EXTENSIONS OF THE CONSTANT-MODULUS ALGORITHM AND THE PHASE-LOCKED LOOP FOR BLIND MULTIUSER DETECTION

A Dissertation

Presented to

The Academic Faculty

by

Anuj Batra

In Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy in Electrical Engineering

School of Electrical and Computer Engineering

Georgia Institute of Technology

Atlanta, Georgia

April, 2000

To my parents, Anand and Mridul.

ACKNOWLEDGMENTS

I would like to begin by extending my most sincere thanks to my advisor, Dr. John Barry for guiding me through my graduate studies and research, and for his patience and understanding, without which this dissertation would have never been completed. I have learned much from working with him. I am also grateful to my thesis committee members, Dr. Douglas Williams, Dr. Mary Ann Ingram, Dr. Steven McLaughlin, and Dr. Marcus Spruill for their guidance and support.

I express my deepest gratitude to Dr. David Delchamps, my undergraduate advisor at Cornell, who showed me the beauty of signal processing, communications, and linear systems.

I greatly appreciate the support that I have received from the faculty and staff at Georgia Tech. In particular, I would like to thank Dr. Dale Ray, who has been a constant source of encouragement and comfort during my stay at Georgia Tech. I thank Steve Flynn, Dave Webb, and Joe Amond for their friendship and support with the computer facilities.

My special thanks to my former roommates Ram Rao and James Caffery. Thanks for all the great memories! I am grateful to Jim Kosmach, Joey Arrowood, Haluk Aydinoglu, Mike Gazarik, Chris Lanciani, and Peter Cardillo for their invaluable support and camaraderie. I would especially like to thank Ravi Sivasankaran and Sarat Krishnan who helped me in the editing of my dissertation. I would be remiss if I did not thank my colleagues in

the Communications Theory Research Group: Ricky Causey, Hyuncheol Park and Chen-Chu Yeh.

There are also many people outside of my academic life that I would like to thank, Alysse Rosewater, Tara Lane, and Marcy Lipp for their friendship. I especially thank Jennifer Horton who has always supported me since I met her. Special thanks go to the members of *Our Lady Peace*, Mike, Raine, Jeremy, and Duncan, who provided me with an amazing creative outlet. I am grateful to Natalie Turano, Rob Lanni, and Eric Lawrence who have supported me in my creative endeavors.

Finally, and most importantly, I would like to thank my family. My mother and father have provided me with endless support and infinite wisdom. My father deserves special praise for editing this very long dissertation. I would also thank my brother, Arun for the encouragement he has always provided. I could never say enough to thank them, so I only say that I love them.

AB

TABLE OF CONTENTS

List of Tables x

List of Figures xi

Summary xix

1 INTRODUCTION 1

2 BACKGROUND 6

2.1 Multiuser Channel Model 7

2.1.1 Conversion from Continuous-Time to Discrete-Time 7

2.1.2 Conditions for the Existence of a Zero-Forcing Linear Detector 10

2.1.3 Minimum-Phase Channels 11

2.2 Non-Blind Detection 14

2.2.1 Optimal Detection 15

2.2.2 Suboptimal Detection 16

2.2.3 Adaptive Detection 17

2.3 Blind Detection 19

2.3.1 Blind Single-User Detection 19

2.3.2 Phase-Locked Loop 23

2.3.3 Blind Multiuser Detection 29

3 MULTIDIMENSIONAL PHASE-LOCKED LOOP 36

3.1 Phase-Locked Loop 38

3.1.1 First-Order PLL 38

3.1.2 Convergence Analysis of a First-Order PLL in the Absence of Noise 39

3.1.3 Convergence Analysis of a First-Order PLL in the Presence of Noise 47

3.1.4 Second-Order PLL 52

3.1.5 Convergence Analysis of a Second-Order PLL 54

3.2 Alternative Model for Phase-Locked Loop 57

3.2.1 Alternative-Model First-Order PLL 57

3.2.2 Alternative-Model Second-Order PLL 60

3.3 Multidimensional Phase-Locked Loop 62

3.3.1 Rotation Detector 65

3.3.2 Loop Filter 70

3.3.3 First-Order MPLL 73

3.3.4 Second-Order MPLL 74

3.4 Alternative Model for the MPLL 76

3.5 Convergence Analysis for a First-Order MPLL 82

3.5.1 In the Absence of Noise 82

3.5.2 In the Presence of Noise 94

3.6 Convergence Analysis for a Second-Order MPLL 97

3.7 Experimental Results 99

3.7.1 Noiseless Memoryless Unitary Channel 101

3.7.2 Noisy Memoryless Gaussian Channel 106

3.7.3 Complexity Comparison 116

3.7.4 Trained MPLL versus Decision-Directed MPLL 118

3.8 Summary 120

Appendix 3.1: Proof of Theorem 3-5 122

4 SPATIAL VECTOR CMA 125

4.1 Channel Model and Assumptions 127

4.2 Vector CMA Cost Function 130

4.3 Local Minima in the Absence of Noise 136

4.4 Local Minima in the Presence of Noise 142

4.5 Vector CMA with Gram-Schmidt Constraint 149

4.6 Adaptive Stochastic Algorithm 152

4.6.1 Vector CMA 153

4.6.2 Vector CMA with Gram-Schmidt Constraint 154

4.7 Experimental Results 156

4.7.1 Rotational Ambiguity and Performance Measure 157

4.7.2 Uniform Linear Array 159

4.7.3 Synchronous CDMA 166

4.7.4 Shaped Constellation 172

4.7.5 Computational Complexity 175

4.8 Summary 176

Appendix 4.1: Proof of Theorem 4-3 179

Appendix 4.2: Proof of Lemma 4-1 183

Appendix 4.3: Proof of Lemma 4-2 185

Appendix 4.4: Proof of Theorem 4-4 188

Appendix 4.5: Proof of Lemma 4-3 191

Appendix 4.6: Proof of Theorem 4-5 194

Appendix 4.7: Derivation of (4-99) 198

Appendix 4.8: Proof of Theorem 4-6 204

Appendix 4.9: Derivation of (4-55) 207

Appendix 4.10: Derivation of (4-56) 209

5 SPATIO-TEMPORAL VECTOR CMA 211

5.1 Channel Model and Assumptions 212

5.2 Tall Channels 215

5.3 Vector CMA 218

5.3.1 Cost Function 220
5.3.2 Stochastic Algorithm 222
5.4 Local Minima in the Absence of Noise 223
5.5 Vector CMA with Gram-Schmidt Constraint 231
5.5.1 Cost Function 231
5.5.2 Local Minima in the Absence of Noise 232
5.5.3 Stochastic Algorithm 234
5.6 Performance in Noise 239
5.7 Experimental Results 243
5.7.1 Uniform Linear Array with Multipath 243
5.7.2 Asynchronous CDMA 246
5.7.3 Small First Tap 250
5.7.4 Non-Minimum Square Right Factor 253
5.8 Summary 256
Appendix 5.1: Proof of Theorem 5-2 258
Appendix 5.2: Proof of Lemma 5-1 265
Appendix 5.3: Proof of Lemma 5-2 267
Appendix 5.4: Proof of Theorem 5-3 270
Appendix 5.5: Proof of Theorem 5-4 274
Appendix 5.6: Proof of Theorem 5-5 276
Appendix 5.7: Proof of Theorem 5-6 279

6 CONCLUSIONS AND FUTURE RESEARCH 285

6.1 Conclusions 285
6.2 Future Research 287
6.2.1 MPLL Convergence 287
6.2.2 Fading Channels 288

6.2.3 Undesirable Local Minima for FIR Linear Detectors 288

REFERENCES 290

VITA 301

LIST OF TABLES

3-1 Estimates of the phase error at $\varepsilon_k = \pm 22.5°$ for all points in a 16-QAM input alphabet. 43

3-2 Optimal Parameters for a 3×2 Gaussian Channel with 4-QAM input alphabet. 109

3-3 Optimal Parameters for a 3×2 Gaussian Channel with 16-QAM input alphabet. 112

3-4 Optimal Parameters for a 3×2 Gaussian Channel with shaped 16-QAM input alphabet. 114

3-5 Optimal Parameters for a 5×3 Gaussian Channel with 16-QAM input alphabet. 117

4-1 Optimized Parameters for a Noisy Uniform Linear-Array Application. 164

4-2 Optimized Parameters for a Noisy Synchronous CDMA Application. 170

LIST OF FIGURES

1-1 A block of the general whiten-rotate structure. 4

2-1 A block diagram of a complex baseband $p \times n$ continuous-time channel model. ... 7

2-2 (a) A $p \times n$ continuous-time channel followed by an oversampling receiver front end; (b) an equivalent $m \times n$ baud-rate discrete-time channel model. 9

2-3 A block diagram of an $m \times n$ channel with followed by an $n \times m$ linear detector with memory. 17

2-4 Block diagram of the basic structure of a decision-directed phase-locked loop. The main components for phase-locked loop are the phase detector, loop filter, and complex VCO. 24

2-5 If the ambiguity in the input signal to the PLL is a constant phase offset, then the received constellation will be a tilted version of the transmitted constellation as shown in (a). The o's represent the transmitted 16-QAM constellation and the x's represent the received constellation. If the ambiguity is a constant frequency offset, the received constellation will spin, as illustrated in (b). 26

3-1 A normalized S-curve for a first-order DD PLL with 16-QAM input alphabet. ...42

3-2 The fraction of trials that converge within k symbols is plotted versus the number of symbols for a noiseless first-order DD PLL with 16-QAM input alphabet and various step sizes: (a) the number of symbols ranges from 0 to 100,000; (b) an expanded view of the first 2000 symbols. 45

3-3 Normalized S-curves for a first-order DD PLL with 4-QAM input alphabet and a noisy input signal. 49

3-4 Normalized S-curves for a first-order DD PLL with 16-QAM input alphabet and a noisy input signal. 50

3-5 The fraction of trials that converge within *k* symbols is plotted versus the number of symbols for a noisy first-order DD PLL with 16-QAM input alphabet and various step sizes: (a) the number of symbols ranges from 0 to 10,000; (b) an expanded view of the first 1000 symbols. 53

3-6 A phase-plane portrait for a second-order DD PLL with parameters $\alpha_1 = 0.1$ and $\alpha_2 = 10^{-3}$. 56

3-7 Block diagram of the basic structure of a first-order PLL: (a) conventional model consisting of a phase detector, loop filter, and complex VCO; (b) alternative model consisting of a rotation detector, loop filter, and a product accumulator. 58

3-8 A graphical representation of a complete and partial rotation on the unit circle. 59

3-9 Block diagram of the basic structure of a decision-directed multidimensional phase-locked loop. The main components for MPLL are the rotation detector, loop filter, and product accumulator. 63

3-10 A comparison of the total number of multiplications that are needed to generate the update in (3-58) and (3-59). 75

3-11 The squared error between the two intermediate points $\boldsymbol{v}_\lambda$ and $\boldsymbol{v}_\mu$ as a function of μ. 79

3-12 Comparison of the conventional-model MPLL and the alternative-model MPLL in terms of MSE_1 versus time. 80

3-13 Comparison of computational complexity for the conventional-model MPLL and the alternative-model MPLL. 81

3-14 The fraction of trials that converge within k symbols is plotted versus the number of symbols for a noiseless two-user first-order MPLL with 16-QAM input alphabet and various step sizes: (a) the number of symbols ranges from 0 to 500,000; (b) an expanded view of the first 10,000 symbols. 89

3-15 The fraction of trials that converge within k symbols is plotted versus the number of symbols for a noiseless three-user first-order MPLL with 16-QAM input alphabet and various step sizes: (a) the number of symbols ranges from 0 to 500,000; (b) an expanded view of the first 20,000 symbols. 91

3-16 The fraction of trials that converge within k symbols is plotted versus the number of symbols for a noiseless four-user first-order MPLL with 16-QAM input alphabet and various step sizes: (a) the number of symbols ranges from 0 to 500,000; (b) an expanded view of the first 20,000 symbols. 92

3-17 The minimum step size that guarantees convergence within 500,000 symbols λ_{min} is plotted versus the number of users n. 93

3-18 The fraction of trials that converge within k symbols is plotted versus the number of symbols for a noisy three-user first-order MPLL with 16-QAM input alphabet, various step sizes, and $SNR_1 = 20$ dB: (a) the number of symbols ranges from 0 to 250,000; (b) an expanded view of the first 25,000 symbols. 96

3-19 A phase-plane portrait for a second-order MPLL with parameters $\lambda_1 = 0.3$ and $\lambda_2 = 0.1$. 98

3-20 A block diagram of an $m \times n$ memoryless channel followed by an $n \times m$ ideal whitener. This model is used to generate the input signal for all computer simulations. 99

3-21 Comparison of the MPLL, JADE, and EASI, in terms of MSE_1 versus time, for a 2×2 noiseless unitary channel. The shaded regions represent the variation in convergence time for each of the detector. For each detector, the lower curve is an average of the fastest 10% of the trials, the middle represents the average MSE, and the upper curve is an average of the slowest 10% of the trials. 101

3-22 Histogram of convergence times for a 2×2 unitary channel: (a) MPLL; (b) EASI; (c) JADE. 104

3-23 Comparison of the MPLL, JADE, and EASI, in terms of MSE_1 versus time, for a 3×3 noiseless unitary channel. The shaded regions represent the variation in convergence time for each of the detector. For each detector, the lower curve is an average of the fastest 10% of the trials, the middle represents the average MSE, and the upper curve is an average of the slowest 10% of the trials. 105

3-24 Histogram of convergence times for a 3×3 unitary channel: (a) MPLL; (b) EASI; (c) JADE. 107

3-25 The convergence time for each detector is plotted versus SNR_1 for a 3×2 noisy complex Gaussian channel followed by a 2×3 ideal whitener and a 4-QAM input alphabet. The optimal parameters for each detector can be

found in Table 3-2. 108

3-26 The convergence time for each detector is plotted versus SNR_1 for a 3×2 noisy complex Gaussian channel followed by a 2×3 ideal whitener and a 16-QAM input alphabet. The optimal parameters for each detector can be found in Table 3-3. 111

3-27 The convergence time for each detector is plotted versus SNR_1 for a 3×2 noisy complex Gaussian channel followed by a 2×3 ideal whitener and a shaped 16-QAM input alphabet. The optimal parameters for each detector can be found in Table 3-4. 113

3-28 The convergence time for each detector is plotted versus SNR_1 for a 5×3 noisy complex Gaussian channel followed by a 3×5 ideal whitener and a 16-QAM input alphabet. The optimal parameters for each detector can be found in Table 3-5. 116

3-29 Comparison of computational complexity of MPLL, JADE, and EASI. 118

3-30 Comparison of a trained MPLL and a decision-directed MPLL in terms of MSE_1 versus time. 119

4-1 A block diagram of an $m \times n$ noisy memoryless channel model. 127

4-2 A block diagram of an $m \times n$ memoryless channel followed by an $n \times m$ memoryless linear detector. 131

4-3 A noiseless single-user system with a complex Gaussian input signal. 138

4-4 A comparison of the mean-squared error versus SNR of the vector CMA detector and the minimum-MSE detector. 148

4-5 A block diagram of a memoryless channel followed by a memoryless linear detector and a memoryless unitary rotator. 157

4-6 Comparison of the vector CMA detector, decorrelation CMA detector, combination CMA detector, and the PWR detector, in terms of MSE_1 versus time, for a noiseless 16×3 uniform linear-array with half-wavelength spacing application with 16-QAM input alphabet, assuming both actual rotators and MMSE rotators. 161

4-7 Comparison of the convergence time of the three detector versus SNR_1 for a 16×3 uniform linear-array with half-wavelength spacing application with 16-QAM input alphabet. The optimal parameters for each detector can be found in Table 4-1. 163

4-8 Comparison of the total complexity of the three detector versus SNR_1 for a 16×3 uniform linear-array with half-wavelength spacing application with 16-QAM input alphabet. The optimal parameters for each detector can be found in Table 4-1. 165

4-9 Comparison of the vector CMA detector, decorrelation CMA detector, and the PWR detector, in terms of MSE_1 versus time, for a noiseless synchronous CDMA application with 4-QAM input alphabet, assuming both actual rotators and MMSE rotators. 167

4-10 Comparison of the convergence time of the three detector versus SNR_1 for a synchronous CDMA application. The optimal parameters for each detector can be found in Table 4-2. 169

4-11 Comparison of the total complexity of the three detector versus SNR_1 for a synchronous CDMA application.The optimal parameters for each detector can be found in Table 4-2. 171

4-12 The vector CMA detector applied to a uniform linear-array with half-wavelength spacing and a shaped 16-QAM input alphabet: (a) learning curves, assuming both an actual MPLL and an MMSE MPLL; (b) constellations from the last trial, baud 9000 to 10,000. 174

4-13 Comparison of the computational complexity of the various detectors versus the number of users n for a fixed number of sensors $m = 20$. 175

5-1 A block diagram of an $m \times n$ noisy channel with memory. 213

5-2 A block diagram of an $m \times n$ channel with followed by an $n \times m$ linear detector with memory. 219

5-3 A comparison of the mean-squared error versus SNR of the optimal vector CMA with GSC detector, the optimal minimum-MSE detector, and optimal forward-backward LP detector. 241

5-4 Comparison of the vector CMA with GSC detector and decorrelation CMA detector, in terms of MSE_1 versus time, for a 2×2 uniform linear with multipath. 244

5-5 Models for a two-user asynchronous CDMA application: (a) continuous-time model with a chip-rate receiver; (b) equivalent discrete-time model. 246

5-6 Comparison of the vector CMA with GSC detector, decorrelation CMA detector, and a zero-delay forward LP detector, in terms of MSE_1 versus time,

for a two-user asynchronous CDMA application. 249

5-7 Comparison of the vector CMA with GSC detector and the forward LP, in terms of MSE_1 versus time, for a channel with a small first tap. 252

5-8 Comparison of the vector CMA with GSC detector and the forward LP, in terms of MSE_1 versus time, for a reducible tall channel. 255

SUMMARY

Multiuser detection is the process of mitigating interference among users in a multiuser communications system. This dissertation addresses the problem of blind multiuser detection, where the transmitters do not provide training sequences or other assistance to the receiver. We propose new low-complexity, adaptive, linear blind detection algorithms for square and tall channels by extending two well-established single-user algorithms, the constant-modulus algorithm (CMA) and the phase-locked loop (PLL), to multiuser channel framework.

