \mathbf{a}^*(n)) \cdot (\mathbf{t}^T \cdot \mathbf{a}(n)) \cdot (\mathbf{H}^* \cdot \mathbf{a}^*(n)) &= R_2 \cdot (\mathbf{t}^T \cdot \mathbf{a}(n)) \cdot (\mathbf{H}^* \cdot \mathbf{a}^*(n)) \Rightarrow
\end{aligned}$$

$$\begin{aligned}\mathbf{H}^* \cdot (\mathbf{a}^*(n) \cdot \mathbf{a}^T(n)) \cdot (\mathbf{t} \cdot \mathbf{t}^H) \cdot (\mathbf{a}^*(n) \cdot \mathbf{a}^T(n)) \cdot \mathbf{t}^T &= R_2 \cdot \mathbf{H}^* \cdot (\mathbf{a}^*(n) \cdot \mathbf{a}^T(n)) \cdot \mathbf{t}^T \Rightarrow \\ \mathbf{H}^* \cdot \mathbf{A} \cdot \mathbf{T} \cdot \mathbf{A} \cdot \mathbf{t} &= R_2 \cdot \mathbf{H}^* \cdot \mathbf{A} \cdot \mathbf{t}, \end{aligned} \tag{E.1}$$

where $\mathbf{A} = \mathbf{a}^*(n) \cdot \mathbf{a}^T(n)$ and $\mathbf{T} = \mathbf{t} \cdot \mathbf{t}^H$. Taking the average of Equation (E.1) over all the possible data sequences $\mathbf{a}(n)$, under the assumption that the input sequence is independent identically distributed (i.i.d.), we can readily see that matrix $\mathbf{A}$ becomes:

$$E[\mathbf{A}] = E[|a(n)|^2] \cdot \mathbf{I} \tag{E.2}$$

where $\mathbf{I}$ is the unity matrix. After a few manipulations we can also see that the product $\mathbf{ATA}$ has the following mean value:

$$E[\mathbf{ATA}] = (E[|a(n)|^4] - 2 \cdot (E[|a(n)|^2)^2]) \cdot \mathbf{T_d}\mathbf{T_d} + (E[|a(n)|^2])^2 \cdot (\mathbf{t}^H\mathbf{t} \cdot \mathbf{I} + \mathbf{t}\mathbf{t}^H), \tag{E.3}$$

where $\mathbf{T_d} = diag(t_{-N_1-L_1}, \cdots, t_{N_2+L_2})$ and the terms $E[\mathbf{a}^2(n)]$ as well as $E[(\mathbf{a}^*(n))^2]$ have been assumed to be zero, given the symmetry of the QAM constellation concerned. We now substitute these mean values into Equation (E.1) in order to arrive at:

$$\mathbf{H}^* \cdot ((E[|a(n)|^4] - 2 \cdot (E[|a(n)|^2)^2]) \cdot \mathbf{T_d}\mathbf{T_d} + (E[|a(n)|^2])^2 \cdot (\mathbf{t}^H\mathbf{t} \cdot \mathbf{I} + \mathbf{t}\mathbf{t}^H) - R_2 \cdot E[|a(n)|^2] \cdot \mathbf{I}) \cdot \mathbf{t} = 0. \tag{E.4}$$

If we stipulate the assumption that the channnel matrix $\mathbf{H}^*$ is left–invertible, then the system of Equation (E.4) reduces to:

$$((E[|a(n)|^4] - 2 \cdot (E[|a(n)|^2)^2]) \cdot \mathbf{T_d}\mathbf{T_d} + (E[|a(n)|^2])^2 \cdot (\mathbf{t}^H\mathbf{t} \cdot \mathbf{I} + \mathbf{t}\mathbf{t}^H) - R_2 \cdot E[|a(n)|^2] \cdot \mathbf{I}) \cdot \mathbf{t} = 0. \tag{E.5}$$

More specifically, the left–invertible assumption implies that the rows of the channel matrix are linearly independent. However, closer scrutiny of the structure of this matrix reveals that this assumption is not generally true. Nevertheless, for a high number of equaliser taps, the channel matrix becomes square shaped and invertible. In this case the above assumption is valid. Naturally, this is an ideal situation, but when the number of equaliser taps is sufficiently high, then the approximation is true.

In this case, from Equation (E.5) we obtain a set of equations in the form of:

$$\left(E[|a(n)|^4] \cdot (|t_i|^2 - 1) + 2 \cdot (E[|a(n)|^2])^2 \cdot \sum_{j \neq i} |t_j|^2 \right) \cdot t_i = 0. \tag{E.6}$$

From these equations we can see that there are two possibilities for Equation (E.6) to hold, namely:

- $t_i = 0$ or

- $E[|a(n)|^4] \cdot (|t_i|^2 - 1) + 2 \cdot (E[|a(n)|^2])^2 \cdot \sum_{j \neq i} |t_j|^2 = 0.$

Naturally, some of the vector components of the combined channel and equaliser impulse response can be zero. These components of $\mathbf{t}$ which are not zero can be found from the solution of the following set of equations:

$$\begin{pmatrix} \mu_4 & 2{\mu_2}^2 & 2{\mu_2}^2 & \cdots & 2{\mu_2}^2 \\ 2{\mu_2}^2 & \mu_4 & 2{\mu_2}^2 & \cdots & 2{\mu_2}^2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 2{\mu_2}^2 & 2{\mu_2}^2 & \cdots & \mu_4 & 2{\mu_2}^2 \\ 2{\mu_2}^2 & 2{\mu_2}^2 & \cdots & 2{\mu_2}^2 & \mu_4 \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \\ \vdots \\ p_{M-2} \\ p_{M-1} \end{pmatrix} = \begin{pmatrix} \mu_4 \\ \mu_4 \\ \vdots \\ \mu_4 \\ \mu_4 \end{pmatrix}. \tag{E.7}$$

where p_i are the squares of the magnitudes of those specific t_i values, which are not zero. Furthermore, we have $\mu_4 = E[|a(n)|^4]$, $\mu_2 = E[|a(n)|^2]$ and M is the number of non–zero components that the vector $\mathbf{t}$ has. The solution of this linear system was found to be:

$$p_i = \frac{\mu_4}{\mu_4 + 2(M-1){\mu_2}^2} \quad \text{for all } p_i. \tag{E.8}$$

For $M = 1$ all t_j components of the vector $\mathbf{t}$ would be zero, except for one, for which we have $|t_i| = 1$. This yields the desired impulse response in the form of $(0, \cdots, 0, e^{j\phi}, 0, \cdots, 0)^T$. We will now show that only this solution corresponds to a local minimum, while all the other points associated with $(M > 1)$ constitute saddle points. In order to achieve this we will consider the second derivative of the cost–function of Equation (1.74) with respect to the combined channel and equaliser impulse response, rather than to the equaliser tap vector. This is carried out under the assumption that the equaliser tap vector's dimension is doubly infinite, in which case the channel matrix is square–shaped and invertible. From the second derivative of the cost–function we determine the Hessian matrix of the cost–function in order to define the kind of points we have, i.e. whether they are local minima, local maxima, or saddle points. First we repeat the cost–function of the algorithm as:

$$J(n) = \frac{1}{4} \cdot E[(|z(n)|^2 - R_2)^2]. \tag{E.9}$$

The gradient with respect to the transfer function is given by:

$$\begin{aligned} \nabla_{\mathbf{t}} J(n) &= \frac{1}{4} \cdot \nabla_{\mathbf{t}} (|z(n)|^2 - R_2)^2 \\ &= \frac{1}{2} \cdot (|z(n)|^2 - R_2) \nabla_{\mathbf{t}} (|z(n)|^2 - R_2) \\ &= \frac{1}{2} \cdot (|z(n)|^2 - R_2) z(n) \mathbf{a}^*(n). \end{aligned} \tag{E.10}$$

The second derivative calculation is formulated as follows:

$$\begin{aligned}\nabla_{\mathbf{tH}}(\nabla_{\mathbf{t}}J(n)) &= \nabla_{\mathbf{tH}}[\frac{1}{2}\cdot(|z(n)|^2-R_2)z(n)\mathbf{a}^*(n)] \\ &= \frac{1}{2}\cdot\left(z^*(n)\cdot z^2(n)-R_2z(n)\right)\mathbf{a}^*(n) \\ &= \frac{1}{2}\cdot\left(2|z(n)|^2-R_2\right)\mathbf{a}^*(n)\cdot\mathbf{a}^T(n). \end{aligned} \tag{E.11}$$

Equation (E.11) defines the Hessian matrix of the cost–function. Its entries h_{ij} associated with the vector points of interest can be found by evaluating Equation (E.11). Following a number of further manipulations we obtain:

$$\left.\begin{aligned} h_{ii} &= \tfrac{1}{4}\cdot\tfrac{\mu_4}{\mu_4+2{\mu_2}^2(M-1)} \\ h_{ij} &= {\mu_2}^2 t_i t_j^* \end{aligned}\right\} \quad \text{for } M\neq 1 \tag{E.12}$$

$$\left.\begin{aligned} h_{ii} &= \tfrac{\mu_4}{2} \\ h_{ij} &= 0 \end{aligned}\right\} \quad \text{for } M=1. \tag{E.13}$$