We propose a multidimensional phase-locked loop (MPLL) for blindly resolving a unitary ambiguity. The MPLL, a multidimensional generalization of the PLL, is a decision-directed algorithm that exploits the discrete nature of digital communication signals. We investigate the convergence behavior of a first-order and second-order MPLL. Using computer simulations, we show that the MPLL offers fast convergence, low complexity, and excellent steady-state performance.

For memoryless channels, we propose the vector constant-modulus algorithm, which is a unique generalization of the CMA to vector-valued signals. In the absence of noise, we show that the vector CMA detector is a whitener for all non-CM input alphabets. As a result, this detector is compatible with shaped-input alphabets. In the presence of noise, this detector displays near MMSE-like performance. We also propose another whitener, the vector CMA with Gram-Schmidt constraint detector, that can be used for all input

alphabets. Using numerical examples, we compare that the performance and complexity of these two detectors with other blind detectors.

We also extend both the vector CMA and vector CMA with GSC detectors to channels with memory. In the absence of noise, we show that the vector CMA detector converges to both unitary and non-unitary matrices and that the vector CMA with GSC detector is a whitener for a sub-Gaussian input alphabet. In the presence of noise, the performance of optimal vector with GSC detector is similar to that of the optimal MMSE detector. Finally, through simulations, we demonstrate that the vector CMA with GSC detector compares favorably with other blind detectors.

CHAPTER 1

INTRODUCTION

With the advent of cellular technology, more and more people are using digital mobile phones for personal as well as business needs, which places greater demands on service providers to ensure reliable voice and data transmissions over a wireless communications channel. Improvements in integrated chip technology, which have led to cheaper phones, and substantial reductions in the cost of providing services, have resulted in an overwhelming popularity of the digital mobile phones, which in turn has led to an exponential growth in the transmission of digital information. Unfortunately, the available spectrum for transmission of data from mobile phones is finite. So it is inevitable that multiple users share the same transmission medium and frequencies; in other words, they transmit data over a multiple-access or multiuser communications channel [1-3].

A consequence of having multiple users transmit asynchronously and from different geographic locations over a common channel is that these users interfere with one another. Traditionally, transmission protocols, such as FDMA, TDMA, and CDMA, are used at the transmitter to prevent multiuser interference at the receiver [4]. Each of these techniques seeks to eliminate multiuser interference by orthogonalizing the transmit signals. Unfortunately, in practice, these techniques can only reduce, and not eliminate the multiuser inter-

ference. For example, in systems that employ either FDMA or TDMA, multiuser interference arises from nearby cells re-using the same carrier frequency and from imperfections in either bandpass filters or system timing [5]. In systems that employ CDMA, the combination of asynchronous transmissions and multipath propagation can destroy orthogonality of the spreading sequences at the receiver and lead to significant multiuser interference [6].

The digital wireless cellular network is only one example of a communications system where multiuser interference is present. Other examples include satellite communications, local-area networks, multi-track magnetic recording systems [7, 8], fixed wireless local loops, digital radio, interactive television, twisted-pair bundles [9,10], and dually-polarized radio [11]. Since the multiuser interference has considerable structure, large gains in performance can be achieved by using suitable signal processing techniques to exploit this structure at the receiver. Although the gains in performance come at the expense of complexity at the receiver, this trade-off is often acceptable in many applications [1,2,12,13]. The design and analysis of detectors in the presence of multiuser interference are of much importance, and research in this area has led to the development of *multiuser detection theory.*

Even though digital communication systems were viewed in terms of a multiuser channel as early as the 1960s, it was not until the mid 1980s that the first algorithms were designed to exploit the structure of the multiuser channel [12]. The discrete-time multiuser channel model is described below:

$$\boldsymbol{r}_k = \mathbf{H}_0\boldsymbol{x}_k + \mathbf{H}_1\boldsymbol{x}_{k-1} + \ldots + \mathbf{H}_M\boldsymbol{x}_{k-M} + \boldsymbol{n}_k, \tag{1-1}$$

where $\boldsymbol{x}_k$ is an $n \times 1$ channel input vector whose components are the discrete symbols transmitted by the n independent users, $\mathbf{H}(z) = \mathbf{H}_0 + \mathbf{H}_1 z^{-1} + \ldots + \mathbf{H}_M z^{-M}$ is the $m \times n$ channel transfer function matrix, $\boldsymbol{n}_k$ is an $m \times 1$ noise vector, and $\boldsymbol{r}_k$ is an $m \times 1$ vector sequence comprised of the receiver observations. The dimension of the observations, or equivalently, the number of virtual sensors, m, is related to the number of samples per baud β and the number of actual sensors at the receiver, p, according to $m = \beta p$.

Since the mid 1980s, several different detection strategies, such as maximum-likelihood sequence detectors [14,15], linear detectors [9,16,17,18-20], and decision-feedback detectors [19-24] have been developed for the case when the multiuser channel response is known or when a training sequence is available. In certain applications, such as broadcast digital television, interactive cable television, and certain non-cooperative environments, the channel response is either unknown or the receiver does not have access to a training sequence. In these situations, the receiver must adapt according to a blind detection algorithm [25]. The blind multiuser detection problem requires the determination of the transmitted sequence $\boldsymbol{x}_k$ using only the observations $\boldsymbol{r}_k$ and the information about the properties of the transmitted sequence, such as the modulation scheme for each user. An inherent drawback of blind multiuser detection is its inability to distinguish between the different users. Therefore, it is impossible to recover information from a particular user. Hence, we propose to design algorithms that recover the entire transmit vector $\boldsymbol{x}_k$ and to use some higher-level processing to differentiate between the individual users.

This research considers the design of a blind multiuser detector in the presence of both the multiuser and the intersymbol interference and noise. Our primary objective is to develop low-complexity, adaptive, linear blind detection algorithms for square ($m = n$)

and tall ($m > n$) channels. In our work, we consider only linear detectors, as opposed to nonlinear detectors, because they are more amenable to blind implementation. The basic strategy that we have followed is to extend well-established single-user algorithms to the multiuser channel framework. The new blind multiuser detectors are fashioned after two time-tested single-user blind algorithms: the constant-modulus algorithm (CMA) [26] and the decision-directed phase-locked loop (PLL) [1-3,27-32]. These two algorithms are adaptive, well understood, purported to be more robust in the presence of noise, and have lower computational complexity than many batch-oriented and signal subspace algorithms.

It is often convenient to decompose the blind multiuser detection process into two steps, as illustrated in Fig. 1-2: *whiten* and then *rotate* [33-54]. The first step eliminates the channel memory and whitens the output by using an $n \times m$ whitening filter $\mathbf{W}(z)$. The cascade of the whitening filter $\mathbf{W}(z)$ and the channel matrix $\mathbf{H}(z)$ leaves a unitary ambiguity. The second step in the process resolves this ambiguity by using $n \times n$ rotating filter $\mathbf{U}$. In the following chapters, we develop algorithms that perform both the whitening and rotating tasks.

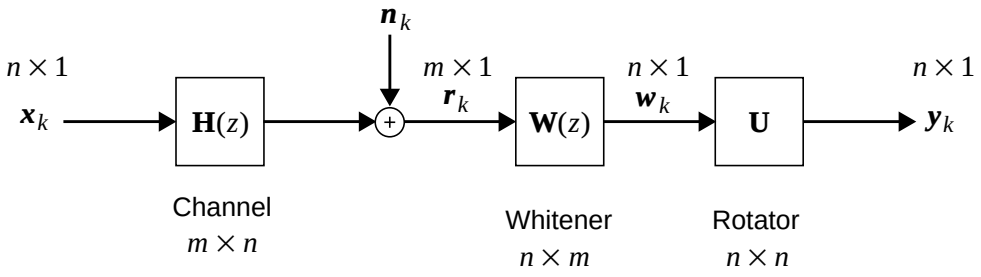


Fig. 1-1. A block of the general whiten-rotate structure.

The remainder of this dissertation is organized as follows. In Chapter 2, we review some of the key concepts from multiuser linear system theory. We also determine the conditions required for the existence of a linear detector for both a continuous-time channel and a discrete-time channel. We then present a survey of prior work in multiuser detection theory. We begin Chapter 3 by reviewing a first-order and second-order phase-locked loop. We then present convergence analyses for these PLLs. Using an alternative model for the PLL, we extend the algorithm to multiple dimensions. This algorithm, which is referred to as the multidimensional PLL (MPLL), is a rotator that can blindly estimate a unitary ambiguity. Finally in this chapter, we present convergence analyses for first-order and second-order MPLLs. In Chapter 4, we present a novel and unique generalization of the constant-modulus algorithm to multiple dimensions. We show that this algorithm and a modification of this algorithm can whiten a memoryless channel for all discrete input alphabets. In Chapter 5, we generalize the whitening detectors presented in Chapter 4 to channels with memory. Finally in Chapter 6, we summarize the key contributions of this research and present ideas for future work.

CHAPTER 2

BACKGROUND

While most communication channels are inherently continuous time, the growing trend in designing receivers is towards digital processing of the received signal. Thus, an equivalent discrete-time model of the continuous-time channel is required. The question naturally arises: given a discrete-time channel, how can we recover the transmitted information? The answer to this question is the focus of this research, with particular emphasis on the design of blind linear detection strategies. In this chapter, we review key concepts that will be used throughout the remainder of the dissertation.

In Section 2.1, we begin by converting the continuous-time channel into an equivalent discrete-time using a canonical receiver front end. We also state the conditions for existence of a zero-forcing linear detector. We then review the notion of minimum phase for discrete-time multiuser channels. In the later part of this chapter, we present a survey of prior work in multiuser detection theory. In Section 2.2, we review the different detection strategies for non-blind detectors, where knowledge of the channel or a training sequence is assumed. Finally in Section 2.3, we review blind equalization methods for a single-user channel, a decision-directed phase-locked loop, and the literature concerning the general problem of blind multiuser detection.

2.1 MULTIUSER CHANNEL MODEL

With the advent of integrated circuit technology, a great deal of the receiver processing is now being performed in discrete time. Indeed, in the following chapters, it is assumed that we have a discrete-time channel for the systems described therein. However, in most communication systems, the underlying physical channel is continuous-time in nature. It is important, therefore, to review the relationship between the underlying continuous-time channel and its equivalent discrete-time model, because sometimes a discrete-time model may obscure interesting and exploitable properties of the continuous-time channel. For example, a baud-rate discrete-time channel hides the inherent cyclostationarity of the underlying continuous-time channel model.

2.1.1 Conversion from Continuous-Time to Discrete-Time

Consider the following baseband additive-white Gaussian noise (AWGN) continuous-time multiuser channel depicted in Fig. 2-1, in which the p continuous-time complex-valued received signals are grouped into a $p \times 1$ vector $\boldsymbol{r}(t)$:

$$\boldsymbol{r}(t) = \sum_{j=-\infty}^{\infty} \mathbf{H}(t - jT)\boldsymbol{x}_j + \boldsymbol{n}(t), \tag{2-1}$$

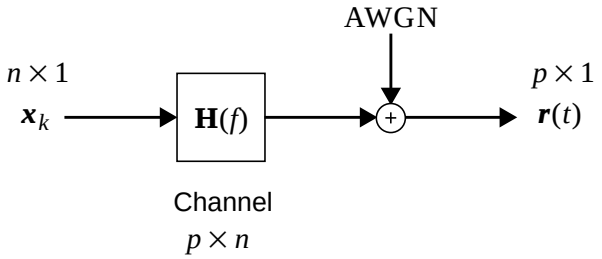


Fig. 2-1. A block diagram of a complex baseband $p \times n$ continuous-time channel model.

where $\boldsymbol{x}_k$ is an $n \times 1$ vector composed of the discrete symbol sequences transmitted by the n users, $\boldsymbol{n}(t)$ is a $p \times 1$ zero-mean white complex Gaussian noise vector with power spectral density (PSD) $\boldsymbol{S}_n(f) = \int_{-\infty}^{\infty} E[\boldsymbol{n}(t)\boldsymbol{n}(t\text{-}\tau)^*]e^{-j2\pi f\tau}d\tau = N_0\mathbf{I}$, and T is the symbol period for all users. The (i, j)-th element of the $p \times n$ matrix $\mathbf{H}(t)$, referred to as the *continuous-time impulse response*, is the response of the i-th output element when an impulse is applied to the j-th input element. The $p \times n$ matrix $\mathbf{H}(f)$, which is referred to as the *continuous-time channel transfer function*, is the corresponding Fourier transform of $\mathbf{H}(t)$. We assume that $\mathbf{H}(f)$ includes the pulse shaping for each user, which is usually bandlimited.

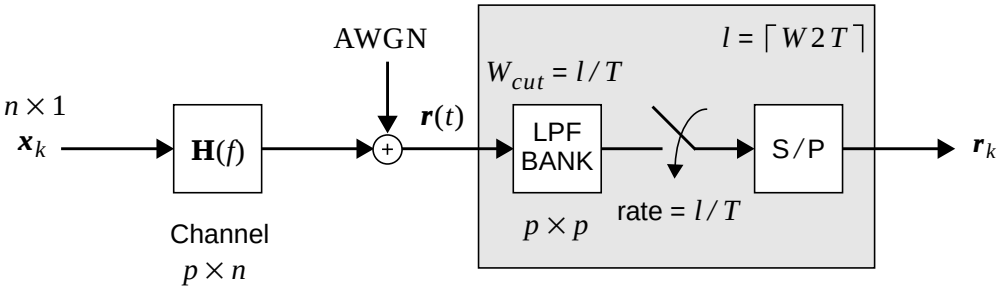
The authors of [53,54] have shown that the continuous-time signal can be discretized by using the receiver front-end illustrated in Fig. 2-2. We assume that each component of the channel $\mathbf{H}(f)$ is bandlimited to $|f| \leq W$. This front end consists of 3 components: a bank of ideal anti-aliasing lowpass filters (LPFs), each with a cut-off frequency of $\lceil W2T \rceil / 2T$, an ideal sampler with a rate of $\lceil W2T \rceil / T$, and a serial-to-parallel (S/P), which concatenates the $\lceil W2T \rceil$ samples into an $m \times 1$ vector $\boldsymbol{r}_k$, where $m = \lceil W2T \rceil p$. This sampling rate was chosen for two reasons: it avoids aliasing, and it ensures that there will be an integral number of samples per baud. As shown in [53,54], the resulting discrete-time received vector $\boldsymbol{r}_k$ can be modeled as the output of an equivalent discrete-time channel:

$$\boldsymbol{r}_k = \sum_{j=-\infty}^{\infty} \mathbf{H}_j \boldsymbol{x}_{k-j} + \boldsymbol{n}_k, \tag{2-2}$$

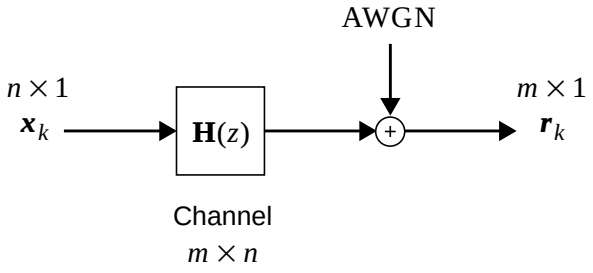
where $\boldsymbol{n}_k$ is an $m \times 1$ zero-mean white Gaussian noise vector with power spectral density $\sigma^2\mathbf{I}$, with $\sigma^2 = N_0\lceil W2T \rceil / T$. The $m \times n$ discrete-time channel $\mathbf{H}(z)$ can be expressed in terms of the continuous-time channel transfer function $\mathbf{H}(f)$ as:

$$\mathbf{H}(z) = \frac{1}{T} \sum_{k=-\infty}^{\infty} \left[\mathbf{I}_p \quad e^{-j2\pi\left(f-\frac{k}{T}\right)\frac{T}{l}} \mathbf{I}_p \quad \cdots \quad e^{-j2\pi\left(f-\frac{k}{T}\right)\frac{(l-1)T}{l}} \mathbf{I}_p \right]^T \mathbf{H}(f), \qquad (2\text{-}3)$$

where $l = \lceil W2T \rceil$.



(a)



(b)

Fig. 2-2. (a) A $p \times n$ continuous-time channel followed by an oversampling receiver front end; (b) an equivalent $m \times n$ baud-rate discrete-time channel model.

We are now in a position to discuss some of the properties of the discrete-time channel transfer function matrix $\mathbf{H}(z)$. From (2-2), it is clear that the channel is causal if $\mathbf{H}_j = \mathbf{0}$ for all $j < 0$ and the channel is anti-causal if $\mathbf{H}_j = \mathbf{0}$ for all $j > 0$. The channel described by (2-2) has an infinite-impulse response (IIR). If the support of the channel is finite, then the channel transfer function matrix is said to have a finite-impulse response (FIR). In many real-world applications, an IIR channel can be approximated by an FIR channel with a large number of taps. This channel is stable if and only if $\mathbf{H}(z)$ converges uniformly on the unit circle [55], *i.e.*,

$$\sum_{k=-\infty}^{\infty} \|\mathbf{H}_k z^{-k}\|_1 < \infty. \tag{2-4}$$

Finally, this channel has full rank if and only if $\mathbf{H}(z_0)$ has full rank for all z_0 on the unit circle ($|z_0| = 1$) [56].

2.1.2 Conditions for the Existence of a Zero-Forcing Linear Detector

Linear detection of all n users is possible if and only if the intersymbol interference and the multiuser interference can be completely removed using only linear filtering, or in other words, if there exists an $n \times m$ stable linear time-invariant filter $\mathbf{C}(z)$ satisfying $\mathbf{C}(z)\mathbf{H}(z) = \mathbf{I}$. The $n \times m$ filter that satisfies $\mathbf{C}(z)\mathbf{H}(z) = \mathbf{I}$ is called a *zero-forcing linear detector* because it forces both the intersymbol and the multiuser interference to zero. As shown in [53,54], the condition for the existence of a zero-forcing linear detector is directly related to the rank of the discrete-time channel matrix on the unit circle.

Theorem 2-1. For an $m \times n$ multiuser discrete-time channel $\mathbf{H}(z)$, a zero-forcing linear detector $\mathbf{C}(z)$ exists if and only if $\mathbf{H}(z)$ has full rank on the unit circle.

Hence, the transmitted sequence $\boldsymbol{x}_k$ can be recovered using only a linear detector if and only if $\mathbf{H}(z)$ has full rank on the unit circle. Since the rank of a channel is inherently related to the determinant of the channel, we can also express the result of the previous theorem as follows [57]:

Corollary 2-1. For an $m \times n$ multiuser discrete-time channel $\mathbf{H}(z)$, a zero-forcing linear detector $\mathbf{C}(z)$ exists if and only if $det(\mathbf{H}^*(1/z^*)\mathbf{H}(z)) \neq 0$ for all z on the unit circle ($|z| = 1$).

2.1.3 Minimum-Phase Channels

Minimum-phase channels are an important class of channels in single-user communication theory. These are stable channels characterized by the fact that all of their poles and zeros lie inside the unit circle, *i.e.*, the channels are stable and causal. As a result, a minimum-phase channel has many special properties. For example, its inverse is minimum phase and its spectrum completely describes the channel. Delfosse has extended the concept of minimum phase to multiuser channels [33].

Definition 2-1. An $m \times n$ casual stable discrete-time channel $\mathbf{H}(z)$ is said to be *minimum phase* if and only if there exists a causal stable zero-forcing linear detector.

This definition reduces to the familiar definition for a single-user minimum-phase channel when $m = n = 1$ [55]. Using this definition, we can determine the conditions for which a square FIR multiuser channel is minimum phase.

Theorem 2-2. A square $n \times n$ FIR causal channel of the form: $\mathbf{H}(z) = \mathbf{H}_0 + \mathbf{H}_1 z^{-1} + \ldots + \mathbf{H}_M z^{-M}$, where M is the channel memory, is minimum phase if and only if $det(\mathbf{H}(z))$ is minimum phase.

Proof: The channel $\mathbf{H}(z)$ is minimum phase if and only there exists a causal stable inverse. A left-inverse, for this channel, is given by:

$$\mathbf{C}(z) = \mathbf{H}(z)^{-1} = \frac{1}{det(\mathbf{H}(z))} adj(\mathbf{H}(z)), \tag{2-5}$$

where $adj(\cdot)$ represents the adjoint of a matrix. Since $\mathbf{H}(z)$ is causal and FIR, $adj(\mathbf{H}(z))$ will also be causal and FIR. Because $adj(\mathbf{H}(z))$ is FIR, it is also stable. Therefore, $\mathbf{C}(z)$ will be a casual stable left-inverse of $\mathbf{H}(z)$ if and only if $\frac{1}{det(\mathbf{H}(z))}$ is causal and stable. Since $det(\mathbf{H}(z))$ is an FIR polynomial in z, the zeros of $\frac{1}{det(\mathbf{H}(z))}$ lie inside the unit circle. For $\frac{1}{det(\mathbf{H}(z))}$ to be causal and stable, the poles of this function must also lie inside the unit circle. In other words, $\frac{1}{det(\mathbf{H}(z))}$ is causal and stable if and only if the poles and zeros of $det(\mathbf{H}(z))$ lie inside the unit circle. This last condition implies that $det(\mathbf{H}(z))$ must be minimum phase. ❑

Hence, a square FIR causal channel $\mathbf{H}(z)$ is minimum phase if and only if the channel has full rank and the determinant is minimum phase.

We can now determine the conditions for a tall FIR multiuser channel to be minimum phase.

Theorem 2-3. [54,34-35] A tall $m \times n$ FIR causal channel of the form: $\mathbf{H}(z) = \mathbf{H}_0 + \mathbf{H}_1 z^{-1} + \ldots + \mathbf{H}_M z^{-M}$, where M is the channel memory, is minimum phase if and only if $rank(\mathbf{H}_M) = rank(\mathbf{H}(z)) = n$ for all z including ∞.

Proof: First, we prove the "if" portion of the proof. As shown in [54], if $rank(\mathbf{H}_M) = rank(\mathbf{H}(z)) = n$ for all values of z including ∞, then the following moving average (MA) channel: $\boldsymbol{r}_k = \sum_{i=0}^{M} \mathbf{H}_i \boldsymbol{x}_{k-i}$, where the input sequence $\boldsymbol{x}_k$ is white: $E[\boldsymbol{x}_k \boldsymbol{x}_{k-l}^*] = \mathbf{I}\delta_l$, can also be expressed in terms of an autoregressive (AR) channel: $\boldsymbol{r}_k = \sum_{i=1}^{N} \mathbf{A}_i \boldsymbol{r}_{k-i} + \mathbf{H}_0 \boldsymbol{x}_k$, where the feedback filter $\mathbf{A}(z) = \sum_{i=1}^{N} \mathbf{A}_i z^{-i}$ is square ($m \times m$) and strictly causal. By equating the transfer functions of the MA and AR channels, we find that $\mathbf{H}(z) = [\mathbf{I} - \mathbf{A}(z)]^{-1}\mathbf{H}_0$, or equivalently, $[\mathbf{I} - \mathbf{A}(z)]\mathbf{H}(z) = \mathbf{H}_0$. We observe that $[\mathbf{I} - \mathbf{A}(z)]$ is causal and FIR. It follows that $\mathbf{C}(z) = \mathbf{H}_0^{\dagger}[\mathbf{I} - \mathbf{A}(z)]$, where $\mathbf{H}_0^{\dagger} = (\mathbf{H}_0^* \mathbf{H}_0)^{-1}\mathbf{H}_0^*$, is a causal FIR left-inverse of $\mathbf{H}(z)$. Since $\mathbf{C}(z)$ is FIR, it is also stable. Hence, $\mathbf{C}(z)$ is a causal stable left-inverse of $\mathbf{H}(z)$ and therefore, $\mathbf{H}(z)$ is a minimum-phase channel.

Next, we prove the "only if" portion of the proof. If $\mathbf{H}(z)$ is minimum phase, then there exists a causal stable filter $\mathbf{C}(z)$ that satisfies $\mathbf{C}(z)\mathbf{H}(z) = \mathbf{I}$. This constraint can also be expressed in block-matrix notation:

$$[\mathbf{C}_0 \; \mathbf{C}_1 \; \ldots \; \mathbf{C}_N]\mathcal{H} = [\mathbf{I} \; \mathbf{0} \; \ldots \; \mathbf{0}], \tag{2-6}$$

where $\mathbf{C}(z) = \mathbf{C}_0 + \mathbf{C}_1 z^{-1} + \ldots + \mathbf{C}_N z^{-N}$ and H is the $(N+1)m \times (M+N+1)n$ block-Toeplitz matrix:

$$H = \begin{bmatrix} \mathbf{H}_0 & \mathbf{H}_1 & \cdots & \mathbf{H}_M & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_0 & \mathbf{H}_1 & \cdots & \mathbf{H}_M & \mathbf{0} & \cdots \\ & & \ddots & & & \ddots & \\ \mathbf{0} & & \cdots & \mathbf{0} & \mathbf{H}_0 & \mathbf{H}_1 & \cdots \mathbf{H}_M \end{bmatrix}. \tag{2-7}$$

Since $\mathbf{C}(z)$ exists, the block-Toeplitz matrix H must have full-row rank. Forney showed in [58] that H has full-row rank only if $rank(\mathbf{H}_M) = rank(\mathbf{H}(z)) = n$ for all z including ∞. ❑

Thus, every tall FIR causal channel $\mathbf{H}(z)$ that satisfies the condition that $rank(\mathbf{H}_M) = rank(\mathbf{H}(z)) = n$ for all z including ∞ is minimum phase.

2.2 NON-BLIND DETECTION

In the first part of this chapter, we derived the discrete-time multiuser channel and determined the conditions for the existence of a zero-forcing linear detector. The question still remains: how can we recover the transmitted data? The answer to this question depends on how much information is available. We can classify the detection strategies into two groups: non-blind detection, where we either know the channel or have access to a training sequence; and blind detection, where we only have knowledge of the statistics of the transmitted symbols. We begin this section by examining non-blind detection strategies. In the next section, we review both single-user and multiuser blind detection strategies.

2.2.1 Optimal Detection

Consider the $m \times n$ multiuser channel model described by (1-1), where the received vector is given by:

$$\boldsymbol{r}_k = \mathbf{H}_0\boldsymbol{x}_k + \mathbf{H}_1\boldsymbol{x}_{k-1} + \ldots + \mathbf{H}_M\boldsymbol{x}_{k-M} + \boldsymbol{n}_k. \tag{2-8}$$

If $\boldsymbol{n}_k$ is a zero-mean white Gaussian noise vector, then the optimal detector for this multiuser channel is the *maximum-likelihood sequence detector* (MLSD), which was derived by Van Etten [14,15] following the single-user approach suggested by Forney [59]. The MLSD generates an estimate of the input sequence by minimizing the Euclidean distance between the received sequence and a noiseless received sequence:

$$\min_{\hat{\boldsymbol{x}}_k} \sum_{k=1}^{\infty} \left\| \boldsymbol{r}_k - \sum_{i=0}^{M} \mathbf{H}_i \hat{\boldsymbol{x}}_{k-i} \right\|^2. \tag{2-9}$$

It can be efficiently implemented via a Viterbi decoder with L^{nM} states, where L is the size of the input alphabet, n is the number of users, and M is the channel memory. The Viterbi algorithm [60] is quite efficient, but it can be computationally complex even for moderate values of L, n, and M, thus making the MLSD impractical for most real-world applications. Therefore, much research effort has been recently directed towards developing reduced-complexity techniques having near-optimal performance.

2.2.2 Suboptimal Detection

As we mentioned previously, the zero-forcing linear detector forces both the intersymbol and the multiuser interference to zero in the absence of noise. The transfer function for the detector is given by $\mathbf{C}_{ZF}(z) = \mathbf{H}(z)^{\dagger} = [\mathbf{H}^{*}(1/z^{*})\mathbf{H}(z)]^{-1}\mathbf{H}^{*}(1/z^{*})$. As in a single-user channel, the zero-forcing linear detector can suffer from severe noise enhancement. The output of the zero-forcing detector is given by $\boldsymbol{y}_k = \boldsymbol{x}_k + \boldsymbol{v}_k$, where $\boldsymbol{v}_k$ is a colored noise vector with PSD $N_0\mathbf{H}(z)^{\dagger}$. In theory, the mean-squared error (MSE) of the zero-forcing linear detector will be nearly infinite if $det(\mathbf{H}(z))$ has a zero close to the unit circle.

In contrast, the minimum mean-squared error (MMSE) linear detector seeks to minimize the MSE between the detector output $\boldsymbol{y}_k$ and the channel input vector $\boldsymbol{x}_k$; in other words, it attempts to minimize the total MSE = $E[\|\boldsymbol{y}_k - \boldsymbol{x}_k\|^2]$. Let $\text{MSE}_i = \text{E}[|y_k^{(i)} - x_k^{(i)}|^2]$ denote the MSE for the *i*-th user. Since the total MSE is the sum of the MSEs for each user, *i.e.*, $E[\|\boldsymbol{y}_k - \boldsymbol{x}_k\|^2] = \sum_{i=i}^{n} \text{E}[|y_k^{(i)} - x_k^{(i)}|^2]$, the MMSE linear detector minimizes the MSE for each individual user. This detector is desirable because it provides a reasonable balance between the suppression of interference and the enhancement of noise. The transfer function of the MMSE linear detector can be expressed in two equivalent ways [61]:

$$\mathbf{C}_{MMSE}(z) = \mathbf{H}^{*}(1/z^{*})[\mathbf{H}(z)\mathbf{H}^{*}(1/z^{*}) + \sigma^2\mathbf{I}_m]^{-1}, \tag{2-10}$$

$$= [\mathbf{H}^{*}(1/z^{*})\mathbf{H}(z) + \sigma^2\mathbf{I}_n]^{-1}\mathbf{H}^{*}(1/z^{*}). \tag{2-11}$$

We should emphasize that both (2-10) and (2-11) produce the same result, except when the noise is zero ($\sigma^2 = 0$), and when the channel is tall ($m > n$), in which case (2-10) is not valid. We observe that when $\sigma^2 = 0$, (2-11) reduces to a zero-forcing linear detector. We also observe that the MMSE linear detector would exist even when $det(\mathbf{H}(z))$ has zeros on the unit circle.

2.2.3 Adaptive Detection

So far we have focused on unconstrained receivers with full knowledge of the channel and of the signal and noise characteristics. However, in practice these parameters will be unknown and possibly time-varying, and so adaptive detection techniques are required. In this section, we present the vector least-mean squared (LMS) algorithm [19,20], which is a straightforward extension of the single-user LMS algorithm.

Consider the adaptive multiuser linear detector shown in Fig. 2-3. We assume that the linear detector has a transfer function of $\mathbf{C}(z) = \mathbf{C}_0 + \mathbf{C}_1 z^{-1} + \ldots + \mathbf{C}_N z^{-N}$, where N is the memory of the detector. The detector output can be expressed as follows:

$$\boldsymbol{y}_k = \mathbf{C}\mathbf{R}_k, \tag{2-12}$$

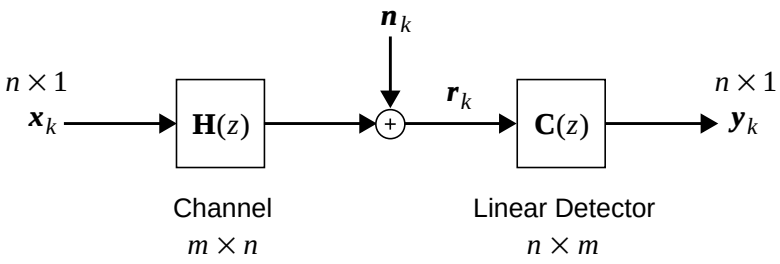


Fig. 2-3. A block diagram of an $m \times n$ channel with followed by an $n \times m$ linear detector with memory.

where $\mathbf{C} = [\mathbf{C}_0\ \mathbf{C}_1\ \ldots\ \mathbf{C}_N]$ is an $n \times (N+1)m$ matrix composed of the multiuser detector coefficients and $\mathbf{R}_k{}^T = [\boldsymbol{r}_k{}^T\ \boldsymbol{r}_{k-1}{}^T\ \ldots\ \boldsymbol{r}_{k-N}{}^T]$ is an $(N+1)m \times 1$ stacked-observation vector.

The goal of the vector LMS algorithm is to minimize the MSE cost function, defined by:

$$J = E[\|\boldsymbol{y}_k - \boldsymbol{x}_k\|^2], \tag{2-13}$$

where $\boldsymbol{x}_k$ is the channel input or the desired input signal. In the classical steepest-descent algorithm, the detector tap weights are adjusted according to the following update equation:

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \frac{\mu}{4}\nabla_{\mathbf{C}} J, \tag{2-14}$$

where $\nabla_{\mathbf{C}}$ is the complex gradient[1] of the cost function with respect to the detector tap weights and μ is the step size. The complex gradient of (2-13) with respect to the linear detector $\mathbf{C}$ is given by:

$$\nabla_{\mathbf{C}} J = 4E[\boldsymbol{e}_k \mathbf{R}_k{}^*], \tag{2-15}$$

where $\boldsymbol{e}_k = \boldsymbol{y}_k - \boldsymbol{x}_k$ is the error signal between the detector output and the channel input.

Substituting $4\boldsymbol{e}_k\mathbf{R}_k{}^*$ as a stochastic approximation for the gradient in the steepest-descent algorithm, we arrive at the following update equation for the linear detector:

---

1. The complex gradient of J_v with respect to $\mathbf{C}$ is defined as follows: $\nabla_{\mathbf{C}} J_v = \nabla_{\mathbf{C}_R} J_v + j\nabla_{\mathbf{C}_I} J_v$, where $\mathbf{C}_R = Re(\mathbf{C})$ and $\mathbf{C}_I = Im(\mathbf{C})$ [1].

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \mu \boldsymbol{e}_k \mathbf{R}_k^*. \tag{2-16}$$

We refer to this algorithm as the vector LMS algorithm. Observe that this algorithm reduces to the familiar single-user LMS algorithm when $n = 1$. We should point out that the vector LMS is *not* equivalent to n independent single-user LMS algorithms operating in parallel. For example, when the input signals are highly cross-correlated, as in the case of multiuser interference, the vector LMS will outperform a bank of n independent single-user LMS algorithms. Finally, it is important to emphasize that the steady-state performance of the vector LMS algorithm approaches that of the MMSE linear detector, if the step size of the algorithm is appropriately chosen.

2.3 BLIND DETECTION

In certain applications, the channel response is unknown and the receiver does not have access to training sequence. In these situations, the receiver must adapt according to a blind detection algorithm [25]. In this section, we review both the single-user and multiuser blind detection strategies.

2.3.1 Blind Single-User Detection

Over the last quarter century, many different blind equalization algorithms have been developed. These algorithms can be classified into three distinct groups: nonlinear algorithms, algorithms that are based on higher-order statistics (HOS), and those that are based on steepest-descent techniques. Nonlinear blind equalization algorithms include maximum-likelihood estimation techniques [62,63], decision-feedback equalizers [64,65], and neural networks [66,67]. These algorithms are typically used when the channel distortion

is too severe for linear equalizers [68]. The maximum-likelihood approach, neural network, and decision feedback methods are not very popular in practical applications because they are either too complex or have convergence problems.

The second class of algorithms, those based on higher-order statistics, include polyspectra algorithms [69-71] and algorithms that explicitly use fourth-order cumulants [72-74]. These algorithms generate higher-order cumulants by applying a nonlinear transformation to the equalizer inputs; they generally have faster convergence rates and are guaranteed to converge globally. However, their improved performance comes at the expense of higher computational complexity.