A particular way of determining the positive (negative) (in–)definiteness of the Hessian matrix of the cost–function in Equation (E.11) is by means of its subdeterminants. For $M=1$ we have $\Delta_i = (\frac{\mu_4}{2})^i > 0$, which implies that all the subdeterminants are positive, i.e. the Hessian matrix is positive definite and the associated point is a local minimum. For $M\neq 1$ the first subdeterminant is $\Delta_1 = \frac{1}{4}\cdot\frac{\mu_4}{\mu_4+2{\mu_2}^2(M-1)} > 0$. The second subdeterminant is given by:

$$\begin{aligned}\Delta_2 &= \left(\frac{1}{4}\cdot\frac{{\mu_4}^2}{\mu_4+2{\mu_2}^2(M-1)}\right)^2-{\mu_2}^4\left(\frac{\mu_4}{\mu_4+2{\mu_2}^2(M-1)}\right)^2 \Rightarrow \\ \Delta_2 &= \left(\left(\frac{\mu_4}{4}\right)^2-{\mu_2}^4\right)\left(\frac{{\mu_4}^2}{\mu_4+2{\mu_2}^2(M-1)}\right)^2. \end{aligned} \tag{E.14}$$

For the symmetric QAM constellations of interest the ratio $\frac{{\mu_4}^2}{{\mu_2}^4}$ takes values less than 2, which results in $\Delta_2 < 0$. Since an even–order subdeterminant is negative, the Hessian matrix is indefinite, i.e. the associated points are saddle points.

In this section we have shown that only the desirable combined channel and equaliser impulse response constitutes a stable point for the CMA in the case of doubly infinite equaliser tap vector dimensions, compared to the maximum channel delay spread. In the realistic case of a high, but finite number of equaliser taps the presence of undesirable equilibria depends on the CIR. As an example, Ding *et al.* [22] proved the existence of such equilibria for the special case of an autoregressive channel and found these stable points [22]. A range of further studies of this issue have been carried out recently and intensive research interests have been attracted by it in an effort to facilitate the blind equalisation

of channels exhibiting severe distortions.

E.1 The CMA Convergence Points

Appendix F

Fourth–Order Statistics and Polycepstra Definitions and Properties

Based on [5,15,31,76], in this appendix we give some fourth–order statistics definitions and properties, which are useful for the blind equalisation algorithms of Section 1.3.11. The definitions are based on [5], in which the polycepstra–based algorithms first appeared and on [31], where the super–exponential algorithm was proposed, using fourth–order cumulants. In general, the joint cumulant of the random variables x_{n_1}, x_{n_2},
$\cdots x_{n_m}$ is given by [31]:

$$L(x_{n_1}; x_{n_2}; \cdots; x_{n_m}) = (-j)^m \frac{\partial^m ln\phi(\omega)}{\partial\omega_{n_1} \ldots \partial\omega_{n_m}|_{\omega=0}}, \tag{F.1}$$

where $\phi(\omega)$ is the so-called joint characteristic function of the random variables x_i, defined as [147]:

$$\phi(\omega) = E\left[e^{j\omega \mathbf{x}}\right] = E\left[e^{j\sum_{i=1}^{m}\omega_i x_i}\right]. \tag{F.2}$$

For zero–mean random variables and cumulants up to fourth–order, the definition is equivalent to [5]:

$$L(x_1) = 0 \tag{F.3}$$

$$L(x_1; x_2) = E[x_1 \cdot x_2] \tag{F.4}$$

$$L(x_1; x_2; x_3) = E[x_1 \cdot x_2 \cdot x_3] \tag{F.5}$$

$$\begin{aligned} L(x_1; x_2; x_3; x_4) = {} & E[x_1 \cdot x_2 \cdot x_3 \cdot x_4] - \\ & E[x_1 \cdot x_2] \cdot E[x_3 \cdot x_4] - \\ & E[x_1 \cdot x_3] \cdot E[x_2 \cdot x_4] - \end{aligned}$$

$$E[x_1 \cdot x_4] \cdot E[x_2 \cdot x_3] \tag{F.6}$$

When the random variables are actually samples from a zero–mean stochastic process, such as the received signal $y(k)$, then the fourth–order cumulant is given by:

$$\begin{aligned} L_y(m,n,l) &= E[y(k)y(k+m)y(k+n)y(k+l)] - \\ &\quad E[y(k)y(k+m)] \cdot E[y(k)y(k+l-n)] - \\ &\quad E[y(k)y(k+n)] \cdot E[y(k)y(k+l-m)] - \\ &\quad E[y(k)y(k+l)] \cdot E[y(k)y(k+m-n)]. \end{aligned} \tag{F.7}$$

Following the above definition, we will now summarise some of the associated cumulant properties. First of all, the cumulant of the sum of two independent and zero–mean stationary signals $x_1(k)$ and $x_2(k)$ is equal to the sum of the cumulants of $x_1(k)$ and $x_2(k)$, i.e.:

$$L_{x_1+x_2}(m,n,l) = L_{x_1}(m,n,l) + L_{x_2}(m,n,l). \tag{F.8}$$

Furthermore, the cumulant of the convolution of two zero–mean stationary signals corresponds to the convolution of their cumulants in the cumulant domain:

$$L_{x_1 * x_2}(m,n,l) = L_{x_1}(m,n,l) * L_{x_2}(m,n,l), \tag{F.9}$$

where $*$ stands for convolution. Two further properties related to the signal distributions:

- If $x(k)$ is Gaussian, then the 4–th order cumulant of $x(k)$ is 0
- If $x(k)$ is i.i.d., then the 4–th order cumulant of $x(k)$ is given by:

$$L_x(m,n,l) = K(x(k)) \cdot \delta(m,n,l), \tag{F.10}$$

where $K(x(k))$ is the Kurtosis of $x(k)$ defined as:

$$K(x(k)) = E[|x(k)|^4] - 3\left(E[x^2(k)]\right)^2 \tag{F.11}$$

for real-valued signals Equation (1.80) defined the Kurtosis for complex-valued signals, which is repeated here:

$$K(z) = E[|z(n)|^4] - 2 \cdot E^2[|z(n)|^2] - |E[z^2(n)]|^2.$$

Finally, below we will now give a cumulant property of stochastic signals passing through linear systems. We assume that the i.i.d. signal $x(k)$ is input to a system having an impulse response h_k. The output of the system is given by:

$$y(k) = \sum_i h_i \cdot a(k-i). \tag{F.12}$$

The cumulant of the output is then given by:

$$L_y(n,k,l) = K(a(k)) \cdot L_h(m,n,l), \tag{F.13}$$

where $L_h(m,n,l)$ is the fourth–order moment of the CIR. Here, the moment has replaced the cumulant because $h(k)$ is a deterministic signal. We can now form the cumulant of the received signal $y(k)$, which is given by the convolution of the i.i.d. input signal $a(k)$ with the CIR h_k plus the additive Gaussian noise:

$$y(k) = h_k * x(k) + e(k) \Rightarrow \tag{F.14}$$

$$L_y(m,n,l) = K(x(k)) \cdot L_h(m,n,l), \tag{F.15}$$

where the Gaussian noise term vanishes, since its fourth–order cumulant is zero. We will now define the tricepstrum of a signal $x(k)$ as:

$$c_x(m,n,l) = Z_{(3)}^{-1}\{ln(Z_{(3)}\{L_x(m,n,l)\})\}, \tag{F.16}$$

where $Z_{(3)}$ stands for the 3D z–transform. Complicated as this definition may seem, it simplifies the cumulant–based relationship of Equation (F.15) to the tricepstrum domain as:

$$c_y(m,n,l) = c_h(m,n,l) \qquad (m,n,l) \neq (0,0,0), \tag{F.17}$$

which means that the tricepstrum of the received signal is the same as the tricepstrum of the CIR, provided that the input is i.i.d. and the noise is Gaussian distributed. By measuring the tricepstrum of the received signal, we can now have an estimate of the tricepstrum of the CIR. It can be shown that the following identity holds [5]:

$$\begin{aligned} &\sum_{I=1}^{p} A^{(I)}[L_y(m-I,n,l) - L_y(m+I,n+I,l+I)] + \\ &+ \sum_{J=1}^{q} B^{(J)}[L_y(m-J,n-J,l-J) - L_y(m+J,n,l)] = \\ &= -mL_y(m,n,l), \end{aligned}$$

(F.18)

where

$$K \cdot c_y(K,0,0) = \begin{cases} -A^{(K)}, & K = 1,\ldots,p \\ B^{(-K)}, & K = -1,\ldots,-q\ . \end{cases} \tag{F.19}$$