The last class of algorithms are those that are based on steepest-descent techniques. They are the most popular blind algorithms to date and hence, we will describe them in somewhat greater detail. These algorithms are typically implemented using the least-mean-square (LMS) adaptation [75,76], where the “desired response” at each iteration is generated by applying a memoryless nonlinearity $g(\cdot)$ to the output of linear filter. The basic difference between these algorithms lies in the choice of the memoryless nonlinearity. Although the algorithms are different in this respect, they can all be classified as Bussgang[2] algorithms [78,79].

The most well-known Bussgang algorithm is the decision-directed equalizer. This equalizer is not suited for blind equalization because, in general, the initial decisions are unreliable and therefore convergence can never be guaranteed. In 1975, Sato introduced the first blind equalization algorithm for the recovery of pulse-amplitude modulated

2. Since the equalizer output y_k satisfies $E[y_k g(y_k)^*] = E[y_k y_k^*]$ in equilibrium, it is referred to as a Bussgang process [77].

(PAM) signals [80]. He proposed to the generation of the error signal for LMS by using the nonlinearity $g(y_k) = \gamma\,\mathrm{sgn}(y_k)$, where the gain $\gamma = E[x_k^2] / E[|x_k|]$ is a function of the channel input x_k. Sato later extended this algorithm to two-dimensional PAM signals (quadrature-amplitude modulated signals) [81]. Both of these algorithms, unfortunately, have slow convergence because the error signal $e_k = g(y_k) - y_k$ is very noisy around the desired solution. In an attempt to speed up convergence, Benveniste and Goursat proposed using a combination of the Sato and decision-directed errors [82,83]. Their algorithm relies on the more dependable Sato errors before the eye diagram opens and the more dependable decision-directed errors thereafter. Similarly, Picchi and Prati [84] proposed a technique that updates only when the error is deemed reliable, *i.e.*, when the Sato and decision-directed errors have the same sign. Since this algorithm may not update weights at each iteration, it was referred to as the "Stop-and-Go" algorithm.

Realizing that equalizing a channel up to a phase offset is equivalent to forcing the magnitude of the ISI to zero, Godard suggested a cost function that characterizes the ISI present at the equalizer output, independent of the carrier phase [85]. This cost function is given by:

$$J(p, q) = E[(|y_k|^p - |x_k|^p)^q], \tag{2-17}$$

where p and q are positive integers. Since, in practice, the input sequence x_k is unknown, Godard suggested using a dispersion function instead of (2-17):

$$D(p, q) = E[(|y_k|^p - R_p)^2], \tag{2-18}$$

where $R_p = E[|x_k|^{2p}]/E[|x_k|^p]$ for some positive integer p. Treichler and Agee independently proposed the constant-modulus algorithm (CMA) cost function, which is a special case of Godard's cost function for $p = q = 2$ [26]. The CMA was originally designed to restore the constant-modulus property of phase-shift keyed (PSK) constellation at the equalizer output. Godard showed that for $p = q = 2$ and input alphabets that satisfy $E[|x_k|^4] < 2E[|x_k|^2]$, the global minima of (2-18) corresponds to the case of zero ISI [85]. For infinite-length equalizers, Foschini showed that (2-18) has many unstable saddle points but only one global minimum [86].

Example 2-1. The constant-modulus algorithm cost function is defined as follows:

$$J = E[(|y_k|^2 - M)^2], \tag{2-19}$$

where the modulus $M = \mathrm{E}[|x_k|^4] / \mathrm{E}[|x_k|^2]$.

Shalvi and Weinstein [87] introduced a blind equalization scheme based on matching the kurtosis of the channel input and the equalizer output. The kurtosis of a complex input sequence is defined as:

$$K(x) = E[|x|^4] - 2E^2[|x|^2] - |E[x^2]|^2. \tag{2-20}$$

These authors showed that if $E[|x_k|^2] = E[|y_k|^2]$, then $|K(y_k)| \leq |K(x_k)|$ with equality only when perfect equalization has been achieved. Thus, they suggested an algorithm that

maximizes $|K(y_k)|$ subject to the constraint $E[|x_k|^2] = E[|y_k|^2]$. We should point out that a special case of the Shalvi-Weinstein algorithm is the original Godard algorithm.

Many of the Bussgang algorithms are based on minimizing a non-convex cost function and so, global convergence cannot always be guaranteed [88-91]. Even the algorithms that are guaranteed to converge for infinite-length equalizers suffer from ill-convergence for finite-length equalizers [89,90]. To deal with ill-convergence, Godard [85] suggested a tap-initialization procedure, while Foschini [86] suggested an algorithm to track and center the primary tap. Moreover, the convergence for the Bussgang algorithms is often slow because the convergence rate of the underlying adaptation scheme (LMS algorithm) is dependent on the eigenvalue spread of the channel. To speed up convergence, Nikias and colleagues [92,93] proposed the CRIMNO (criterion with memory non-linearity) algorithm, which uses additional nonlinearity to exploit the correlation between symbols. Another drawback of Bussgang algorithms is that they are invariant to an arbitrary rotation. Typically, an estimate of this rotation can be generated by using a decision-directed phase-locked loop (PLL). In the next section, we describe the basic structure of a PLL.

2.3.2 Phase-Locked Loop

The basic structure of a decision-directed (DD) phase-locked loop consists of three major components [1-3,27]: a *phase detector,* a *loop filter,* and a *complex voltage-controlled oscillator* (VCO), as illustrated in Fig. 2-4. The phase detector compares the phase of the received signal z_k with the phase of the decision $\hat{x}_k$. The output of the phase detector is a measure of the phase error between the two input signals. The estimate of the phase error is then filtered by a loop filter to create a control signal which drives the VCO.

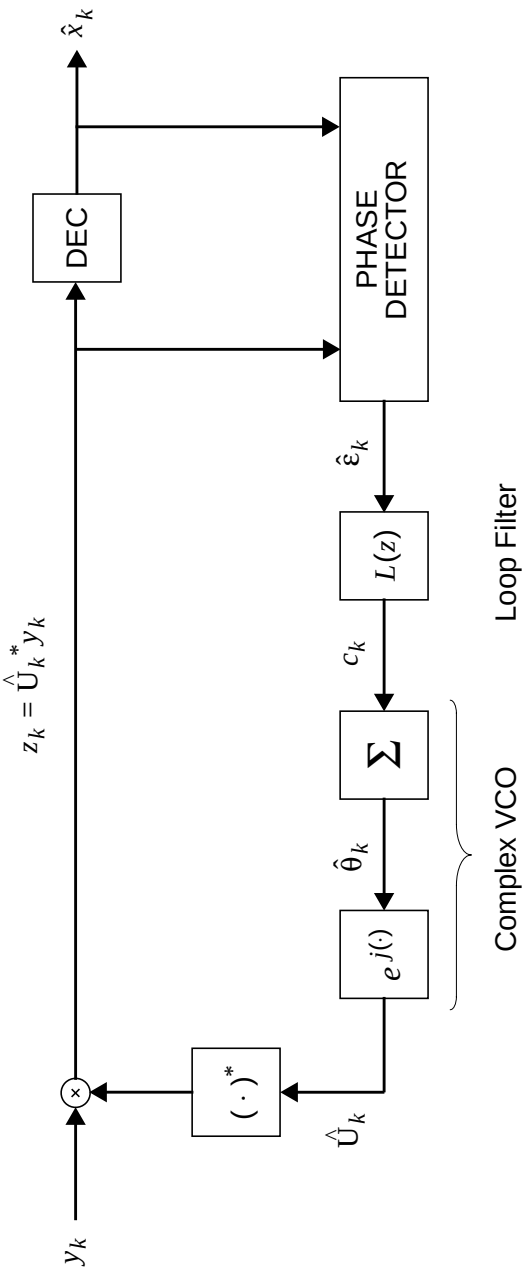


Fig. 2-4. Block diagram of the basic structure of a decision-directed phase-locked loop. The main components for phase-locked loop are the phase detector, loop filter, and complex VCO.

The control signal changes the phase of the VCO output in a direction that reduces the phase error between z_k and $\hat{x}_k$.

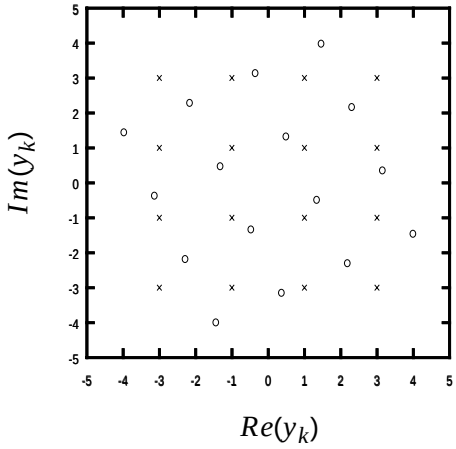
Suppose that the input signal y_k to the PLL has the following form:

$$y_k = exp(j\theta_k)x_k + n_k, \tag{2-21}$$

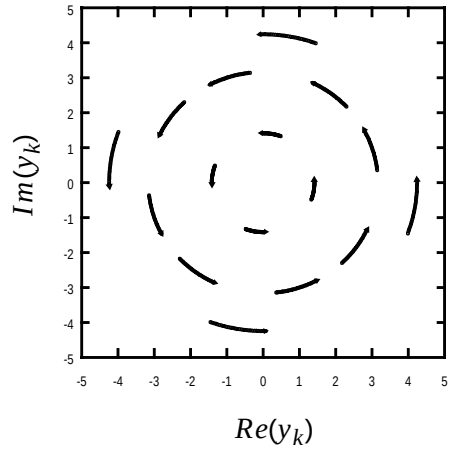
where x_k is the channel input, θ_k is the phase offset, and n_k represents the noise. Two of the most common types of offsets include the constant phase offset $\theta_k = \theta \in [0, 2\pi)$, and the constant frequency offset $\theta_k = \omega k + \theta$, where ω is the angular frequency offset, and $\theta \in [0, 2\pi)$. To provide a better understanding of these two types of offsets, we look at the following examples:

Example 2-2. If the phase offset in (2-21) is constant, then the received constellation is a tilted version of the transmitted constellation in the absence of noise. An example of a tilted received constellation is shown in Fig. 2-5(a). The tilt in the received constellation poses two problems: first, it can result in the symbols crossing decision boundaries leading to incorrect decisions; and second, it can degrade the immunity of the receiver to noise by bringing the received symbols closer to the decision boundaries.

Example 2-3. For a constant frequency offset, the received constellation is a rotating version of the transmitted constellation in the absence of noise. An example of a rotating received constellation is shown in Fig. 2-5(b). This



(a)



(b)

Fig. 2-5. If the ambiguity in the input signal to the PLL is a constant phase offset, then the received constellation will be a tilted version of the transmitted constellation as shown in (a). The o's represent the transmitted 16-QAM constellation and the x's represent the received constellation. If the ambiguity is a constant frequency offset, the received constellation will spin, as illustrated in (b).

rotation results in severe degradation of the transmitted signal at the receiver when the received symbols cross the decision boundaries.

The output of the complex VCO can be defined as follows:

$$\hat{U}_k = exp(j\hat{\theta}_k), \tag{2-22}$$

where $\hat{\theta}_k$ is an estimate of the phase offset. If the estimate of the phase offset is correct ($\hat{\theta}_k \approx \theta_k$), then the PLL is said to be in *phase-lock*, and the received signal $z_k = \hat{U}_k^* y_k$ reduces to:

$$z_k = exp(j\varepsilon_k)x_k + exp(-j\hat{\theta}_k)n_k, \tag{2-23}$$

$$\approx x_k + w_k, \tag{2-24}$$

where $\varepsilon_k = \theta - \hat{\theta}_k$ is the actual phase error and $w_k = exp(-j\hat{\theta}_k)n_k$ is a rotated noise term. At high SNR, (2-24) reduces to $z_k \approx x_k$.

As shown in Fig. 2-4, the first step in generating $\hat{\theta}_k$ is to estimate the phase error ε_k between $\hat{x}_k$ and z_k [1]:

$$\hat{\varepsilon}_k = sin^{-1}\left[\frac{Im(\hat{x}_k^* z_k)}{|\hat{x}_k||z_k|}\right]. \tag{2-25}$$

We should point out that the estimate of the phase error is only an approximation because of occasional decision errors and noise. Since this phase detector is decision-directed, it

suffers from a phase ambiguity when the transmitted constellation is $\frac{\pi}{M}$-symmetric, where M is a positive integer. For example, if the phase error is an integral multiple of $\frac{\pi}{M}$, then the constellation for z_k has the same appearance as the transmitted constellation and therefore, the output of the phase detector is identically zero. Thus, the decision-directed phase detector given by (2-25) cannot properly estimate this type of phase ambiguity. Typically, the problem is resolved at a higher layer using differential encoding.

The phase error estimate $\hat{\varepsilon}_k$ is then passed through a loop filter $L(z)$ to produce the control signal c_k. The loop filter serves two purposes: it generates the necessary control signal for the VCO, and it filters both the noisy and the incorrect estimates produced by the phase detector [27]. The output of the loop filter c_k drives a sum-accumulator to produce an estimate of the phase offset:

$$\hat{\theta}_k = \sum_{i=0}^{k-1} c_i. \tag{2-26}$$

This equation can also be written in terms of a recursive update:

$$\hat{\theta}_{k+1} = \hat{\theta}_k + c_k, \tag{2-27}$$

where $\hat{\theta}_0 = 0$ is the initial condition. Finally, the VCO output $\hat{U}_k$ is generated by passing $\hat{\theta}_k$ through a complex exponentiator. The cascade of the sum-accumulator and the complex exponentiator is often referred as the complex VCO.

A first-order PLL is sufficient to track a constant phase-offset, but a second-order PLL is necessary to track a constant frequency offset. In practice, a second-order PLL is typi-

cally used to resolve the residual phase error that remains after the Bussgang algorithms. A thorough treatment of a first-order and second-order PLL is given in Chapter 3.

2.3.3 Blind Multiuser Detection

Many of the early blind multiuser detectors were based explicitly on higher-order statistics. Cardoso [94] and Comon [95] independently recognized that the second-order statistics of the channel output are not sufficient to blindly recover a memoryless channel. Clearly, **H** and **HU** produce the same covariance matrix at their output for any unitary matrix **U**. Hence, second-order statistical information can only provide an estimate of the channel up to an arbitrary unitary matrix. Higher-order statistics are needed to resolve the remaining unitary ambiguity.

Cardoso proposed to estimate a memoryless channel by first whitening the channel output and then estimating the remaining unitary ambiguity by diagonalizing a cumulant matrix. Comon, on the other hand, proposed to estimate a memoryless channel by maximizing a contrast function. He showed that if the channel inputs are non-Gaussian and statistically independent, the transmitted vector is recovered if and only if the components of the detector output are also statistically independent. He also introduced the idea of independent component analysis (ICA) of a random vector, which consists of searching for a linear transformation that minimizes the statistical dependence between its components. In order to implement the linear transformation of ICA, Comon suggested the use of contrast functions, which were originally introduced by Donoho [96]. Giannakis *et al*. [97] showed that the parameters of a moving average (MA) multiuser channel model can be uniquely identified from third-order cumulants. Later Swami *et al*. [98] extended Giannakis' algorithm for an auto-regressive moving average (ARMA) channel model. Tugnait [99] later

proposed an algorithm for channel identification based on both second and fourth-order cumulants. Recently Comon [36] has extended his idea of contrast functions to channels with memory. Moreau and Pesquet [37] have also proposed an algorithm for identifying a channel based on generalized contrast functions. Algorithms based on higher-order statistics are impractical for many real-world applications because of their high computational complexity.

Recently, a new class of algorithm based on linear prediction (LP) has been introduced only for tall channels [33-35,38-46,54]. Tong, Xu, and Kailath [38] showed that a memoryless single-input, multiple-output (SIMO) channel can be identified up to an arbitrary scalar rotation $e^{j\theta}$ using only SOS. Slock [39-41] showed that the output of an FIR SIMO channel has not only a moving-average (MA) representation, but also a finitely parameterized autoregressive (AR) representation. Gorokhov, Loubaton, and Moulines [34] extended the work of Tong *et al*. and Slock to multiuser channels. They showed that a tall multiuser channel can be identified up to an arbitrary unitary matrix $\mathbf{U}$ using only SOS and that the channel simultaneously has both an MA and AR representation. Identification of the residual unitary ambiguity necessarily requires HOS. The majority of these algorithms are batch-oriented and have high computational complexity. Recently, an adaptive multiuser linear predictor has been proposed that significantly reduces the computational complexity to the order of the vector LMS [46,54]. Some of the drawbacks of LP-based algorithms are: they can lead to significant noise enhancement, especially when the leading taps of the channel are small; they require knowledge of the channel memory; they only work well when the tap of interest contains most of the channel energy; and they can be computationally complex when m is much larger than n.

Another class of algorithms is based upon generalizing CMA, which has complexity on the order of the LMS algorithm, to multiple dimensions [20,100-102]. A common extension of CMA to multiuser systems is to impose the constant-modulus constraint on each component of the multiuser detector output [103]:

$$J_p = \sum_{i=1}^{n} E[(|y_k^{(i)}|^2 - M_i)^2], \tag{2-28}$$

where $M_i = E[|x_k^{(i)}|^4]/E[|x_k^{(i)}|^2]$ is the modulus for the i-th user, and $x_k^{(i)}$ and $y_k^{(i)}$ denote the i-th component of the channel input $\boldsymbol{x}_k$ and the detector output $\boldsymbol{y}_k$, respectively. We refer to this cost function as the *pointwise CMA* cost function.

The local minima for this cost function are defined by the following theorem:

Theorem 2-4. Assuming an infinite-length multiuser linear detector, the global minima of (2-28), in absence of noise, have the form [20,53]:

$$\mathbf{F}(z) = \begin{bmatrix} 0 & \cdots & 0 & e^{j\theta_1} z^{-N_1} & 0 & \cdots & 0 \\ 0 & & \cdots & 0 & \cdots & 0 & e^{j\theta_2} z^{-N_2} \\ \vdots & & & \vdots & & \vdots & \\ e^{j\theta_n} z^{-N_n} & 0 & \cdots & 0 & \cdots & & 0 \end{bmatrix}, \tag{2-29}$$

for any integers N_1, N_2, …, N_n, and for any angles θ_1, θ_2, …, θ_n, and where the i-th row of $\mathbf{F}(z)$ has only one nonzero term of the form $e^{j\theta_i} z^{N_i}$, which can be located in any column.

This theorem implies that the pointwise CMA cost function is minimized if and only if the i-th component of the multiuser detector output is equal to a possibly rotated and delayed version of the j_i-th input, $y_k^{(i)} = e^{j\theta_i} x_{k-N_i}^{(j_i)}$ where $j_i \in \{1, 2, \ldots, n\}$. The arbitrary rotation for each user is not troublesome because it can be resolved using a bank of n independent single-user decision-directed phase-locked loops. The arbitrary delays and permutation of the users can be resolved at some higher layer of processing.

The pointwise CMA cost function has both desirable and undesirable local minima. The desirable local minima occur when all of the transmitted signals have been recovered up to a possible delay and rotation for each user. Examples of desirable local minima include:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & j \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & z^{-2} \end{bmatrix}, \begin{bmatrix} 0 & z^{2} \\ -1 & 0 \end{bmatrix}, \text{ and } \begin{bmatrix} 0 & z^{-1} \\ e^{j\pi/3} & 0 \end{bmatrix}. \tag{2-30}$$

In each case, information from both users has been recovered. We observe that all of desirable local minima are in fact unitary matrices.

The undesirable local minima occur when information from one or more users is lost. Examples of undesirable local minima include:

$$\begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ z^{-1} & 0 \end{bmatrix}, \begin{bmatrix} 0 & z^{-1} \\ 0 & z^{-2} \end{bmatrix}, \begin{bmatrix} -1 & 0 \\ j & 0 \end{bmatrix}, \text{ and } \begin{bmatrix} 0 & z^{-1} \\ 0 & e^{j\pi/2} z \end{bmatrix}. \tag{2-31}$$

In each case, information for either user 1 or user 2 is completely lost. We observe that, for the undesirable local minima, a subset of the detector outputs are correlated. Several extensions of pointwise CMA cost function have been proposed to eliminate these undesirable pointwise minima. One extension is based on adding a term to the pointwise CMA cost function that penalizes any correlations among detector output components [100,101]:

$$J_d = A\ J_p + B \sum_{n\,=\,-\delta}^{\delta} \sum_{i \neq j} |\ \mathrm{E}[y_k^{(i)}(y_{k-n}^{(j)})^*]\ |^2, \tag{2-32}$$

where A and B are positive constants and δ is some positive integer. We refer to this cost function as the *decorrelation CMA* cost function. A drawback of this cost function is that it requires estimates of the cross-correlations of $\boldsymbol{y}_k$ at different delays, which can be difficult to obtain because $\boldsymbol{y}_k$ is a non-stationary signal. Also, if δ is less than the sum of the memory in the channel and the detector, the additional term does not eliminate all of the undesirable solutions, it only eliminates the undesirable pointwise minima that have delays within the range $[z^{\delta}, \ldots, z^{-\delta}]$.

Another extension of the pointwise CMA cost function was introduced by Oda and Sato [102]. They suggested that the additional term should be a cost function that penalizes the variations in the norm of the multiuser detector output relative to a reference modulus:

$$J_c = A\ J_p + B\ \mathrm{E}[(\|\ \boldsymbol{y}_k\ \|^2\ - M)^2], \tag{2-33}$$

where A and B are positive constants and M is the reference modulus. We refer to this cost function as the *combination CMA* cost function. Oda and Sato reasoned that if the second term in (2-33) is minimized by a unitary matrix, that additional term would eliminate the undesirable local minima because the non-unitary matrices would produce a higher cost. Unfortunately, as we will show in Chapters 4 and 5, the second term of (2-33) is only minimized by a unitary matrix when the channel is memoryless and the input alphabet is non-constant modulus. For memoryless channels with a constant-modulus input alphabet and channels with memory, the second term in (2-33) is minimized by both unitary and non-unitary matrices. Hence, the combination CMA cost function will still have undesirable local minima for these cases. For example, if the channel is memoryless and the input alphabet is BPSK, then

$$\mathbf{F} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}, \tag{2-34}$$

is a local minima of (2-33). On the other hand, if the channel has memory, then

$$\mathbf{F}(z) = \begin{bmatrix} 1 & 0 \\ z^{-1} & 0 \end{bmatrix}, \tag{2-35}$$

minimizes the combination CMA cost function.

Despite the modifications to the pointwise CMA cost function, both decorrelation CMA and combination CMA can still converge to an undesirable local minima. The discussion on CMA-based detectors provides a foundation for the development of a new

blind multiuser detector based on a unique generalization of the constant-modulus algorithm to be presented in Chapters 4 and 5.

CHAPTER 3

MULTIDIMENSIONAL PHASE-LOCKED LOOP

Even though a unitary matrix almost never correctly models a real-world multiuser channel, it often appears in the blind multiuser detection process. For example, when the detection process is separated into a whitening step and a rotation step, in the absence of noise, the cascade of the whitening filter and the channel matrix reduces to a unitary matrix. Because all unitary matrices produce the same covariance matrix, second-order statistics are insufficient for resolving the unitary ambiguity. Therefore, the estimation of a unitary matrix requires higher-order statistics.

In the last ten years, a class of blind algorithms, which are referred to as blind unitary estimators, has been developed. These algorithms can estimate a unitary matrix using only the observations and information about the properties of the transmitted sequence. Cardoso and Souloumiac [47-49] proposed the first blind unitary estimator, which jointly and approximately diagonalizes a set of cumulant matrices in order to estimate the columns of the unitary matrix. This algorithm is referred to as the joint approximate diagonalization of eigenmatrices (JADE) algorithm. JADE is a batch-oriented algorithm with a relatively high computational complexity of $O(n^5)$, where n is the number of users. Recently, Car-

doso and Laheld [50] have introduced a class of blind unitary estimators, based on a serial update equation, called equivariant adaptive separation via independence (EASI) algorithms. The EASI algorithms are adaptive and have low complexity, $O(n^3)$. Both JADE and EASI are flexible blind unitary estimators because they make no assumption about the structure of the input signals, other than that they are independent and non-Gaussian.

As an alternative to JADE and EASI, we propose an adaptive blind unitary estimator that exploits the discrete nature of digital communication signals. We view the unitary matrix as a constant rotation akin to a constant phase offset in a single-user communication system. The most popular structure for estimating and resolving a constant phase offset in a single-user communication system is a decision-directed phase-locked loop (PLL) [4-3,27-32]. We generalize the structure of the PLL to multiple dimensions in this chapter and show that the resulting algorithm is able to blindly estimate a unitary ambiguity.

In Section 3.1, we review the basic structure of a first-order and second-order decision-directed phase-locked loop and present a convergence analyses for each. In Section 3.2, we derive an equivalent structure for a first-order and second-order PLL that can be used to generalize the PLL to multiple dimensions. In Section 3.3, we derive the multidimensional phase-locked loop (MPLL), and in Section 3.4, we develop an alternative model for the MPLL, which has a slightly lower computational complexity. In Sections 3.5 and 3.6, we present a convergence analysis for a first-order and second-order MPLL, respectively. Finally in Section 3.7, we compare the performance and complexity of the MPLL to other well-known blind unitary estimators, JADE and EASI.

3.1 PHASE-LOCKED LOOP

The basic structure of a decision-directed (DD) phase-locked loop was described in Section 2.3.2. In this section, we present the equations for, and analyze the convergence behavior of a first-order and second-order PLL.

3.1.1 First-Order PLL

A first-order PLL, which is sufficient to track a constant phase offset [27-32], uses a loop filter of the form [1]: $L(z) = \alpha$, where α is a small positive constant. The estimate of the phase error, given earlier in (2-25) and shown below in (3-1):

$$\hat{\varepsilon}_k = sin^{-1}\left[\frac{Im(\hat{x}_k^* z_k)}{|\hat{x}_k||z_k|}\right], \tag{3-1}$$

is passed through the loop filter $L(z)$ and a sum-accumulator to produce an estimate of the phase offset:

$$\hat{\theta}_k = \sum_{i=0}^{k-1} \alpha\hat{\varepsilon}_i. \tag{3-2}$$

We can also write (3-2) in terms of a recursive update:

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \alpha\hat{\varepsilon}_k, \tag{3-3}$$

where $\hat{\theta}_0 = 0$ is the initial condition. As before, the VCO output is given by

$$\hat{U}_k = exp(j\hat{\theta}_k). \tag{3-4}$$

In summary, (3-1), (3-3), and (3-4) define a first-order DD PLL.

3.1.2 Convergence Analysis of a First-Order PLL in the Absence of Noise

Since the PLL is decision-directed, errors in the decision will result in errors at the phase detector. Proper operation of the DD PLL depends on the decisions being correct. We need to verify that the errors at the phase detector will not prevent the DD PLL from correctly estimating the phase offset in the input signal. The following analysis closely parallels the discussion presented by Simon and Smith [104]. We assume that the channel input x_k is drawn uniformly from a discrete-input alphabet and that the phase offset is constant: $\theta_k = \theta \in [0, 2\pi)$. We also assume that the noise term is zero. The effect of noise on the convergence of the DD PLL will be considered in the next section.

By subtracting θ from both sides of (3-3) and manipulating the resulting equation, we find an equivalent update equation for the actual phase error:

$$\varepsilon_{k+1} = \varepsilon_k - \alpha\hat{\varepsilon}_k, \tag{3-5}$$

where $\hat{\varepsilon}_k$ is defined in (3-1). Since $z_k = exp(j\varepsilon_k)x_k$ in the absence of noise, $\hat{\varepsilon}_k$ is a function of both the channel input vector x_k and the actual phase error ε_k. For a very small step size α, we can assume that $\hat{\varepsilon}_k$ is independent of ε_k and therefore, we can take the expectation of (3-5) with respect to $\hat{\varepsilon}_k$. A first-order DD PLL converges in the mean when the update term in (3-5) is zero on average, *i.e.*, when the estimate of the phase error is zero on average:

$$E[\hat{\varepsilon}_k] = 0. \tag{3-6}$$

Conditioning (3-6) with respect to ε_k, we arrive at the following function[1]:

$$S(\varepsilon) = E[\hat{\varepsilon} \mid \varepsilon], \tag{3-7}$$

where the expectation is taken over the entire input alphabet. This function, which is referred to as the "S-curve" [27], is a useful tool for determining the stable points of (3-3).

> **Definition 3-1.** A stable point of a first-order DD PLL is defined to be a point ε_s on the S-curve where $S(\varepsilon_s) = 0$ and $S'(\varepsilon_s) > 0$; in other words, a stable point occurs at a point where the S-curve crosses the zero axis and has a positive slope.

We observe that if the DD PLL is near a stable point ε_s and $\varepsilon > \varepsilon_s$ ($\varepsilon < \varepsilon_s$), then $S(\varepsilon)$ is positive (negative) on average, and the correction term in (3-5) will reduce the phase error and *drive* the DD PLL towards the stable point (provided that $0 < \alpha < 1$).

Using this S-curve, we determine the stable points of a first-order DD PLL for two common input alphabets: 4-QAM and 16-QAM. Since a QAM constellation is $\frac{\pi}{2}$ symmetric, we need only consider $\varepsilon \in [-\frac{\pi}{4}, \frac{\pi}{4}]$ in (3-7). The stable points for a 4-QAM input alphabet are summarized by the following theorem:

---

1. The dependence on time has been suppressed, for the remainder of this section, to simplify notation.

Theorem 3-1. For a 4-QAM input alphabet and a noiseless input signal, the only stable point of a first-order DD PLL is given by $\varepsilon_s = 0$; in other words, a first-order DD PLL can correctly estimate any phase offset for a 4-QAM input alphabet.

Proof: For $\varepsilon \in [-\frac{\pi}{4}, \frac{\pi}{4}]$, it is easy show that (2-25) reduces to $\hat{\varepsilon} = \varepsilon$. Hence, the S-curve is the identity function: $S(\varepsilon) = \varepsilon$, and the only stable point occurs when $\varepsilon = 0$. ❑

Corollary 3-1. For a 4-QAM input alphabet, the phase error reduces by a factor of α at each iteration.

Proof: Substituting $\hat{\varepsilon}_k = \varepsilon_k$ into (3-5), we obtain:

$$\varepsilon_{k+1} = (1-\alpha)\,\varepsilon_k. \tag{3-8}$$

Since the step size $\alpha \in (0, 1)$, the phase error ε_k decreases at each iteration by a factor of α. ❑

We observe that once the DD PLL reaches the stable point of $\varepsilon_s = 0$, it will remain there indefinitely, because when $\varepsilon_k = 0$, $\hat{\varepsilon}_k = 0$ for any x_k.

Definition 3-2. The stable point of $\varepsilon_s = 0$ is a *desirable stable point*.

The stable points for a 16-QAM input alphabet are summarized by the following theorem:

Theorem 3-2. For a 16-QAM input alphabet and a noiseless input signal, the stable points of a first-order DD PLL are at $\varepsilon_s = 0, \pm 22.5^\circ$.

Proof: The S-curve for a 16-QAM constellation is shown in Fig. 3-1. This curve, normalized to have a maximum value of unity, was evaluated using (2-25). For each actual phase error $\varepsilon \in [-\pi / 4, \pi / 4]$, an arithmetic average of the 16 different estimates of the phase error (one for each symbol) was taken. This S-curve shows that the PLL has stable points at $\varepsilon_s = 0$ and at $\varepsilon_s = \pm 22.5^\circ$. The

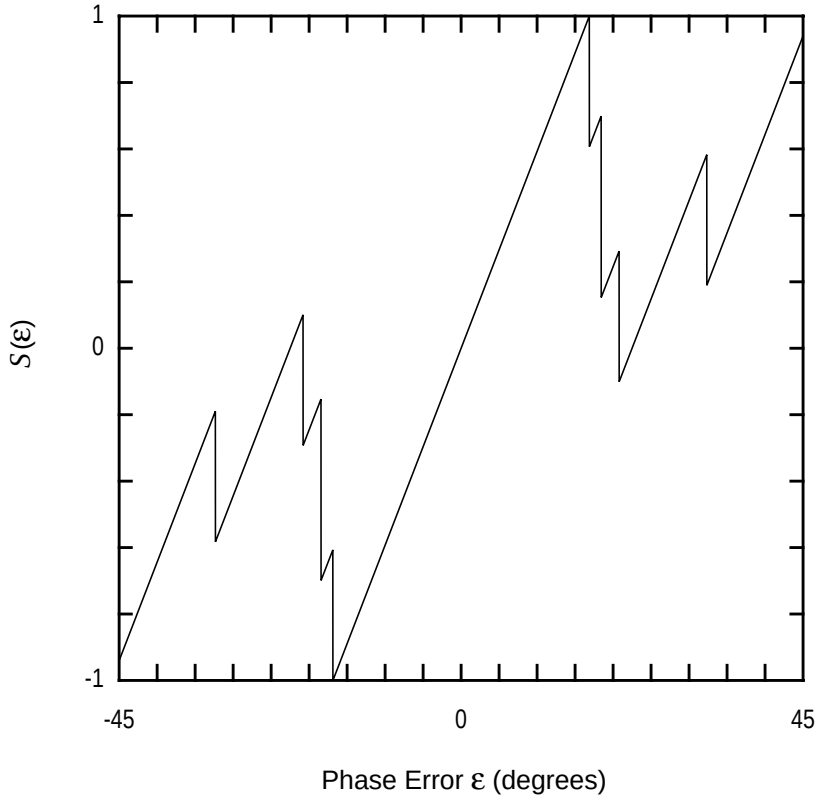


Fig. 3-1. A normalized S-curve for a first-order DD PLL with 16-QAM input alphabet.

point at $\varepsilon_s = 0$ is stable point, because $\hat{\varepsilon}_k = 0$ for any x_k when $\varepsilon_k = 0$. The points at $\varepsilon_s = \pm 22.5^\circ$ are also stable points, because the sum of estimates of the phase error for all points in the 16-QAM input alphabet, which are listed in Table 3-1, equals zero. ❑

Observe that the two stable points at $\varepsilon_s = \pm 22.5^\circ$ are undesirable because there is a non-zero residual phase error in z_k, and as a result, the channel input cannot be recovered with a simple decision device.

Definition 3-3. Any non-zero stable point is an *undesirable stable point*.

For $\varepsilon \in [0, 20.69°]$, Fig. 3-1 shows that $S(\varepsilon)$ is positive (on average) and therefore, the correction term in (3-5) will reduce the phase error and drive the PLL towards the stable point of $\varepsilon_s = 0$. This figure also indicates that it only takes a small perturbation of less than -1.81° to move the DD PLL from the undesirable stable point at $\varepsilon_s = 22.5^\circ$ to a phase error that lies in the range $[0, 20.69°]$. As indicated above, this perturbation would guarantee convergence of the DD PLL (on average) to the desired stable point. The same is also true

TABLE 3-1: Estimates of the phase error at $\varepsilon_k = 22.5^\circ$ for all points in a 16-QAM input alphabet.

x_k	$\hat{\varepsilon}_k$, when $\varepsilon_k = -22.5°$	$\hat{\varepsilon}_k$, when $\varepsilon_k = 22.5°$
$\pm 1 \pm j$	$-22.5°$	$22.5°$
$\pm(1 + 3j), \pm(3 - j)$	$3187/784°$	$-5633/392°$
$\pm(1 - 3j), \pm(3 + j)$	$5633/392°$	$-3187/784°$
$\pm 3 \pm 3j$	$3187/784°$	$-3187/784°$

for the undesirable stable point at $\varepsilon_s = -22.5^\circ$. In contrast, a perturbation of larger than 20.69° is needed to force the DD PLL away from the desired stable point of $\varepsilon_s = 0$.