This relationship holds for any (m,n,l). Nonetheless, a rule of thumb, suggested in [5] is to define $w = max(p,q)$, $z \leq w/2$, $s \leq z$ and then select $m = -w,\ldots,-1,1,\ldots,w$, $n = -z,\ldots,0,\ldots,z$ and $l = -s,\ldots,0,\ldots,s$. It is shown in [5] that $A^{(I)}, B^{(J)}$ are the minimum– and maximum–phase differential cepstrum coefficients related to the zeros of the CIR. Thus, estimating $A^{(I)}, B^{(J)}$ from the system of Equations (F.18) means that we can invert the CIR apart from an uncertainty of the magnitude A. Theoretically, it is required that $p,q \to \infty$, but in practice we can use a sufficiently large discrete value for each of them. Equations (F.18) can then be rewritten in matrix form as:

$$\mathbf{P} \cdot \mathbf{a} = \mathbf{p}, \tag{F.20}$$

where $\mathbf{P}$ is the matrix containing the L_y–elements of the left–hand side of Equation (F.18), $\mathbf{a}$ is the vector containing $A^{(I)}$ and $B^{(J)}$ and $\mathbf{p}$ is the vector containing the L_y– elements of the right–hand side of Equation (F.18). We have to estimate $\mathbf{a}$, in order to invert the CIR. This can be carried out upon invoking the LMS solution of Equation (F.20), as in [5], yielding:

$$\hat{\mathbf{a}}^{(i+1)} = \hat{\mathbf{a}}^{(i)} + \mu^{(i)} \cdot (\hat{\mathbf{P}}^{(i)})^H \cdot (\hat{\mathbf{p}}^{(i)} - \hat{\mathbf{P}}^{(i)} \cdot \hat{\mathbf{a}}^{(i)}) \tag{F.21}$$

with

$$0 < \mu^{(i)} < 2/tr\{(\hat{\mathbf{P}}^{(i)})^H \cdot \hat{\mathbf{P}}^{(i)}\},$$

where $tr()$ stands for the trace of a matrix. When $\mu^{(i)}$ is in this region, stability is ensured, as shown in [15].

The exact relationship of the coefficients $A^{(I)}$ and $B^{(I)}$ with the poles and zeros of the equaliser filters' polynomials is as follows. If the z–domain channel transfer function polynomial $H(z)$ is given by $H(z) = I(z^{-1}) \cdot O(z)$, as in Equation (1.159), where

$$I(z^{-1}) = \frac{\Pi_{i=1}^{L_1}(1 - a_i z^{-1})}{\Pi_{i=1}^{L_3}(1 - c_i z^{-1})} \tag{F.22}$$

with $|a_i| < 1$ and $|c_i| < 1$ and

$$O(z) = \Pi_{i=1}^{L_2}(1 - b_i z) \tag{F.23}$$

with $b_i < 1$, then the coefficients $A^{(I)}$ and $B^{(I)}$ are given by:

$$A^{(I)} = \sum_{i=1}^{L_1} a_i^I - \sum_{i=1}^{L_3} c_i^I \tag{F.24}$$

$$B^{(I)} = \sum_{i=1}^{L_2} b_i^I \tag{F.25}$$

Having estimated $A^{(I)}$ and $B^{(I)}$ we can now use Equation (F.25) to estimate the time–domain coefficients $\hat{i}_{inv}^{(i)}(k)$ and $\hat{o}_{inv}^{(i)}(k)$, which are the coefficients of $1/I(z^{-1})$ and $1/O(z)$, respectively, as in [5] :

$$\hat{i}_{inv}^{(i)}(k) = -\frac{1}{k}\sum_{n=2}^{k+1}[-A^{(n-1),(i)}] \cdot \hat{i}_{inv}^{(i)}(k-n+1), \quad k = 1, \ldots, N_1 \tag{F.26}$$

$$\hat{o}_{inv}^{(i)}(k) = \frac{1}{k}\sum_{n=k+1}^{0}[-B^{(1-n),(i)}] \cdot \hat{o}_{inv}^{(i)}(k-n+1), \quad k = -1, \ldots, -N_2, \tag{F.27}$$

where the superscript i denotes the i-th iteration of the algorithm. The equaliser's impulse response is finally given by the convolution $\hat{i}_{inv}^{(i)}(k) * \hat{o}_{inv}^{(i)}(k)$. In the case of a DFE we also have to estimate the inverse z–transforms of $I(z^{-1})$ and $O^*(z^{-1})$, which is given by [5]:

$$\hat{i}_o^{(i)}(k) = \frac{1}{k}\sum_{n=2}^{k+1}[A^{(n-1),(i)} + (B^{(n-1),(i)})^*]\hat{i}_o^{(i)}(k-n+1) \qquad k = 1, \ldots, N_3 \tag{F.28}$$

$$\hat{o}_o^{(i)}(k) = -\frac{1}{k}\sum_{n=2}^{k+1}[B^{(n-1),(i)}]^* \cdot \hat{o}_o^{(i)}(k-n+1), \quad k = 1, \ldots, N_4. \tag{F.29}$$

Finally, the feedforward and feedback equaliser impulse responses are now extracted as $\hat{o}_o^{(i)}(k) * \hat{o}_{inv}^{(i)}(k)$ and $\hat{i}_{inv}^{(i)}(k) * \hat{o}_o^{(i)}(k) - \delta(k)$, respectively. The channel attenutation parameter A can be estimated using Automatic Gain Control (AGC). A synopsis of the equations involved in the TEA is given in Table F.

$\sum_{I=1}^{p} A^{(I)}[L_y(m-I,n,l) - L_y(m+I,n+I,l+I)] +$ $+\sum_{J=1}^{q} B^{(J)}[L_y(m-J,n-J,l-J) - L_y(m+J,n,l)] =$ $= -mL_y(m,n,l)$
$\hat{\mathbf{a}}^{(i)} + \mu^{(i)} \cdot (\hat{\mathbf{P}}^{(i)})^H \cdot (\hat{\mathbf{p}}^{(i)} - \hat{\mathbf{P}}^{(i)} \cdot \hat{\mathbf{a}}^{(i)}))$ $\mathbf{a} = (A^{(1)}, \ldots, A^{(p)}, B^{(1)}, \ldots, B^{(q)})^T$ $[\mathbf{p}]_i = -m \cdot L_y(m,n,l)$ $[\mathbf{P}]_{ij} = L_y(m-I,n,l) - L_y(m+I,n+I,l+I)$ $i = 1, \ldots, p$ $[\mathbf{P}]_{ij} = L_y(m-J,n-J,l-J) - L_y(m+J,n,l)$ $i = p+1, \ldots, p+q$ $w = max(p,q),\ z \leq w/2,\ s \leq z$ $m = -w, \ldots, -1, 1, \ldots, w$ $n = -z, \ldots, 0, \ldots, z$ $l = -s, \ldots, 0, \ldots, s$
$\hat{i}_{inv}^{(i)}(k) = -\frac{1}{k}\sum_{n=2}^{k+1}[-A^{(n-1),(i)}] \cdot \hat{i}_{inv}^{(i)}(k-n+1)$ $k = 1, \ldots, N_1$
$\hat{o}_{inv}^{(i)}(k) = \frac{1}{k}\sum_{n=k+1}^{0}[-B^{(1-n),(i)}] \cdot \hat{o}_{inv}^{(i)}(k-n+1)$ $k = -1, \ldots, -N_2$
$\hat{i}_{o}^{(i)}(k) = \frac{1}{k}\sum_{n=2}^{k+1}[A^{(n-1),(i)} + (B^{(n-1),(i)})^*]\hat{i}_{o}^{(i)}(k-n+1)$ $k = 1, \ldots, N_3$
$\hat{o}_{o}^{(i)}(k) = -\frac{1}{k}\sum_{n=2}^{k+1}[B^{(n-1),(i)}]^* \cdot \hat{o}_{o}^{(i)}(k-n+1)$ $k = 1, \ldots, N_4$

Table F.1: A synopsis of the equations related to the TEA

Appendix G

Convergence in the Presence of Channel Noise: Extraction of the Local Minima

In this appendix the derivation of the relationship describing the convergence points of the CMA in the presence of channel noise is given. The derivation is based on Equation (1.117), which is repeated here for convenience:

$$\mathbf{y}^*(n) \cdot z(n) \cdot (|z(n)|^2 - R_2) = \mathbf{0}. \tag{1.117}$$

In the presence of channel noise, the signals associated with this equation become:

$$\mathbf{y}(n) = \mathbf{H} \cdot \mathbf{a}(n) + \mathbf{e}(n) \tag{G.1}$$

$$z(n) = \mathbf{t}^T \cdot \mathbf{a}(n) + \mathbf{c}^T \cdot \mathbf{y}(n) \tag{G.2}$$

where the definitions of the variables were given in Chapter 1. Substituting the above relationships into Equation (1.117) gives:

$$(\mathbf{H} \cdot \mathbf{a}(n) + \mathbf{e}(n)) \cdot \left(\mathbf{t}^T \cdot \mathbf{a}(n) + \mathbf{c}^T \cdot \mathbf{e}(n)\right) \cdot \left(\left|\mathbf{t}^T \cdot \mathbf{a}(n) + \mathbf{c}^T \cdot \mathbf{e}(n)\right|^2 - R_2\right) = \mathbf{0}. \tag{G.3}$$

Since we are interested in the statistical behaviour of the equaliser coefficient update algorithm, we will only consider the mean values of the random variables of Equation (G.3). From this equation we observe that the terms which having a non–zero mean value are the terms that have an even number of $\mathbf{a}(n)$ and $\mathbf{a}^*(n)$ vector multipliers and hence are associated with an even power of $a(n)$ or $a^*(n)$.