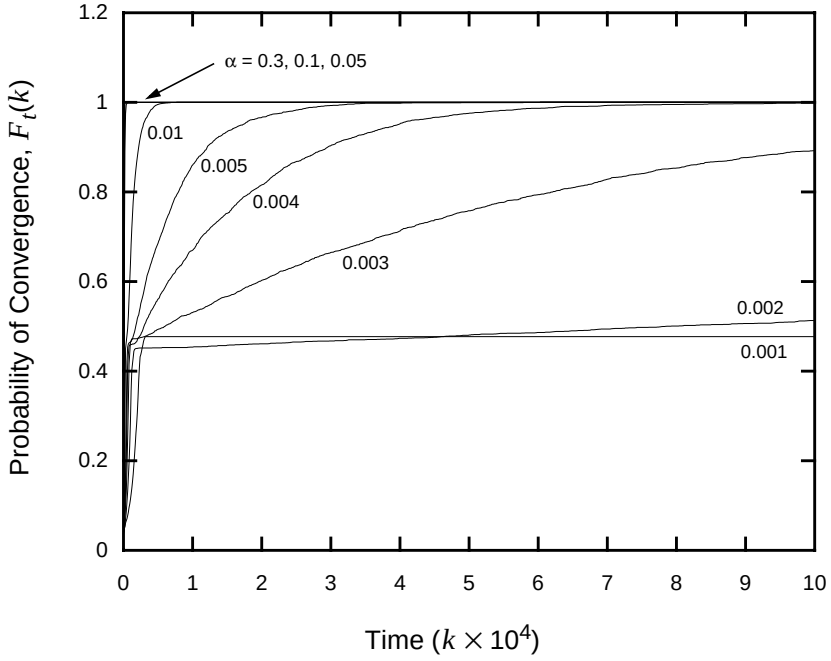
As mentioned earlier, (3-6) is only valid for very small α, and consequently, the results of Theorem 3-2 are also only valid when α is very small. Therefore, this theorem suggests that, if $\alpha \ll 1$, it is possible for a first-order DD PLL with a 16-QAM input alphabet to converge to an undesirable stable point and remain there indefinitely. To explore this possibility, we consider the following experiment which examines the effects of the step size on the convergence of a first-order DD PLL with 16-QAM input alphabet.

Experiment 3-1. Suppose that the input signal to the PLL is given by $y_k = exp(j\theta)x_k$, where θ is a randomly generated constant-phase offset and x_k is uniformly drawn from a 16-QAM input alphabet. For a given phase offset, we implemented a first-order DD PLL for 100,000 random symbols and several different step sizes $\alpha \in \{0.3, 0.1, 0.05, 0.01, 0.005, 0.004, 0.003, 0.002, 0.001\}$. For each step size, we determined the minimum number of symbols required for the DD PLL to converge to the desired stable point. Convergence to a stable point is defined to have been achieved when the transfer function $F_k = \hat{U}_k^* exp(j\theta) = exp(j\varepsilon_k)$ satisfies:

$$| F_k - P |^2 \le 10^{-3}, \tag{3-9}$$

where $P \in \{1, j, -1, -j\}$ represents the phase ambiguity associated with a QAM constellation. In all, we considered 3000 different phase offsets. In Fig. 3-2, $F_t(k)$, the fraction of trials that converged within k symbols, is

(a)



(b)

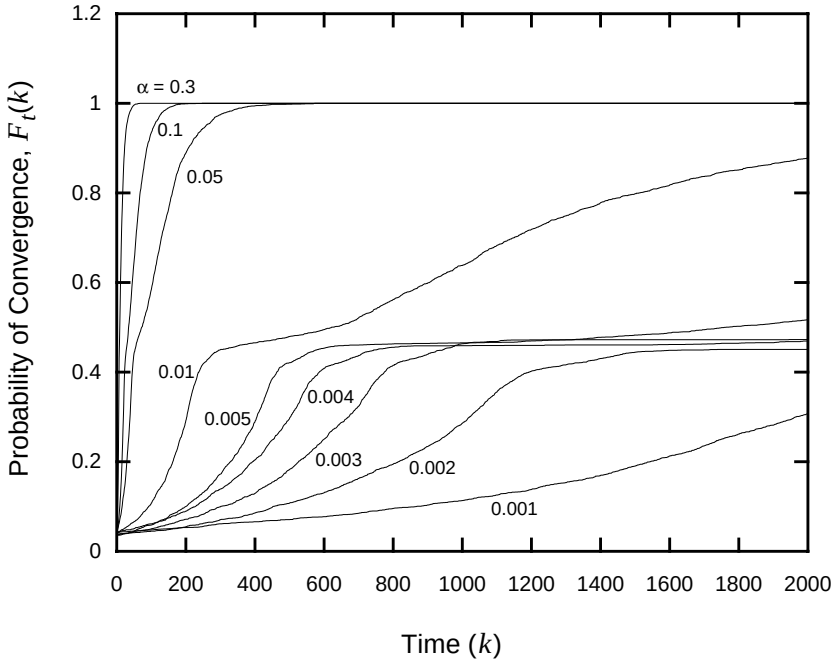


Fig. 3-2. The fraction of trials that converge within k symbols is plotted versus the number of symbols for a noiseless first-order DD PLL with 16-QAM input alphabet and various step sizes: (a) the number of symbols ranges from 0 to 100,000; (b) an expanded view of the first 2000 symbols.

plotted versus the number of the symbols. From these curves, we observe that the DD PLL always converges to the desired stable point, within 100,000 symbols, when the step size $\alpha \geq 0.004$. (In fact, if we increase the number of symbols to 500,000, then the DD PLL will always converges to the desired stable point when $\alpha \geq 0.003$.) If $\alpha \leq 0.001$, the DD PLL only converges for a small fraction of the 3000 trials; in fact, for majority of the phase offsets, the DD PLL remains trapped around an undesirable stable point even after 100,000 symbols.

This experiment confirms that it is possible for the DD PLL to converge to an undesirable stable point when the step size $\alpha \leq 0.001$. Because of the fact that we implemented the DD PLL for only 100,000 symbols, the information that this experiment consequently fails to provide is whether, for an $\alpha \leq 0.001$, the DD PLL remains at an undesirable point indefinitely. However, we do gain an insight for the selection of an appropriate step size for a real-world implementation of the DD PLL. To guarantee convergence of a first-order DD PLL with a 16-QAM input and no noise, the general rule of thumb is to choose $\alpha \geq 0.004$.

If on the other hand, one includes an infinite number of symbols, Simon and Smith have shown that a first-order DD PLL always converges to the desired stable point regardless of the step size.

Theorem 3-3. For an infinite number of symbols and no noise, a first-order DD PLL always converges to the desired stable point of $\varepsilon_s = 0$.

Proof: See Simon and Smith [104].

We observe that the result of this theorem is independent of both the input alphabet and the step size. This limiting situation considered by Simon and Smith is only of academic interest because it would require an unlimited time for the first-order DD PLL to converge; in practice, however, the speed of convergence is an important consideration.

3.1.3 Convergence Analysis of a First-Order PLL in the Presence of Noise

So far we have only considered the convergence of a first-order DD PLL in the absence of noise. In this section, we examine the effects that noise has on the convergence of a first-order DD PLL. We assume that the input signal y_k to the PLL has the following form:

$$y_k = exp(j\theta_k)x_k + n_k, \tag{3-10}$$

where n_k is a complex Gaussian random variable with variance σ^2. The corresponding received signal z_k can be expressed as:

$$z_k = exp(j\varepsilon_k)x_k + w_k, \tag{3-11}$$

where $w_k = exp(-j\hat{\theta}_k)n_k$ is also a complex Gaussian random variable with variance σ^2. Substituting (3-11) into (3-1) and expanding (3-7), we obtain an equivalent relationship for the S-curve:

$$S(\varepsilon) = \varepsilon + E\left[sin^{-1}\left(\frac{Im[\hat{x}^*(x + w)]}{|\hat{x}||x + w|}\right)\right], \tag{3-12}$$

where the expectation is taken over both the channel input and the noise term. In deriving (3-12), we have assumed that the input alphabet is QAM, and therefore, we need only consider $\varepsilon \in [-\frac{\pi}{4}, \frac{\pi}{4}]$.

Unfortunately, we are unable to simplify (3-12) any further because of the difficultly theoretically evaluating the expectation. However, we can obtain noisy S-curves by using numerical integration techniques [105]; in particular, the one-dimensional extended trapezoidal rule. The noisy stable points for a 4-QAM are summarized in the following theorem:

Theorem 3-4. For a 4-QAM input alphabet and a noisy input signal, the only stable point of a first-order DD PLL is given by $\varepsilon_s = 0$.

Proof: In Fig. 3-3, we plot the noisy S-curves for several values of SNR = $E[|x_k|^2]/\sigma^2$. Each S-curve was evaluated using (3-12). For each SNR, the S-curve only has one stable point at $\varepsilon_s = 0$.

An interesting property of the noisy S-curves is that they tend to become smoother as the SNR decreases. In fact, when the SNR = 5 dB, the noisy S-curve is completely devoid of any sharp edges.

In following experiment, we consider the noisy stable points for a 16-QAM input alphabet:

Experiment 3-2. In Fig. 3-4, we plot the noisy S-curve for several values of SNR = $E[|x_k|^2]/\sigma^2$. Each S-curve was evaluated using (3-12). For an SNR ≥ 40 dB, the S-curve does change appreciably from the one shown for infinite

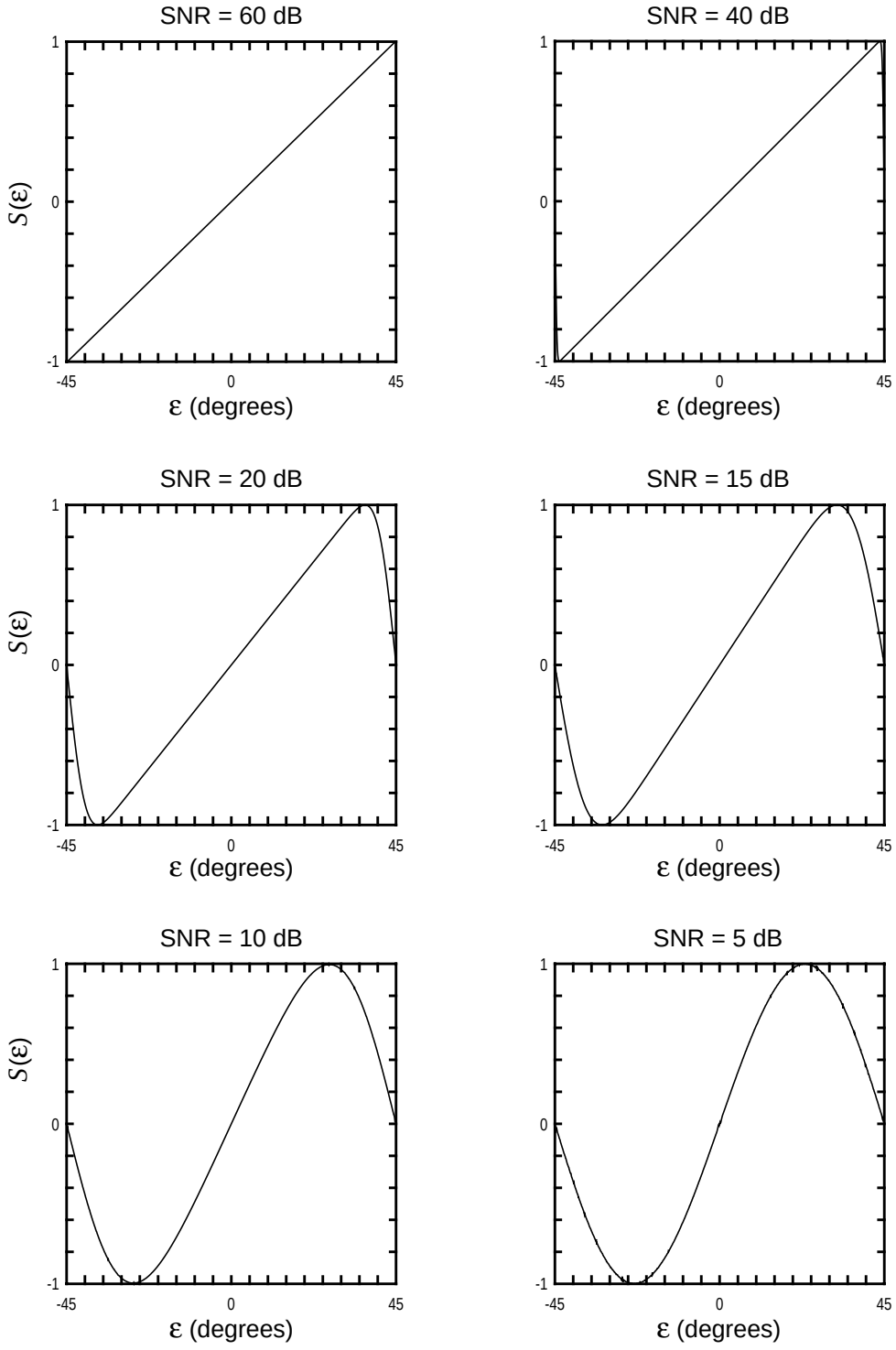


Fig. 3-3. Normalized S-curves for a first-order DD PLL with 4-QAM input alphabet and a noisy input signal.

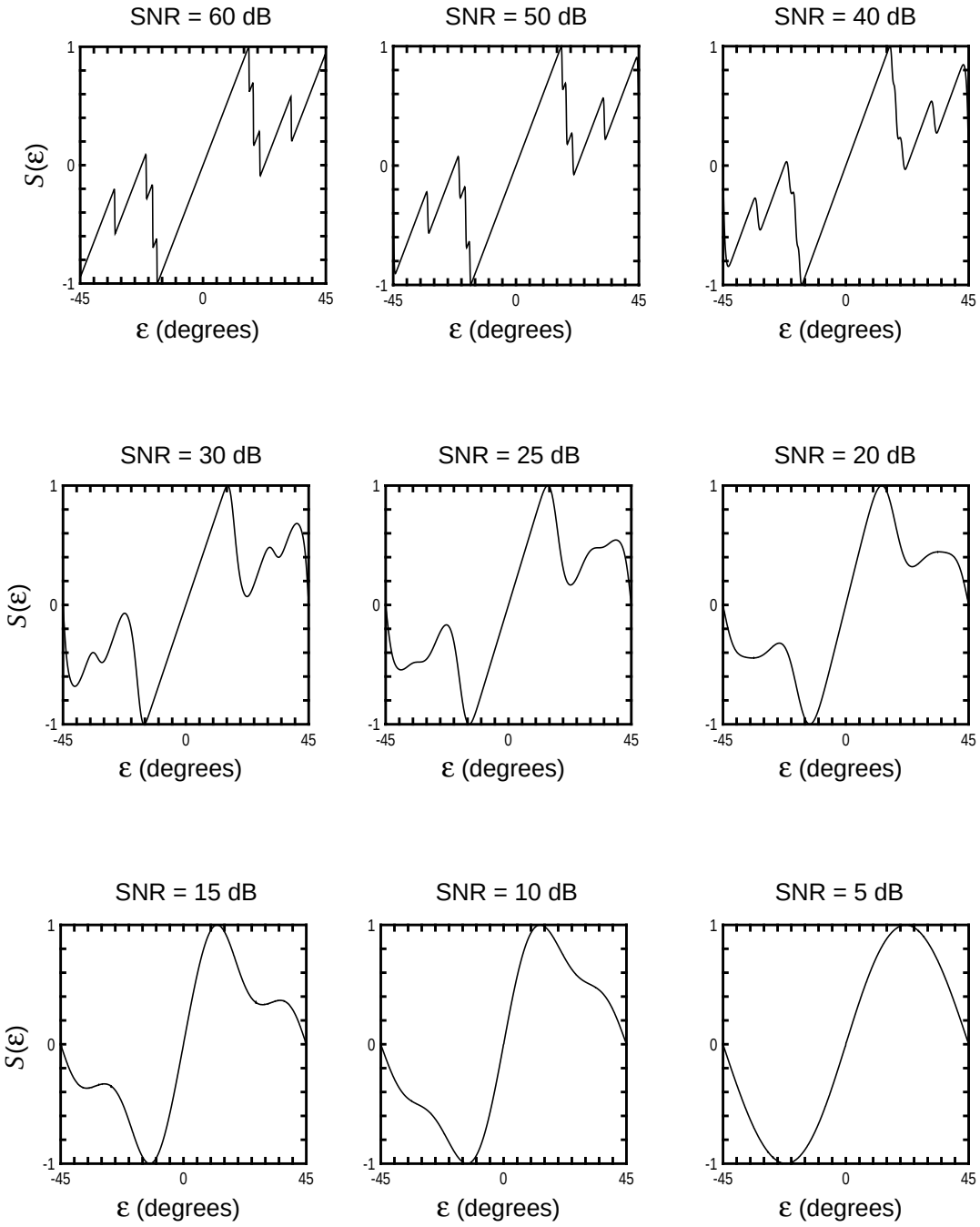


Fig. 3-4. Normalized S-curves for a first-order DD PLL with 16-QAM input alphabet and a noisy input signal.

SNR. In fact, the PLL still has a desirable stable point at $\varepsilon_s = 0$ and two undesirable stable points at $\varepsilon_s = \pm 22.5^\circ$. However, when SNR < 40 dB, the undesirable stable points disappear from the S-curves and the only remaining stable point occurs when $\varepsilon_s = 0$.

Observation 3-1. For a 16-QAM input alphabet and a noisy input signal, the stable points of a first-order DD PLL are $\varepsilon_s = 0, \pm 22.5^\circ$ when SNR ≥ 40 dB, and $\varepsilon_s = 0$ when SNR < 40 dB.

This experiment implies that a noisy signal can prevent a first-order DD PLL with a 16-QAM input alphabet from converging to an undesirable stable point, no matter what the step size. This result is due to the fact that the noise provides the necessary perturbation for the DD PLL to escape the shallow undesirable stable point. To verify this result, we consider the following experiment:

Experiment 3-3. Suppose that the noisy input signal to the PLL is given by $y_k = exp(j\theta)x_k + n_k$, where θ is a randomly generated constant-phase offset, x_k is uniformly drawn from a 16-QAM input alphabet, and n_k is a zero-mean white Gaussian noise term. We assume that SNR = 25 dB. For a given phase offset, we implemented a first-order PLL for 100,000 random symbols and several different step sizes $\alpha \in \{0.3, 0.1, 0.05, 0.01, 0.005, 0.001\}$. For each step size, we determined the minimum number of symbols required for the DD PLL to converge to the desired stable point. Convergence to a stable point is defined to have been achieved when the transfer function $F_k = \hat{U}_k^* exp(j\theta) = exp(j\varepsilon_k)$ satisfies:

$$| F_k - P |^2 \leq 10^{-3}, \tag{3-13}$$

where $P \in \{1, j, -1, -j\}$ represents the phase ambiguity associated with a QAM constellation. In all, we considered 3000 different phase offsets. In Fig. 3-5, we plot $F_t(k)$, the fraction of trials that converged within k symbols, versus time. It is observed that the PLL converges to the desired stable point, within 10,000 symbols, for all step sizes.

This experiment confirms that noise can prevent the DD PLL from converging to an undesirable stable point, even for step sizes $\alpha \leq 0.001$. This result is reassuring because all real-world applications are corrupted by noise.

3.1.4 Second-Order PLL

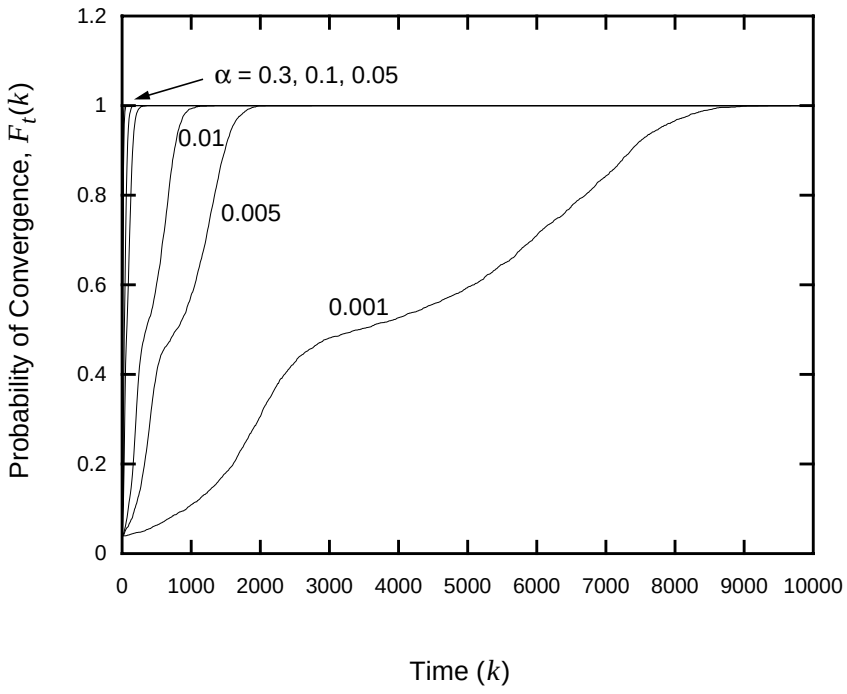
A second-order PLL, which is necessary to track a constant frequency offset: $\theta_k = \omega k + \theta$, $\theta \in [0, 2\pi)$, and $\omega \neq 0$, uses a loop filter of the form [1]:

$$L(z) = \alpha_1 + \frac{\alpha_2}{1 - z^{-1}}, \tag{3-14}$$

where both α_1 and α_2 are positive constants and $\alpha_2 \ll \alpha_1$. From (2-26), we see that for this filter, the estimate of the phase offset $\hat{\theta}_k$ is related to the estimate of the phase error $\hat{\varepsilon}_k$ by:

$$\hat{\theta}_k = \sum_{i=0}^{k-1} \left[\alpha_1 \hat{\varepsilon}_i + \sum_{l=0}^{i} \alpha_2 \hat{\varepsilon}_l \right]. \tag{3-15}$$

(a)



(b)

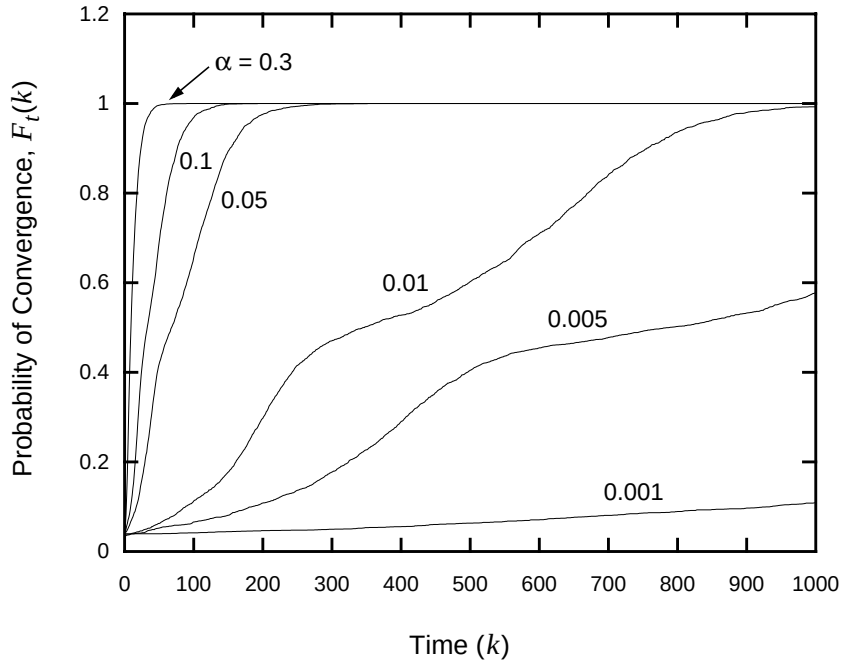


Fig. 3-5. The fraction of trials that converge within k symbols is plotted versus the number of symbols for a noisy first-order DD PLL with 16-QAM input alphabet and various step sizes: (a) the number of symbols ranges from 0 to 10,000; (b) an expanded view of the first 1000 symbols.

We observe that when $\alpha_2 = 0$, (3-15) reduces to the update equation for a first-order PLL. Equation (3-15) can also be expressed as a recursive update:

$$\phi_{k+1} = \phi_k + \alpha_2 \hat{\varepsilon}_k, \tag{3-16}$$

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \alpha_1 \hat{\varepsilon}_k + \phi_{k+1}, \tag{3-17}$$

with initial conditions $\hat{\theta}_0 = 0$ and $\phi_0 = 0$. We see that (3-16) represents the integration loop for the second-order PLL. As before, the VCO output is given by (3-4).

In summary, a second-order DD PLL is defined by (3-1), (3-4), (3-16), (3-17).

3.1.5 Convergence Analysis of a Second-Order PLL

In this section, we investigate the convergence of a second-order DD PLL in the absence of noise. We assume that the channel input x_k is drawn uniformly from a discrete-input alphabet and that the only impairment in the input signal to the PLL is a constant frequency offset: $\theta_k = \omega k + \theta$, where $\theta \in [0, 2\pi)$, and $\omega = 2\pi f$ is the angular frequency. Typically, a constant frequency offset arises because of the difference between the frequencies of the transmitter oscillator and the receiver oscillator. We should point out that a second-order loop is capable of compensating for a small frequency difference so that the steady-state phase error is zero. This result is one of the reasons why second-order loops are used extensively [32].

We are interested in determining the values of ω for which a second-order PLL would converge to the desirable stable point of zero phase error. For a continuous-time second-order PLL, they are determined by using *phase-plane techniques* [27-32]. By numerically

solving the state equations, a phase plane is constructed and the trajectories are plotted for various initial conditions. This plot shows the dynamics of the second-order loop as it settles (or fails to settle) towards a point of equilibrium.

It is not clear how the phase-plane techniques of a continuous-time second-order PLL may be extended to a discrete-time second-order DD PLL; nevertheless, we can generate a phase-plane portrait experimentally. In the following experiment, we determine the range of ω that a second-order PLL can resolve for fixed values of α_1 and α_2.

Experiment 3-4. This experiment determines the range of frequencies that a second-order DD PLL is able to resolve. Suppose that the input signal to the PLL is given by $y_k = exp(j2\pi fk + j\theta)x_k$, where f is the frequency offset, θ is the constant phase offset, and x_k is drawn uniformly from a 16-QAM input alphabet. For a given frequency offset $f = \frac{\varphi}{360}$, where φ is an integer in the range $[-25, 25]$, and a given phase offset $\theta \in [-\frac{\pi}{4}, \frac{\pi}{4}]$, we implemented a second-order DD PLL with parameters $\alpha_1 = 0.1$ and $\alpha_2 = 10^{-3}$ for 1,000,000 symbols and determined whether or not the PLL converged to the desired stable point. Convergence to a stable point is defined to have been achieved when the transfer function $F_k = \hat{U}_k^* exp(j\theta) = exp(j\varepsilon_k)$ satisfies the following relationship for 15 consecutive symbols:

$$| F_k - P |^2 \leq 10^{-3}, \tag{3-18}$$

where $P \in \{1, j, -1, -j\}$ represents the phase ambiguity associated with a QAM constellation. The shaded region in the plot of frequency offset versus

phase offset, displayed in Fig. 3-6, represents the values for which the second-order DD PLL converged. This type of plot is referred to as a phase-plane portrait. This plot shows that, with parameters $\alpha_1 = 0.1$ and $\alpha_2 = 10^{-3}$, a second-order DD PLL always converges to the desired stable point if $|f| \leq \frac{19}{360}$, regardless of the phase offset, and it can therefore, resolve a frequency offset of up to 19 degrees per baud.

This experiment shows, for $\alpha_1 = 0.1$ and $\alpha_2 = 10^{-3}$, that it is possible for a second-order DD PLL to resolve a constant frequency offset. In general, the range of frequency offsets that a second-order DD PLL can resolve will depend upon the choice of α_1 and α_2. As

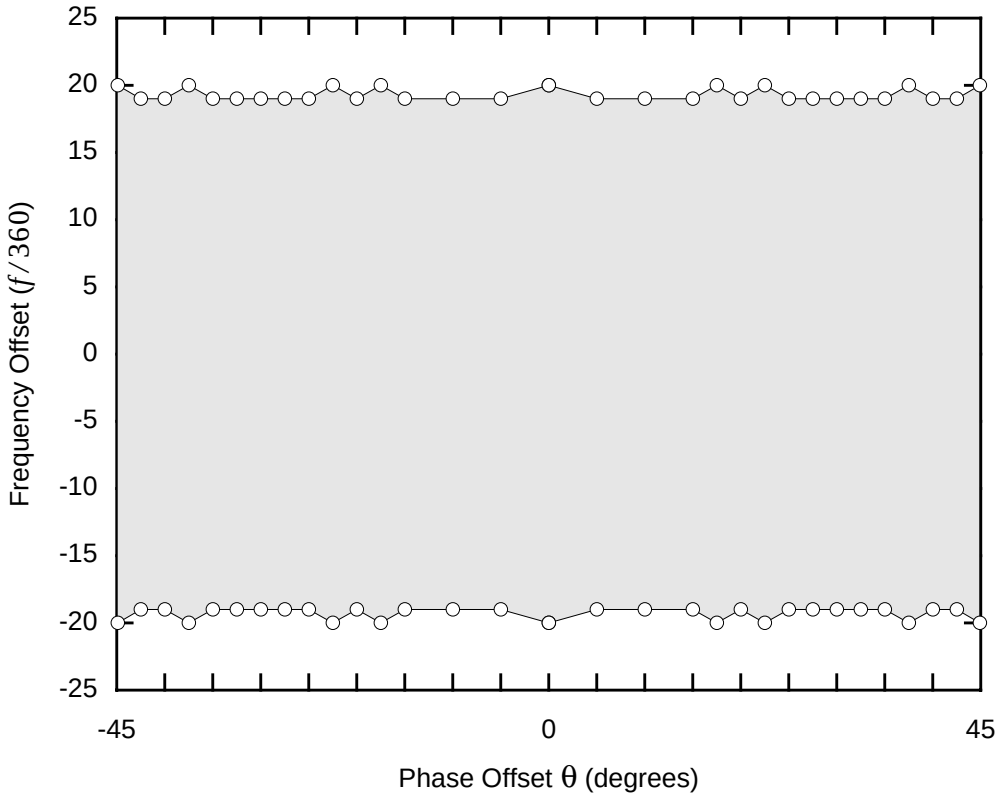


Fig. 3-6. A phase-plane portrait for a second-order DD PLL with parameters $\alpha_1 = 0.1$ and $\alpha_2 = 10^{-3}$.

seen earlier, with a first-order PLL, a noisy input signal should assist in the convergence of a second-order PLL.

3.2 Alternative Model for Phase-Locked Loop

In higher dimensions ($n \geq 2$), a rotation is completely described by a unitary matrix. A useful property of unitary matrices is that the product of two unitary matrices is also a unitary matrix. Therefore, a unitary matrix can easily be updated using a multiplicative update equation. Unfortunately, the update equations for both the first-order and the second-order PLL are additive and therefore do not easily extend to multiple dimensions. However, the structure of the PLL can be rearranged so that the update equations become multiplicative.

3.2.1 Alternative-Model First-Order PLL

From (3-2) and (3-4), the VCO output $\hat{U}_k$ is given by:

$$\hat{U}_k = exp\left[\sum_{i=0}^{k-1} \alpha \hat{\varepsilon}_i\right]. \tag{3-19}$$

We can exploit some of the properties of exponential function to rewrite (3-19) as follows:

$$\hat{U}_k = \prod_{i=0}^{k-1} \left[exp(j\hat{\varepsilon}_i)^{\alpha}\right]. \tag{3-20}$$

The last two equations imply that we can view a first-order PLL in two distinct and different ways (see Fig. 3-7): in (3-19), it is seen as the cascade of a phase detector, a loop

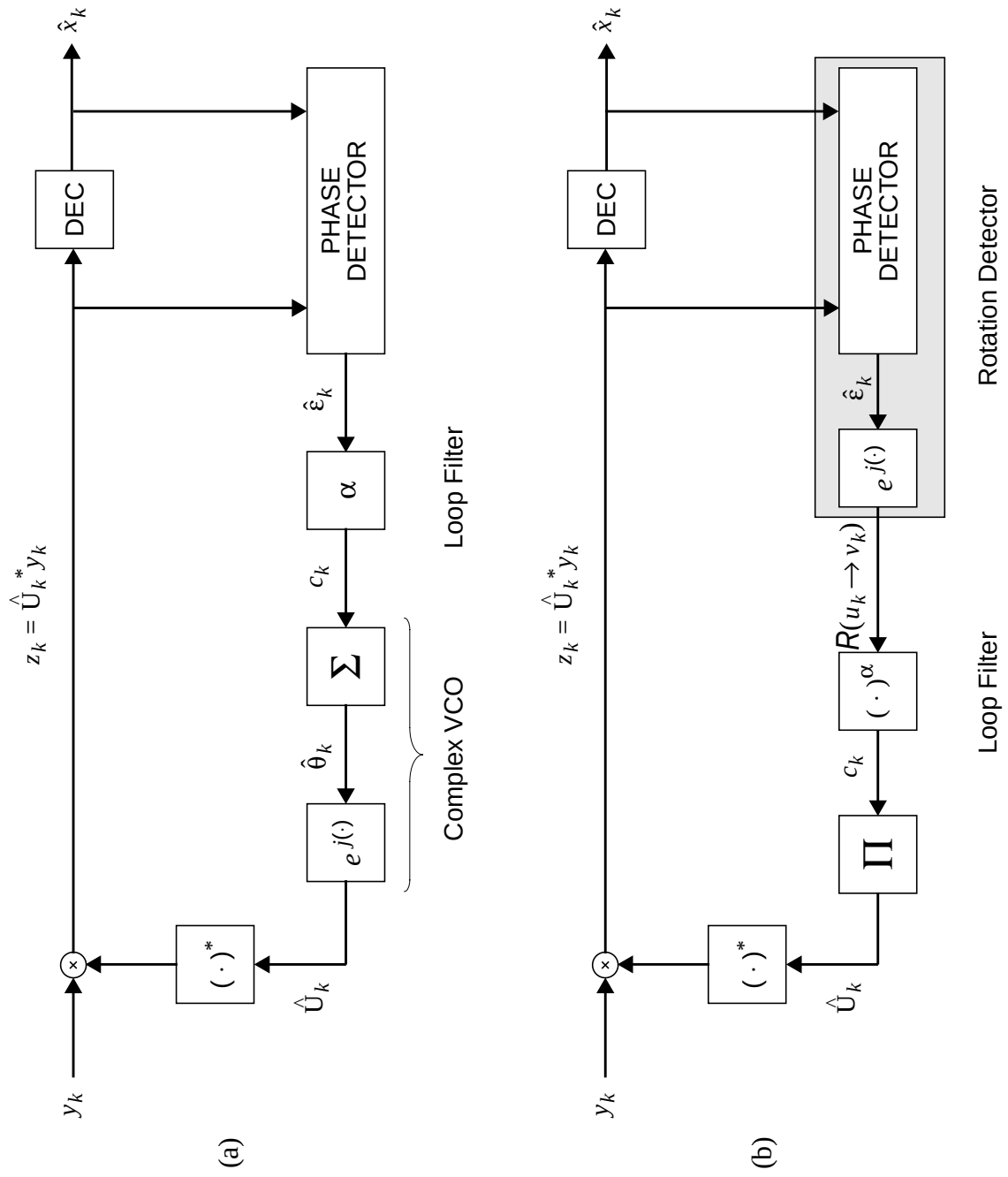


Fig. 3-7. Block diagram of the basic structure of a first-order PLL: (a) conventional model consisting of a phase detector, loop filter, and complex VCO; (b) alternative model consisting of a rotation detector, loop filter, and a product accumulator.

filter, and a complex VCO; and in (3-20), as the cascade of a phase detector, a complex exponentiator, a loop filter, and a product-accumulator. The former description is referred to as the conventional-model DD PLL, while the latter description is referred to as the alternative-model DD PLL.