The terms for which this number is odd will always give a zero mean value. This is a consequence of the symmetry of the QAM constellation and the i.i.d. assumption of the transmitted symbols, which, again, implies that only the terms of the form $|a(n)|^k$, k even and $k \geq 2$ will give a positive non-zero value. By contrast, it is impossible for products consisting of an odd number of $\mathbf{a}(n)$ and $\mathbf{a}^*(n)$ terms, to produce terms which are equal to a power of $|a(n)|$, because $|a(n)|^k = a(n)^{k/2} \cdot a(n)^{*k/2}$, which implies that even for our specific case of $k = 4$, only terms of the form $a(n)^2$ and $a(n)^{*2}$ may produce a power of $|a(n)|$, when multiplied. However, these products will contribute a term of the form $a(n)^k \cdot (a(n)^*)^m$, where k, m are odd and hence $k+m$ is even. Again, the associated terms always have a zero mean value. Therefore, Equation (G.3) reduces to:

$$\begin{aligned} \mathbf{H} \cdot \mathbf{a}^*(n) \cdot \mathbf{a}^T(n) \left(\mathbf{t} \cdot \mathbf{t}^H \cdot \mathbf{a}^* \cdot \mathbf{a}^T + \mathbf{c}^H \cdot \mathbf{e}^* \cdot \mathbf{e}^T \mathbf{c} - R_2 \right) \mathbf{t} + \\ \mathbf{e}^* \cdot \mathbf{e}^T \left(\mathbf{t}^T \cdot \mathbf{a} \cdot \mathbf{a}^H \cdot \mathbf{t}^* + \mathbf{t}^H \cdot \mathbf{a}^* \cdot \mathbf{a}^T \cdot \mathbf{t} + \mathbf{c}^H \cdot \mathbf{e}^* \cdot \mathbf{e}^T \cdot \mathbf{c} - R_2 \right) \mathbf{c} \quad = \quad \mathbf{0}. \end{aligned} \tag{G.4}$$

Upon finding the mean values of the vectors hosting stochastic variables in Equation (G.4) we arrive at Equation (1.130). As mentioned before, this process is based on the assumption that the QAM constellation concerned is symmetric and that the transmitted symbols are i.i.d..

Appendix H

Glossary

- TDMA: Time Division Multiple Acess
- CMA: Constant Modulus Algorithm
- NCMA: Normalised Constant Modulus Algorithm
- MCMA: Modified Constant Modulus Algorithm
- SCS: Soft Constraint Satisfaction
- LMS: Least Mean Squares
- RLS: Recursive Least Squares
- DFE: Decision–Feedback Equaliser
- DD: Decision–Directed
- ZF: Zero–Forcing
- PSP: Per–Survivor Processing
- ML: Maximum Likelihood
- MLSE: Maximum–Likelihood Sequence Estimation
- TEA: Tricepstrum Equalisation Algorithm
- NN: Neural Network
- PLL: Phase Locked Loop
- BCH: Bose–Chauduri–Hocquenghem

- RS: Reed Solomon
- RSC: Recursive Systematic Convolutional
- CM: Coded Modulation
- TCM: Trellis–Coded Modulation
- TTCM: Turbo trellis–Coded Modulation
- BICM: Bit–Interleaved Coded Modulation
- RCPCC: Rate Compatible Punctured Convolutional Coding
- MAP: Maximum A–Posteriori
- APP: A–Posteriori Probability
- VA: Viterbi Algorithm
- SOVA: Soft Output Viterbi Algorithm
- LLR: LogLikelihood Ratio
- SP: Set Partitioning
- ETSI: European Telecommunications Standards Institute
- DVB: Digital Video Broadcasting
- GSM: Groupe Special Mobile
- WATM: Wireless Asynchronous Transfer Mode
- SWATM: Shortened (channel) Wireless Asynchronous Transfer Mode
- MPEG: Motion Pictures Experts Group
- CIR: Channel's Impulse Response
- AWGN: Additive White Gaussian Noise
- IID: Independent Identically Distributed
- ISI: InterSymbol Interference
- MSE: Mean Squared Error
- BER: Bit–Error Rate

- SER: Symbol–Error Rate
- SNR: Signal–to–Noise Ratio
- PSNR: Peak Signal–to–Noise Ratio
- PDF:Probability Density Function
- QAM: Quadrature Amplitude Modulation
- PAM: Pulse–Amplitude Modulation
- PSK: Phase Shift Keying
- BPSK: Binary Phase Shift Keying
- QPSK: Quadrature Phase Shift Keying
- GMSK: Gaussian Minimum Shift Keying
- OFDM: Orthogonal Frequency Division Multiplexing
- BPS: Bits Per Symbol
- IBPS: Information Bits Per Symbol
- SVD: Singular Value Decomposition
- CAZE:Column–Anchored ZeroForcing Equalisation

Bibliography

[1] S. Vlahoyiannatos and L. Hanzo, "Soft decision–feedback equalisation using the constant modulus algorithm," *submitted to the European Transactions on Telecommunications.*

[2] S. Vlahoyiannatos and L. Hanzo, "Constant modulus algorithm assisted soft decision feedback equalisation," in *to appear in the Proceedings of the ICC*, (Helsinki, Finland), 11–15 June 2001.

[3] Y. Sato, "A method of self-recovering equalization for multilevel amplitude-modulation systems," *IEEE Transactions on Communications*, vol. COM-23, pp. 679–682, June 1975.

[4] D. N. Godard, "Self-recovering equalization and carrier tracking in two-dimensional data communication systems," *IEEE Transactions on Communications*, vol. COM-28, pp. 1867–1875, November 1980.

[5] D. Hatzinakos and C. L. Nikias, "Blind equalization using a tricepstrum based algorithm," *IEEE Transactions on Communications*, vol. COM-39, pp. 669–682, May 1991.

[6] N. Seshadri, "Joint data and channel estimation using blind trellis search techniques," *IEEE Transactions on Communications*, vol. COM-42, pp. 1000–1011, February-April 1994.

[7] L. Tong, G. Xu, and T. Kailath, "Blind identification and equalization based on second-order statistics: a time domain approach," *IEEE Transactions on Information Theory*, vol. COM-40, pp. 380–389, December 1994.

[8] S. Vlahoyiannatos and L. Hanzo, "Blind equalisers: A comparative study," *submitted to the Proceeding of the IEEE.*

[9] L. Hanzo, T. Webb, and T. Keller, *Single- and Multi-carrier Quadrature Amplitude Modulation.* John Wiley-IEEE Press, April 2000.

[10] C. S. Lee, S. Vlahoyiannatos, and L. Hanzo, "Satellite based turbo-coded, blind-equalised 4-QAM and 16-QAM digital video broadcasting," *IEEE Transactions on Broadcasting*, vol. **, March 2000.

[11] C. S. Lee, S. Vlahoyiannatos, and L. Hanzo, "Blind–equalised and turbo–coded satellite based digital video broadcasting," in *Proceedings of the ICT*, (Acapulco, Mexico), pp. 562–566, 22–25 May 2000.

[12] S. Vlahoyiannatos and L. Hanzo, "Blind PSP–based turbo equalisation," in *Proceedings of the VTC*, (Rhodes, Greece), 6–9 May 2001.

[13] S. Vlahoyiannatos, S. X. Ng, and L. Hanzo, "Blind PSP–based turbo equalisation using coded modulation," in *to appear in the Proceedings of the European Conference on Communications*, (Bratislava), 5–7 July 2001.

[14] J. G. Proakis, *Digital Communications.* McGraw-Hill International Editions, 3. ed., 1995.

[15] S. Haykin, *Adaptive Filter Theory.* Prentice Hall International Editions, 3. ed., 1996.

[16] D. Boss, K. D. Kammeyer, and T. Petermann, "Is blind channel estimation feasible in mobile communication systems?; A study based on GSM," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 1479–1492, October 1998.

[17] J. J. Bussgang, "Cross–correlation functions of amplitude-distorted Gaussian signals," *MIT Research Laboratory Technical Report*, no. 216, 1952.

[18] ETSI, *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for 11/12 GHz Satellite Services*, August 1997. EN 300 421 V1.1.2.

[19] ETSI, *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television*, August 1997. EN 300 744 V1.1.2.

[20] ETSI, *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for cable systems*, December 1997. EN 300 429 V1.2.1.

[21] G. Picchi and G. Prati, "Blind equalization and carrier recovery using a "Stop-and-Go" decision-directed algorithm," *IEEE Transactions on Communications*, vol. COM-35, pp. 877–887, September 1987.

[22] Z. Ding, R. A. Kennedy, B. D. O. Anderson, and C. R. Johnson, "Ill-convergence of Godard blind equalizers in data communications systems," *IEEE Transactions on Communications*, vol. COM-39, pp. 1313–1327, September 1991.

[23] R. Raheli, A. Polydoros, and C. K. Tzou, "Per-survivor processing: a general approach to MLSE in uncertain environments," *IEEE Transactions on Communications*, vol. COM-43, pp. 354–364, February–April 1995.

[24] G. Kechriotis, E. Zervas, and E. S. Manolakos, "Using recurrent neural networks for adaptive communication channel equalization," *IEEE Transactions on Neural Networks*, vol. 5, pp. 267–278, March 1994.

[25] D. Boss and K. D. Kammeyer, "Blind GSM channel estimation," in *Proceedings of the 1997 47th IEEE Vehicular Technology Conference*, (Phoenix, USA), pp. 1044–1048, 4–7 May 1997.

[26] A. Benveniste, M. Goursat, and G. Ruget, "Robust identification of a nonminimum phase system: Blind adjustment of a linear equalizer in data communications," *IEEE Transactions on Automatic Control*, vol. AC-25, pp. 385–399, June 1980.

[27] G. J. Foschini, "Equalizing without altering or deleting data," *AT&T Technical Journal*, vol. 64, pp. 1885–1911, October 1985.

[28] L. Tong, G. Xu, and T. Kailath, "A new approach to blind identification and equalization of multipath channels," in *Proceedings of the 25th Asilomar Conference*, (Pacific Grive, Canada), pp. 856–860, 4–6 November 1991.

[29] E. Mulines, J. Cardoso, and S. Mayrargue, "Subspace methods for the blind identification of multichannel FIR filters," *IEEE Transactions on Signal Processing*, vol. 43, pp. 516–525, February 1995.

[30] S. Chen and Y. Wu, "Maximum likelihood joint channel and data estimation using genetic algorithms," *IEEE Transactions on Signal Processing*, vol. 46, pp. 1469–1473, May 1998.

[31] O. Shalvi and E. Weinstein, "Super-exponential methods for blind deconvolution," *IEEE Transactions on Information Theory*, vol. 39, pp. 504–519, March 1993.

[32] M. Goursat and A. Benveniste, "Blind equalizers," *IEEE Transactions on Communications*, vol. COM-28, pp. 871–883, August 1984.

[33] S. Bellini, "Bussgang techniques for blind equalisation," in *Proceedings of the IEEE Global Telecommunications Conference*, (Houston, USA), pp. 1634–1640, December 1986.

[34] D. G. Forney, "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," *IEEE Transactions on Information Theory*, vol. IT–18, pp. 363–378, May 1972.

[35] C. K. Tzou, R. Raheli, and A. Polydoros, "Applications of per-survivor processing to mobile digital communications," in *Proceedings of the IEEE Global Telecommunications Conference. Part 4 (of 4)*, (Houston, USA), pp. 77–81, Nov 29– Dec 2 1993.

[36] A. Polydoros and K. Chugg, "MLSE for an unknown channel - Part I: Optimality considerations," *IEEE Transactions on Communications*, vol. COM-44, pp. 836–846, July 1996.

[37] K. Chugg and A. Polydoros, "MLSE for an unknown channel - Part II: Tracking performance," *IEEE Transactions on Communications*, vol. COM-44, pp. 949–958, August 1996.

[38] C. Antón-Haro, A. R. Fonolossa, and J. R. Fonolossa, "Blind channel estimation and data detection using hidden Markov models," *IEEE Transactions on Signal Processing*, vol. 45, pp. 241–247, January 1997.

[39] H. A. Cirpan and M. K. Tsatsanis, "Blind receivers for nonlinearly modulated signals in multipath," *IEEE Transactions on Signal Processing*, vol. 47, pp. 583–586, February 1999.

[40] L. Favalli, A. Mecocci, and P. Savazzi, "Blind MLSE equalizer with fuzzy metric calculation for mobile radio environments," *Electronics Letters*, vol. 33, pp. 1841–1842, October 1997.

[41] K. Chugg, "Acquisition performance of blind sequence detectors using per-survivor processing," in *Proceedings of the 1997 47th IEEE Vehicular Technology Conference*, pp. 539–543, May 1997.

[42] K. Chugg, "Blind acquisition characteristics of PSP-based sequence detectors," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 1518–1529, October 1998.

[43] E. Baccarelli and R. Cusani, "Combined channel estimation and data detection using soft statistics for frequency selective fast-fading digital links," *IEEE Transactions on Communications*, vol. COM-46, pp. 424–427, April 1998.

[44] R. Raheli, G. Marino, and P. Castoldi, "Per-survivor processing and tentative decisions: what is in between?," *IEEE Transactions on Communications*, vol. 44, pp. 127–129, February 1996.

[45] G. Vitetta, M. Taylor, and P. Desmond, "Maximum likelihood decoding of uncoded and coded psk signal sequences transmitted over Rayleigh flat-fading channels," *IEEE Transactions on Vehicular Technology*, vol. 43, pp. 2750–2758, November 1995.

[46] M. Rollins and S. Simmons, "Simplified per-survivor kalman processing in fast frequency-selective fading channels," *IEEE Transactions on Communications*, vol. 45, pp. 543–553, May 1997.

[47] A. Anastasopoulos and A. Polydoros, "Adaptive soft-decision algorithms for mobile fading channels," *European Transactions on Telecommunications and Related Technologies*, vol. 9, pp. 183–190, March–April 1998.

[48] M. K. Tsatsanis and G. B. Giannakis, "Transmitter induced cyclostationarity for blind channel equalization," *IEEE Transactions on Signal Processing*, vol. 45, pp. 1785–1794, July 1997.

[49] A. Chevreuil, F. Desbouvries, A. Gorokhov, P. Loubaton, and C. Vignat, "Blind equalization in the presence of jammers and unknown noise: Solutions based on second-order cyclostationary statistics," *IEEE Transactions on Signal Processing*, vol. 46, pp. 259–263, January 1998.

[50] A. Chevreuil and P. Loubaton, "Blind second–order identification of FIR channels: Forced cyclostationarity and structured subspace method," *IEEE Signal Processing Letters*, vol. 4, pp. 204–206, July 1997.

[51] M. K. Tsatsanis and G. B. Giannakis, "Subspace methods for blind estimation of time-varying FIR channels," *IEEE Transactions on Signal Processing*, vol. 45, pp. 3084–3093, December 1997.

[52] Z. Ding, "Matrix outer-product decomposition method for blind multiple channel identification," *IEEE Transactions on Signal Processing*, vol. 45, pp. 3053–3061, December 1997.

[53] G. B. Giannakis and E. Serpedin, "Blind identification of ARMA channels with periodically modulated inputs," *IEEE Transactions on Signal Processing*, vol. 46, pp. 3099–3104, November 1998.

[54] G. B. Giannakis, "Filterbanks for blind channel identification and equalization," *IEEE Signal Processing Letters*, vol. 4, pp. 184–187, June 1997.

[55] R. W. J. Heath and G. B. Giannakis, "Exploiting input cyclostationarity for blind channel identification in OFDM systems," *IEEE Transactions on Signal Processing*, vol. 47, pp. 848–856, March 1999.

[56] H. E. Wong and J. A. Chambers, "Two-stage interference immune blind equaliser which exploits cyclostationary statistics," *Electronics Letters*, vol. 32, pp. 1763–1764, September 1996.

[57] H. Liu, G. Xu, L. Tong, and T. Kailath, "Recent developments in blind channel equalization: From cyclostationarity to subspace," *Signal Processing*, vol. 50, pp. 83–99, April 1996.

[58] Y. Hua, H. Yang, and W. Qiu, "Source correlation compensation for blind channel identification based on second order statistics," *IEEE Signal Processing Letters*, vol. 1, pp. 119–120, August 1994.

[59] Z. Ding, "Characteristics of band-limited channels unidentifiable from second-order cyclostationary statistics," *IEEE Signal Processing Letters*, vol. 3, pp. 150–152, May 1996.

[60] J. M. F. Xavier, V. A. N. Barroso, and J. M. F. Moura, "Closed-form blind channel identification and sourse separation in SDMA systems through correlative coding," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 1506–1517, October 1998.

[61] X. Wang and H. V. Poor, "Blind equalization and multiuser detection in dispersive CDMA channels," *IEEE Transactions on Communications*, vol. 46, pp. 91–103, January 1998.

[62] X. Wang and H. V. Poor, "Blind joint equalization and multiuser detection for DS-CDMA in unknown correlated noise," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, pp. 886–895, July 1999.

[63] J. Zhu, Z. Ding, and X. R. Cao, "Column–anchored zeroforcing blind equalization for multiuser wireless FIR channels," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 411–423, March 1999.

[64] H. H. Zeng and L. Tong, "Blind channel–estimation using the second–order statistics algorithms," *IEEE Transactions on Signal Processing*, vol. 45, pp. 1919–1930, August 1997.

[65] T. J. Endres, S. D. Halford, C. R. Johnson, and G. B. Giannakis, "Blind adaptive channel equalization using fractionally-spaced receivers: A comparison study," in *Proceedings of the Conference on Information Sciences and Systems*, (Princeton, USA), 20–22 March 1996.

[66] C. R. Johnson, P. Schniter, T. J. Endres, J. D. Behm, D. R. Brown, and R. A. Casas, "Blind equalization using the constant modulus criterion: A review," *IEEE Proceedings of the IEEE*, vol. 86, pp. 1927–1950, October 1998.

[67] L. Tong and S. Perreau, "Analysis of a nonparametric blind equalizer for discrete-valued signals," *Proceedings of the IEEE*, vol. 86, pp. 1951–1968, Mar 1996.

[68] A. K. Nandi, *Blind Estimation using Higher-Order Statistics*. Glasgow: Kluwer Academic Publishers, 1999.

[69] C. Becchetti, A. Cocco, and G. Jacovitti, "Performance comparison of second order based blind equalizers in data communication channels," in *Proceedings of the 1997 13th International Conference on Digital Signal Processing, DSP, Part 1 (of 2)*, vol. 1, (Santorini, Greece), pp. 147–150, 2–4 July 1997.

[70] M. Kristensson and B. Ottersten, "Asymptotic comparison of two blind channel identification algorithms," in *Proceedings of the 1997 1st IEEE Signal Processing Workshop on Signal Processing Advances in Wireless Communications, SPAWC'97*, pp. 361–364, 16–18 April 1997.

[71] J. Altuna and B. Mulgrew, "Comparison of cyclostationary blind equalization algorithms in the mobile radio environment," *International Journal of Adaptive Control and Signal Processing*, vol. 12, pp. 267–282, May 1998.

[72] K. Skowratananont and J. A. Chambers, "Comparison of blind channel estimation and equalisation techniques for a fading environment," in *Proceedings of the 1998 6th IEE Conference on Telecommunications*, no. 451, (Edinburgh, UK), pp. 27–31, 21 March–2 April 1998.

[73] J. J. Shynk, P. R. Gooch, G. Krishnamurthy, and C. K. Chan, "Comparative performance study of several blind equalization algorithms," in *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 1565, (San Diego, USA), pp. 102–117, 22–24 July 1991.

[74] T. Schirtzinger, X. Li, and W. K. Jenkins, "Comparison of three algorithms for blind equalization based on the constant modulus error criterion," in *Proceedings of the 1995 International Conference on Acoustics, Speech, and Signal Processing, Part 2 (of 5)*, (Detroit, USA), pp. 1049–1052, 9–12 May 1995.

[75] T. J. Endres, S. D. Halford, C. R. Johnson, and G. B. Giannakis, "Simulated comparisons of blind equalization algorithms for cold start-up applications," *International Journal of Adaptive Control and Signal Processing*, vol. 12, pp. 283–301, May 1998.

[76] S. Haykin, *Blind Deconvolution.* Prentice Hall, 1 ed., 1994.

[77] D. G. Lainiotis, S. K. Katsikas, and S. D. Likothanassis, "Optimal seismic deconvolution," *Signal Processing*, vol. 15, pp. 375–404, December 1988.

[78] D. Huang and F. Gustafsson, "Sufficient output conditions for identifiability in blind equalization," *IEEE Transactions on Communications*, vol. 47, pp. 191–194, February 1999.

[79] S.-C. Pei and M.-F. Shih, "Fractionally spaced blind equalization using polyperiodic linear filtering," *IEEE Transactions on Communications*, vol. 46, pp. 16–19, January 1998.

[80] K. Dogancay and R. A. Kennedy, "Least squares approach to blind channel equalization," *Signal Processing*, vol. 58, pp. 63–78, April 1997.

[81] T. J. Endres, C. R. Johnson, and M. Green, "Robustness to fractionally-spaced equalizer length using the constant modulus criterion," *IEEE Transactions on Signal Processing*, vol. 47, pp. 544–548, February 1999.

[82] J. P. LeBlanc, I. Fijalkow, and C. R. Johnson, "CMA fractionally spaced equalizers: Stationary points and stability under iid and temporally correlated sources," *International Journal of Adaptive Control and Signal Processing*, vol. 12, pp. 135–155, March 1998.

[83] M. Magarini, A. Spalvieri, and G. Tartara, "Asymptotic analysis of stabilisation technique for the blind fractionally spaced equaliser," *Electronics Letters*, vol. 32, pp. 1947–1948, October 1996.

[84] C. B. Papadias and D. T. M. Slock, "Fractionally spaced equalization of linear polyphase channels and related blind techniques based on multichannel linear prediction," *IEEE Transactions on Signal Processing*, vol. 47, pp. 641–654, March 1999.

[85] V. Y. Yang and D. L. Jones, "A vector constant modulus algorithm for shaped constellation equalization," *IEEE Signal Processing Letters*, vol. 5, pp. 89–91, April 1998.

[86] V. Weerackody, S. A. Kassam, and K. R. Laker, "A convergence model for the analysis of some blind equalization algorithms," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 3, pp. 2136–2139, May 1989.

[87] J. E. Mazo, "Analysis of decision-directed equalizer convergence," *The Bell System Technical Journal*, vol. 59, pp. 1857–1876, December 1980.

[88] K. Dogancay and V. Krishnamurthy, "Blind on–line testing for equalization errors in digital communications systems," *IEEE Transactions on Information Theory*, vol. 44, pp. 1677–1686, July 1998.

[89] K. Dogancay and V. Krishnamurthy, "Nonparametric blind testing for equalization errors in mobile communications systems," in *Proceedings of the IEEE Global Telecommunications Conference*, vol. 4, (Sydney, Australia), pp. 2056–2061, 8–12 November 1998.

[90] C. Carlemalm, V. Krishnamurthy, and K. Dogancay, "Algorithms for blind detection of equalization errors in idden Markov model channels," in *Proceedings of the IEEE Conference on Communications*, vol. 1, (Atlanta, USA), pp. 324–328, 7–11 June 1998.

[91] C. Carlemalm, V. Krishnamurthy, and K. Dogancay, "Blind testing for equalization errors for time–varying communications channels," in *Proceedings of the International Conference on Digital Signal Processing*, vol. 1, pp. 119–122, 2–4 July 1997.

[92] K. Dogancay and R. A. Kennedy, "Blind detection of equalization errors in communication systems," *IEEE Transactions on Information Theory*, vol. 43, pp. 469–482, March 1997.

[93] K. Dogancay and R. A. Kennedy, "Testing equalization performance in blind adaptation," in *Proceedings of the IEEE Conference on Decision and Control*, vol. 3, pp. 2817–2818, 14–16 December 1994.

[94] J. R. Treichler and B. G. Agee, "A new approach to multipath correction of constant modulus signals," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-31, pp. 459–472, April 1983.

[95] O. Shalvi and E. Weinstein, "New criteria for blind deconvolution of nonminimum phase systems (channels)," *IEEE Transactions on Information Theory*, vol. IT-36, pp. 312–321, March 1990.

[96] V. Weerackody, S. A. Kassam, and K. R. Laker, "A simple hard-limited algorithm for blind equalization," *IEEE Transactions on Circuits and Systems*, vol. 39, pp. 482–487, July 1992.

[97] H. H. Chiang and C. L. Nikias, "Adaptive deconvolution and identification of nonminimum phase FIR systems based on cumulants," *IEEE Transactions on Automatic Control*, vol. 35, pp. 36–47, January 1990.

[98] A. K. Nandi, "Robust estimation of third-order cumulants in applications of higher-order statistics," *IEE Proceedings-F*, vol. 140, pp. 1169–1172, August 1993.

[99] J. K. Tugnait, "Blind estimation of digital communication channel impulse response," *IEEE Transactions on Communications*, vol. COM-42, pp. 1606–1616, February-April 1994.

[100] J. A. Cadzow, "Blind deconvolution via cumulant extrema," *IEEE Signal Processing Magazine*, vol. 13, pp. 24–42, May 1996.

[101] A. K. Nandi and V. Zarzoso, "Fourth-order cumulant based blind source separation," *IEEE Signal Processing Letters*, vol. 3, pp. 312–314, December 1996.

[102] K. Wesolowsky, "Analysis and properties of the modified constant modulus algorithm for blind equalization," *European Transactions on Telecommunications and Related Technologies*, vol. 3, pp. 225–230, May–June 1992.

[103] J. Choi, I. Song, and R. H. Park, "Some convergence properties of Godard's quartic algorithm," *Signal Processing*, vol. 56, pp. 313–320, February 1997.

[104] M. Gu and L. Tong, "Geometrical characterizations of constant modulus receivers," *IEEE Transactions on Signal Processing*, vol. 47, pp. 2745–2756, October 1999.

[105] P. A. Regalia, "On the equivalence between the Godard and Shalvi–Weinstein schemes of blind equalization," *Signal Processing*, vol. 73, pp. 185–190, February 1999.

[106] F. B. Ueng, "Adaptive lattice–form IIR blind equaliser," *Electronics Letters*, vol. 32, pp. 1869–1870, September 1996.

[107] B. E. Toh, D. C. McLernon, and I. Lakkis, "Enhancing the super–exponential method of blind equalisation with the fast RLS Kalman algorithm," *Electronics Letters*, vol. 32, pp. 92–94, January 1996.

[108] J. Gomes and V. A. N. Barroso, "Super–exponential algorithm for blind fractionally–spaced equalization," *IEEE Signal Processing Letters*, vol. 3, pp. 283–285, October 1996.

[109] F. Herrmann and A. K. Nandi, "Reduced computation blind super-exponential equaliser," *Electronics Letters*, vol. 34, pp. 2208–2210, November 1998.

[110] C. B. Papadias and D. T. M. Slock, "Normalized sliding window constant modulus (CM) and decision-directed algorithms: a link between blind equalization and classical adaptive filtering," *IEEE Transactions on Signal Processing*, vol. 45, pp. 231–235, January 1997.

[111] K. Ozeki and T. Umeda, "An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties," *Electronics && Communications in Japan*, vol. 67-A, no. 5, 1984.

[112] A. G. Constantinides, O. Tanrikulu, B. Buyurman, and J. A. Chambers, "Constant modulus blind equalization algorithms under soft constraint satisfaction," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 97*, vol. 3, pp. 2517–2520, May 1997.

[113] Z. Ding, C. R. Johnson, and R. A. Kennedy, "On the (non)existence of undesirable equilibria of Godard blind equalizers," *IEEE Transactions on Signal Processing*, vol. 40, pp. 2425–2432, October 1992.

[114] Y. Li, K. L. R. Liu, and Z. Ding, "Length –and cost–dependent local minima of unconstrained blind channel equalizers," *IEEE Transactions on Signal Processing*, vol. 44, pp. 2726–2735, November 1996.

[115] Z. Ding, R. A. Kennedy, B. D. O. Anderson, and C. R. Johnson, "Local convergence of the Sato blind equalizer and generalizations under practical constraints," *IEEE Transactions on Information Theory*, vol. 39, pp. 129–144, January 1993.

[116] Z. Ding and R. A. Kennedy, "On the whereabouts of local minima for blind adaptive equalizers," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 39, pp. 119–123, February 1992.

[117] Z. Ding and C. R. Johnson, "On the nonvanishing stability of undesirable equilibria for FTR Godard blind equalizers," *IEEE Transactions on Signal Processing*, vol. 41, pp. 1940–1944, May 1993.

[118] C. R. Johnson and B. D. O. Anderson, "Godard blind equalizer error surface characteristics: White, zero–mean binary source case," *International Journal of Adaptive Control and Signal Processing*, vol. 9, pp. 301–324, July–August 1995.

[119] Y. Li and Z. Ding, "Convergence analysis of finite length blind adaptive equalizers," *IEEE Transactions on Signal Processing*, vol. 43, pp. 2120–2129, September 1995.

[120] H. H. Zeng, L. Tong, and C. R. Johnson, "Relationships between the constant modulus and Wiener receivers," *IEEE Transactions on Information Theory*, vol. 44, pp. 1523–1539, July 1998.

[121] P. A. Regalia and M. Mboup, "Undermodeled equalization: A characterization of stationary points for a family of blind criteria," *IEEE Transactions on Signal Processing*, vol. 47, pp. 760–770, March 1999.

[122] Y. Li and K. L. R. Liu, "Static and dynamic convergence behaviour of adaptive blind equalizers," *IEEE Transactions on Signal Processing*, vol. 44, pp. 2736–2745, November 1996.

[123] V. Weerackody, S. A. Kassam, and K. R. Laker, "Convergence analysis of an algorithm for blind equalization," *IEEE Transactions on Communications*, vol. 39, pp. 856–865, June 1991.

[124] W. Lee and K. Cheun, "Convergence analysis of the stop-and-go blind equalization algorithm," *IEEE Transactions on Communications*, vol. 47, pp. 177–180, February 1999.

[125] Y. Li and Z. Ding, "Global convergence of fractionally spaced Godard (CMA) adaptive equalizers," *IEEE Transactions on Signal Processing*, vol. 44, pp. 818–826, April 1996.

[126] Z. Ding, "On convergence analysis of fractionally spaced adaptive blind equalizers," *IEEE Transactions on Signal Processing*, vol. 45, pp. 650–657, March 1997.

[127] J. J. Shynk and C. K. Chan, "Performance surfaces of the constant modulus algorithm based on a conditional Gaussian model," *IEEE Transactions on Signal Processing*, vol. 41, pp. 1965–1969, May 1993.

[128] S. C. Douglas, A. Cichocki, and S. I. Amari, "Fast-convergence filtered regressor algorithms for blind equalisation," *Electronics Letters*, vol. 32, pp. 2114–2115, November 1996.

[129] K. Dogancay and R. A. Kennedy, "Testing for the convergence of a linear decision directed equalizer," *IEE Proceedings: Vision, Image and Signal Processing*, vol. 141, pp. 129–136, April 1994.

[130] R. Steele and L. Hanzo, eds., *Mobile Radio Communications*. John Wiley & Sons and IEEE Press, 2 ed., 1999.

[131] S. Evans and L. Tong, "Adaptive channel surfing re-initialization of the constant modulus algorithm," in *Proceedings of the 1997 31st Asilomar Conference on Signals, Systems and Computers. Part 1 (of 2)*, (Pacific Grove, USA), pp. 823–827, 2 – 5 November 1997.

[132] S. Evans and L. Tong, "Online adaptive reinitialization of the constant modulus algorithm," *IEEE Transactions on Communications*, vol. 48, pp. 537–539, April 2000.

[133] K. Wesolowsky, "Adaptive blind equalizers with automatically controlled parameters," *IEEE Transactions on Communications*, vol. 43, pp. 170–172, February–April 1995.

[134] B. K. Letaief, Y. Chen, and J. C. I. Chuang, "Blind equalization for short burst TDMA systems in wireless communications," in *Proceedings of the 1997 47th IEEE Vehicular Technology Conference*, pp. 535–538, May 1997.

[135] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and encoding: Turbo-codes (1)," in *IEEE International Conference on Communications*, (Geneva, Switzerland), pp. 1064–1070, 23–26 May 1993.

[136] F. B. Ueng and Y. T. Su, "Adaptive VSS blind equalizers," *IEEE Signal Processing Letters*, vol. 4, pp. 100–102, April 1997.

[137] F. J. Ross and D. P. Taylor, "An enhancement to blind equalization algorithms," *IEEE Transactions on Communications*, vol. 39, pp. 636–639, May 1991.

[138] T. H. Li, "Analysis of a nonparametric blind equalizer for discrete-valued signals," *IEEE Transactions on Signal Processing*, vol. 47, pp. 925–935, April 1999.

[139] F. Riera-Palou, J. M. Noras, and D. G. M. Cruickshank, "Variable length equalizers for broadband mobile systems," in *Proceedings of the IEEE Vehicular Technology Conference*, (Boston, USA), pp. 2478–2485, 24–28 September 2000.

[140] J. B. Anderson and S. Mohan, "Sequential coding algorithms: A survey and cost analysis," *IEEE Transactions on Communications*, vol. COM-32, pp. 1689–1696, February 1984.

[141] Z. Xie, C. K. Rushforth, R. T. Short, and T. K. Moon, "Joint signal detection and parameter estimation in multiuser communications," *IEEE Transactions on Communications*, vol. 41, pp. 1208–1216, August 1993.

[142] V. Krishnamurthy, S. Dey, and J. P. LeBlanc, "Blind equalization of IIR channels using hidden Markov models and extended least squares," *IEEE Transactions on Signal Processing*, vol. 43, pp. 2994–3006, December 1995.

[143] L. Tong, "Blind sequence estimation," *IEEE Transactions on Communications*, vol. 43, pp. 2986–2994, December 1995.

[144] Y. J. Jeng and C. C. Yeh, "Filterbanks for blind channel identification and equalization," *IEEE Transactions on Signal Processing*, vol. 45, pp. 1161–1172, May 1997.

[145] S. Chen, S. McLaughlin, P. M. Grant, and B. Mulgrew, "Joint channel estimation and data detection using a blind Bayesian decision feedback equaliser," in *IEE Colloquium (Digest)*, (London, UK), pp. 4/1–4/5, 27 September 1995.

[146] S. Chen, S. McLaughlin, P. M. Grant, and B. Mulgrew, "Joint channel estimation and data detection using a blind Bayesian decision feedback equaliser," in *Proceeding of the IEEE Global Telecommunications Conference*, vol. 3, (Houston, USA), pp. 2017–2021, 29 September–2 December 1993.

[147] A. Papoulis, *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill International Editions, 2. ed., 1984.

[148] B. Noble and J. W. Daniel, *Applied Linear Algebra.* Englewood Cliffs, New Jersey, USA: Prentice–Hall International Editions, 3. ed., 1986.

[149] J. Mendel, "Tutorial on higher-order statistics (spectra) in signal processing and system th eory: Theoretical results and some applications," *Proceedings of the IEEE*, vol. 79, pp. 278–305, March 1991.

[150] D. Hatzinakos, "Blind equalization based on prediction and polycepstra principles," *IEEE Transactions on Communications*, vol. 43, pp. 178–181, February–April 1995.

[151] D. Hatzinakos, "Blind equalization using decision feedback prediction and tricepstrum principles," *Signal Processing*, vol. 36, pp. 261–276, April 1994.

[152] A. G. Bessios and C. L. Nikias, "POTEA: the power cepstrum and tricoherence equalization algorithm," *IEEE Transactions on Communications*, vol. 43, pp. 2667–2671, November 1995.

[153] A. P. Petropulu and C. L. Nikias, "Blind deconvolution of coloured signals based on higher–order cepstra and data fusion," *IEE Proceedings, Part F: Radar and Signal Processing*, vol. 140, pp. 356–361, December 1993.

[154] S. I. Amari and A. Cichocki, "Adaptive blind signal processing - neural network approaches," *Proceedings of the IEEE*, vol. 86, pp. 2026–2048, October 1998.

[155] C. You and D. Hong, "Nonlinear blind equalization scheme using complex-valued multilayer feedforward neural networks," *IEEE Transactions on Neural Networks*, vol. 9, pp. 1442–1455, November 1998.

[156] Y. Fang and T. W. S. Chow, "Blind equalization of a noisy channel by linear neural network," *IEEE Transactions on Neural Networks*, vol. 10, no. 4, pp. 918–924, 1999.

[157] S. Choi. and A. Cichocki, "Cascade neural networks for multichannel blind deconvolution," *Electronics Letters*, vol. 34, pp. 1186–1187, June 1998.

[158] S. Mo and B. Shafai, "Blind equalization using higher order cumulants and neural network," *IEEE Transactions on Signal Processing*, vol. 42, pp. 3209–3217, November 1994.

[159] B. Jelonnek, D. Boss, and K. D. Kammeyer, "Generalized eigenvector algorithm for blind equalization," *Signal Processing*, vol. 61, pp. 237–264, September 1997.

[160] A. M. Michelson and A. H. Levesque, *Error Control Techniques for Digital Communication.* Wiley-Interscience, 1985.

[161] L. Hanzo, W. T. Webb, and T. Keller, *Single and multicarrier quadrature amplitude modulation.* John Wiley & Sons and IEEE Press, 1999.

[162] G. Reali, G. Baruffa, S. Cacopardi, and F. Frescura, "Enhancing satellite broadcasting services using multiresolution modulations," *IEEE Transactions on Broadcasting*, vol. 44, pp. 497–506, December 1998.

[163] Y. F. Hsu, Y. C. Chen, C. J. Huang, and M. J. Sun, "MPEG-2 spatial scalable coding and transport stream error concealment for satellite TV broadcasting using Ka-band," *IEEE Transactions on Broadcasting*, vol. 44, pp. 77–86, March 1998.

[164] L. Atzori, F. G. B. D. Natale, M. D. Gregario, and D. D. Giusto, "Multimedia information broadcasting using digital TV channels," *IEEE Transactions on Broadcasting*, vol. 43, pp. 383–392, December 1997.

[165] W. Sohn, O. H. Kwon, and J. S. Chae, "Digital DBS system design and implementation for TV and data broadcasting using Koreasat," *IEEE Transactions on Broadcasting*, vol. 44, pp. 316–323, September 1998.

[166] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: turbo codes," *IEEE Transactions on Communications*, vol. 44, pp. 1261–1271, October 1996.

[167] S. B. Wicker, *Error Control Systems for Digital Communication and Storage.* Prentice Hall, 1994.

[168] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and suboptimal MAP decoding algorithms operating in the log domain," in *IEEE Proceedings of the International Conference on Communications*, pp. 1009–1013, June 1995.

[169] J. Griffiths, *Radio Wave Propagation and Antennas - An Introduction.* Prentice Hall, 1987.

[170] M. S. Karaliopoulos and F.-N. Pavlidou, "Modelling the land mobile satellite channel: a review," *Electronics and Communication Engineering Journal*, vol. 11, pp. 235–248, October 1999.

[171] J. Goldhirsh and W. J. Vogel, "Mobile satellite system fade statistics for shadowing and multipath from roadside trees at UHF and L-band," *IEEE Transactions on Antennas and Propagation*, vol. 37, pp. 489–498, April 1989.

[172] W. J. Vogel and J. Goldhirsh, "Multipath fading at L band for low elevation angle, land mobile satellite scenarios," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 197–204, February 1995.

[173] W. J. Vogel and G. W. Torrence, "Propagation measurements for satellite radio reception inside buildings," *IEEE Transactions on Antennas and Propagation*, vol. 41, pp. 954–961, July 1993.

[174] W. J. Vogel and U. S. Hong, "Measurement and modelling of land mobile satellite propagation at UHF and L-band," *IEEE Transactions on Antennas and Propagation*, vol. 36, pp. 707–719, May 1988.

[175] S. R. Saunders, C. Tzaras, and B. G. Evans, "Physical statistical propagation model for mobile satellite channel," tech. rep., European Commission, 1998.

[176] S. Saunders, *Antennas and Propagation for Wireless Communication Systems Concept and Design.* John Wiley & Sons, 1999.

[177] L. Hanzo, P. J. Cherriman, and J. Streit, *Wireless Video Communications: Second to Third Generation and beyond.* IEEE Press, 2001.

[178] A. Bouttier, "Truly recursive blind equalization algorithm," in *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, Part 6 (of 6)*, (Seattle, USA), pp. 3381–3384, May 12 – 15 1998.

[179] L. Tong and D. Liu, "Blind predictive decision-feedback equalization via the constant modulus algorithm," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, v 5, (1997)*, (Munich, Germany), pp. 3901–3904, April 21 – 24 1997.

[180] L. Tong, D. Liu, and H. H. Zeng, "On blind decision feedback equalization," in *Proceedings of the 1996 30th Asilomar Conference on Signals, Systems and Computers, Part 1 (of 2)*, (Los Alamitos, USA), pp. 305–309, Nov 3 – 6 1996.

[181] K. N. Oh and Y. O. Chin, "New blind equalization techniques based on constant modulus algorithm," in *Proceedings of the 1995 IEEE Global Telecommunications Conference, Part 2 (of 3)*, (Singapore), pp. 865–869, Nov 14 – 16 1995.

[182] C. Douillard, M. Jezequel, C. Berrou, A. Picart, P. Didier, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," *European Transactions on Telecommunications and Related Technologies*, vol. 6, pp. 507–511, September–October 1995.

[183] G. Bauch, H. Khorram, and J. Hagenauer, "Iterative equalization and decoding in mobile communications systems," in *Proceedings of the Second European Personal Mobile Communications Conference (EPMCC'97)*, (Bonn, Germany), pp. –, September 1997.

[184] J. P. Woodard and L. Hanzo, "Comparative study of turbo decoding techniques:an overview," *TRVT*, vol. 49, pp. 2208–2233, November 2000.

[185] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Transactions on Information Theory*, vol. IT-28, pp. 55–67, January 1982.

[186] E. Zehavi, "8-PSK trellis codes for a Rayleigh fading channel," *IEEE Transactions on Communications*, vol. 40, pp. 873–883, May 1992.

[187] P. Robertson and T. Worz, "Bandwidth-Efficient Turbo Trellis-Coded Modulation Using Punctured Component Codes," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 206–218, February 1998.

[188] D. Divsalar and M. K. Simon, "The design of trellis coded MPSK for fading channel: Performance criteria," *IEEE Transactions on Communications*, vol. 36, pp. 1004–1012, September 1988.

[189] D. Divsalar and M. K. Simon, "The design of trellis coded MPSK for fading channel: Set partitioning for optimum code design," *IEEE Transactions on Communications*, vol. 36, pp. 1013–1021, September 1988.

[190] X. Li, J. A. Ritcey, "Trellis-Coded Modulation with Bit Interleaving and Iterative Decoding," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 715–724, April 1999.

[191] X. Li, J. A. Ritcey, "Bit-interleaved coded modulation with iterative decoding using soft feedback," *IEE Electronics Letters*, vol. 34, pp. 942–943, May 1998.

Author Index

Index