The structure for the alternative-model DD PLL is displayed in Fig. 3-7(b). The shaded block is referred to as the *rotation detector*, which is the cascade of the phase detector and the complex exponentiator. Define $u_k = \frac{\hat{x}_k}{|\hat{x}_k|}$ and $v_k = \frac{z_k}{|z_k|}$ to be two points on the unit circle. As illustrated in Fig. 3-8, the rotation detector produces an estimate of the rotation between these two points:

$$R(u_k \rightarrow v_k) = exp(j\hat{\varepsilon}_k) = u_k^* v_k. \tag{3-21}$$

The estimate of the rotation is then filtered by a loop filter to produce a control signal for the product-accumulator. For a first-order DD PLL, the control signal is:

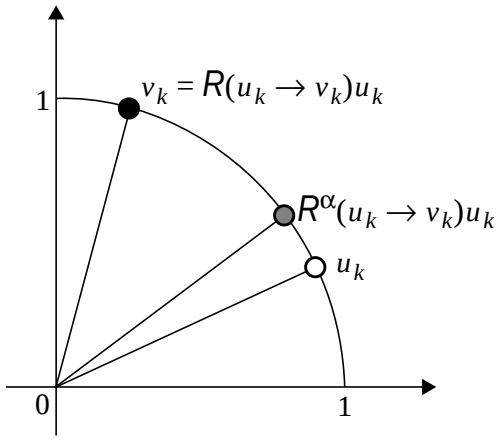


Fig. 3-8. A graphical representation of a complete and partial rotation on the unit circle.

$$C_k = R^{\alpha}(u_k \rightarrow v_k) = exp(j\alpha\hat{\varepsilon}_k). \tag{3-22}$$

As shown in Fig. 3-8, $R^{\alpha}(u_k \rightarrow v_k)$ represents a *partial rotation* that rotates u_k a fraction of the way to v_k.

Finally, the control signal C_k drives the product-accumulator to produce the output of the first-order DD PLL:

$$\hat{U}_k = \prod_{i=0}^{k-1} C_i = \prod_{i=0}^{k-1} R^{\alpha}(u_i \rightarrow v_i). \tag{3-23}$$

We can also express this equation in terms of a recursive update:

$$\hat{U}_{k+1} = \hat{U}_k R^{\alpha}(u_k \rightarrow v_k), \tag{3-24}$$

where $\hat{U}_0 = 1$ represents the initial zero rotation condition, and $u_k = \frac{\hat{x}_k}{|\hat{x}_k|}$ and $v_k = \frac{z_k}{|z_k|}$.

In summary, the alternative-model first-order DD PLL is defined by (3-1), (3-21), and (3-24).

3.2.2 Alternative-Model Second-Order PLL

From (3-4) and (3-15), the VCO output $\hat{U}_k$ is given by:

$$\hat{U}_k = exp\left[\sum_{i=0}^{k-1}(\alpha_1\hat{\varepsilon}_i + \sum_{l=0}^{i}\alpha_2\hat{\varepsilon}_l)\right]. \tag{3-25}$$

Again, we can rewrite (3-25) by exploiting the properties of an exponential function:

$$\hat{\mathrm{U}}_k = \prod_{i=0}^{k-1}\left[exp(j\hat{\varepsilon}_i)^{\alpha_1}\left[\prod_{l=0}^{i} exp(j\hat{\varepsilon}_l)^{\alpha_2} \right] \right]. \tag{3-26}$$

Using the definition of the rotation detector given by (3-21), we can express (3-26) as:

$$\hat{\mathrm{U}}_k = \prod_{i=0}^{k-1}\left[R^{\alpha_1}(u_i \rightarrow v_i)\left[\prod_{l=0}^{i} R^{\alpha_2}(u_l \rightarrow v_l) \right] \right], \tag{3-27}$$

where $R^{\alpha}(u_k \rightarrow v_k) = exp(j\alpha\hat{\varepsilon}_k)$. Finally, we can write (3-27) in terms of a recursive update:

$$\mathrm{W}_{k+1} = \mathrm{W}_k R^{\alpha_2}(u_k \rightarrow v_k), \tag{3-28}$$

$$\hat{\mathrm{U}}_{k+1} = \hat{\mathrm{U}}_k R^{\alpha_1}(u_k \rightarrow v_k)\mathrm{W}_{k+1}, \tag{3-29}$$

where $\hat{\mathrm{U}}_0 = 1$ and $\mathrm{W}_0 = 1$ are the initial zero rotation conditions and $u_k = \frac{\hat{x}_k}{|\hat{x}_k|}$ and $v_k = \frac{z_k}{|z_k|}$. We observe that the alternative model for a second-order PLL is completely described by the output of a first-order loop filter and two recursive updates.

In summary, the alternative-model second-order DD PLL is defined by (3-1), (3-21), (3-28), and (3-29).

3.3 MULTIDIMENSIONAL PHASE-LOCKED LOOP

We can now generalize the structure of the alternative-model PLL to multiple dimensions. The basic structure of the decision-directed multidimensional phase-locked loop (MPLL) consists of three major components [51,52]: a *rotation detector,* a *loop filter,* and a *product-accumulator,* as illustrated in Fig. 3-9. The rotation detector produces an estimate of the rotation between the received signal $\boldsymbol{z}_k$ and the decision $\hat{\boldsymbol{x}}_k$. The resulting estimate of the rotation is then filtered by a loop filter to create a control signal that drives the product-accumulator. The purpose of the control signal is to drive the output of the MPLL to reduce the rotation error between $\hat{\boldsymbol{x}}_k$ and $\boldsymbol{z}_k$.

Suppose that the $n \times 1$ input signal $\boldsymbol{y}_k$ to the MPLL has the following form:

$$\boldsymbol{y}_k = \mathbf{U}_k \boldsymbol{x}_k + \boldsymbol{n}_k, \tag{3-30}$$

where $\boldsymbol{x}_k$ is the $n \times 1$ channel input vector representing the transmitted symbols from the n independent users, $\mathbf{U}_k$ is an $n \times n$ time-varying unitary channel matrix representing the rotation, and $\boldsymbol{n}_k$ is an $n \times 1$ noise vector. We limit the focus of $\mathbf{U}_k$ to a constant rotation, where $\mathbf{U}_k = \mathbf{U}$ and $\mathbf{U}$ is an $n \times n$ unitary matrix, and to an angular rotation, where $\mathbf{U}_k = \mathbf{U}\mathbf{W}^k$, and $\mathbf{U}$ and $\mathbf{W}$ are $n \times n$ unitary matrices.

The object of the MPLL is to generate an output $\hat{\mathbf{U}}_k$ that correctly estimates the rotation $\mathbf{U}_k$ in (3-30). If this estimate is essentially correct ($\hat{\mathbf{U}}_k \approx \mathbf{U}_k$), the MPLL is said to be in rotation-lock, and the received signal $\boldsymbol{z}_k = \hat{\mathbf{U}}_k^* \boldsymbol{y}_k$ reduces to:

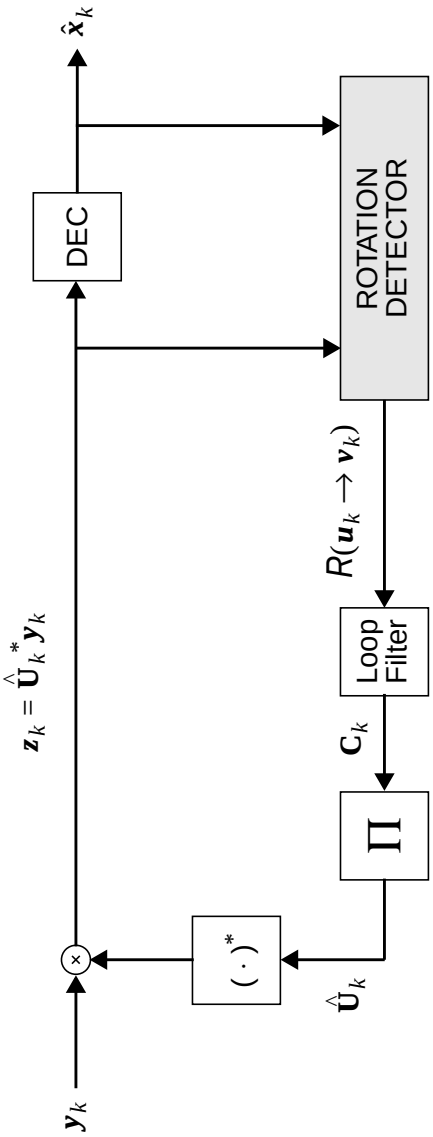


Fig. 3-9. Block diagram of the basic structure of a decision-directed multidimensional phase-locked loop. The main components for MPLL are the rotation detector, loop filter, and product accumulator.

$$\boldsymbol{z}_k = \mathbf{F}_k \boldsymbol{x}_k + \hat{\mathbf{U}}_k^* \boldsymbol{n}_k, \tag{3-31}$$

$$\approx \boldsymbol{x}_k + \boldsymbol{w}_k, \tag{3-32}$$

where $\mathbf{F}_k = \hat{\mathbf{U}}_k^* \mathbf{U}_k$ is the $n \times n$ overall transfer function matrix and $\boldsymbol{w}_k = \hat{\mathbf{U}}_k^* \boldsymbol{n}_k$ is an $n \times 1$ noise vector. At high SNR, (3-32) reduces to $\boldsymbol{z}_k \approx \boldsymbol{x}_k$ and the channel input vector can be recovered by passing $\boldsymbol{z}_k$ through a simple decision device.

As shown in Fig. 3-9, the MPLL generates $\hat{\mathbf{U}}_k$ in a manner similar to the alternative-model PLL. The key component of the MPLL is the rotation detector, which produces an estimate of the rotation by generating a unitary mapping from $\frac{\hat{\mathbf{x}}_k}{\|\hat{\mathbf{x}}_k\|}$ to $\frac{\boldsymbol{z}_k}{\|\boldsymbol{z}_k\|}$. The output of the rotation detector is then passed through a loop filter to produce the control signal $\mathbf{C}_k$. The loop filter serves two purposes: it generates the necessary control signal for the product-accumulator, and it also filters both the noisy and incorrect estimates produced by the rotation detector. The output of the loop filter $\mathbf{C}_k$ drives a product-accumulator to produce the MPLL output $\hat{\mathbf{U}}_k$:

$$\hat{\mathbf{U}}_k = \prod_{i=0}^{k-1} \mathbf{C}_i. \tag{3-33}$$

Because matrix multiplication is not commutative, a decision as to whether the control signal should be pre-multiplied (so that $\hat{\mathbf{U}}_{k+1} = \mathbf{C}_k \hat{\mathbf{U}}_k$) or post-multiplied (so that $\hat{\mathbf{U}}_{k+1} = \hat{\mathbf{U}}_k \mathbf{C}_k$) must be made. The purpose of the control signal is to reduce the rotation error between $\hat{\boldsymbol{x}}_k$ and $\boldsymbol{z}_k$, *i.e.*, the rotation error between $\hat{\boldsymbol{x}}_k$ to $\mathbf{C}_k^* \boldsymbol{z}_k$ is smaller than the rotation

error between $\hat{\mathbf{x}}_k$ and $\mathbf{z}_k$. Since $\mathbf{C}_k^* \mathbf{z}_k = (\hat{\mathbf{U}}_k \mathbf{C}_k)^* \mathbf{y}_k$, it is clear that post-multiplication is the natural choice. Hence, we can express (3-33) in terms of recursive update as follows:

$$\hat{\mathbf{U}}_{k+1} = \hat{\mathbf{U}}_k \mathbf{C}_k, \tag{3-34}$$

where the MPLL is initialized with a zero rotation: $\hat{\mathbf{U}}_0 = \mathbf{I}$. In the remainder of this chapter, a product accumulation as in (3-33) will be taken to mean post-multiplication as in (3-34).

In the following sections, we will derive both the rotation detector and the loop filter, and also define a first-order and a second-order MPLL.

3.3.1 Rotation Detector

Define $\boldsymbol{u}_k = \frac{\hat{\mathbf{x}}_k}{\|\hat{\mathbf{x}}_k\|}$ and $\boldsymbol{v}_k = \frac{\mathbf{z}_k}{\|\mathbf{z}_k\|}$ to be two $n \times 1$ unit-length vectors. The rotation detector produces an estimate of the rotation between these two points. Let the output of the rotation detector be given by $R(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)$ at time k, where the unitary function R is defined as:

> **Definition 3-4.** Let $\boldsymbol{u}$ and $\boldsymbol{v}$ be two $n \times 1$ unit-length vectors. The function R generates a unitary matrix that rotates $\boldsymbol{u}$ to $\boldsymbol{v}$.

For a single-user ($n = 1$), the function R is unique: $R(u_k \rightarrow v_k) = u_k^* v_k$. Unfortunately, for higher dimensions ($n \geq 2$), this function is not unique, but we can derive a particular unitary function R that we will use as the basis for the rotation detector.

In this section[2], we determine a unitary matrix that maps $\boldsymbol{u}$ to $\boldsymbol{v}$. Let p be the normalized inner product: $p = \boldsymbol{u}^* \boldsymbol{v}$. Since $\boldsymbol{u}$ and $\boldsymbol{v}$ have unit length, the magnitude of p is bounded by unity; in other words, $|p| \leq 1$. In deriving the unitary function R, we consider the two cases $|p| = 1$ and $|p| < 1$ separately.

If $|p| = 1$, then the vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ are colinear. Hence, we can express $\boldsymbol{v}$ in terms of $\boldsymbol{u}$ as follows:

$$\boldsymbol{v} = p\boldsymbol{u}. \tag{3-35}$$

We recognize that $\boldsymbol{u}$ is a basis vector for the subspace spanned by $\boldsymbol{u}$ and $\boldsymbol{v}$. Given the vector $\boldsymbol{u}$, we can use the Gram-Schmidt procedure to determine the remaining $n - 1$ vectors $\{\boldsymbol{v}_2, \boldsymbol{v}_3, \ldots, \boldsymbol{v}_n\}$ that form the basis for the n-dimensional space. Let $\mathbf{V}$ be an $n \times n$ matrix whose columns are specified by the basis vectors, *i.e.*,

$$\mathbf{V} = [\boldsymbol{u}\ \boldsymbol{v}_2\ \boldsymbol{v}_3\ \ldots\ \boldsymbol{v}_n]. \tag{3-36}$$

We see that the matrix $\mathbf{V}$ is unitary. In terms of this basis, the vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ can written as $\boldsymbol{u}_V = [1\ 0\ \ldots\ 0]^T$ and $\boldsymbol{v}_V = [p\ 0\ \ldots\ 0]^T$ respectively. A unitary matrix that maps $\boldsymbol{u}_V$ to $\boldsymbol{v}_V$ is clearly given by:

$$\mathbf{Q} = \left[\begin{array}{c|c} p & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{J} \end{array}\right], \tag{3-37}$$

2. The dependence on time has been suppressed, for the rest of this section, to simplify notation.

where $\mathbf{J}$ is an arbitrary $(n-1)\times(n-1)$ unitary matrix. Mapping the unitary matrix (3-37) back to Euclidean space, we see that the unitary matrix that maps $\boldsymbol{u}$ to $\boldsymbol{v}$ is given by:

$$R(\boldsymbol{u}\rightarrow\boldsymbol{v}) = \mathbf{V}\left[\begin{array}{c|c} p & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{J} \end{array}\right]\mathbf{V}^{*}. \tag{3-38}$$

The vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ only span a one-dimensional subspace. Since we do not have information about the remaining $(n-1)$ dimensions, we chose not to rotate vectors that are orthogonal to either $\boldsymbol{u}$ or $\boldsymbol{v}$ by taking $\mathbf{J} = \mathbf{I}_{n-1}$. Substituting the identity matrix for $\mathbf{J}$ in (3-38), we find that $R(\boldsymbol{u}\rightarrow\boldsymbol{v})$ reduces to:

$$R(\boldsymbol{u}\rightarrow\boldsymbol{v}) = \mathbf{I}_n + (p-1)\boldsymbol{u}\boldsymbol{u}^{*}. \tag{3-39}$$

We should emphasize that this result is only valid when $|p| = 1$.

On the other hand, if $|p| < 1$, then the vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ span a two-dimensional subspace. Given the vectors $\boldsymbol{u}$ and $\boldsymbol{v}$, we can use the Gram-Schmidt procedure to determine the n vectors that form the basis for the n-dimensional subspace. These basis vectors are given by $\{\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{v}_3, \boldsymbol{v}_4, \ldots, \boldsymbol{v}_n\}$, where

$$\boldsymbol{w} = \frac{\boldsymbol{v} - p\,\boldsymbol{u}}{\sqrt{1-|p|^2}}. \tag{3-40}$$

Let $\mathbf{V}$ be an $n\times n$ matrix whose columns are specified by the basis vectors, *i.e.*,

$$\mathbf{V} = [\boldsymbol{u}\ \boldsymbol{w}\ \boldsymbol{v}_3\ \boldsymbol{v}_4\ \ldots\ \boldsymbol{v}_n]. \tag{3-41}$$

We observe that $\mathbf{V}$ is a unitary matrix. The vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ can be written in terms of the basis vectors (3-41) as $\boldsymbol{u}_{\mathrm{V}} = [1\ 0\ \ldots\ 0]^T$ and $\boldsymbol{v}_{\mathrm{V}} = [p\ \sqrt{1-|p|^2}\ \ 0\ \ldots\ 0]^T$. A unitary matrix that maps $\boldsymbol{u}_{\mathrm{V}}$ to $\boldsymbol{v}_{\mathrm{V}}$ is given by:

$$\mathbf{Q} = \left[\begin{array}{c|c} \mathbf{R} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{J} \end{array}\right], \tag{3-42}$$

where $\mathbf{J}$ is an arbitrary $(n-2)\times(n-2)$ unitary matrix, and

$$\mathbf{R} = \begin{bmatrix} p & -\sqrt{1-|p|^2} \\ \sqrt{1-|p|^2} & p^* \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & e^{j\beta} \end{bmatrix}, \tag{3-43}$$

where $\beta \in (-\pi, \pi]$. Mapping the unitary matrix (3-42) back to an Euclidean space, we see that the unitary matrix that maps $\boldsymbol{u}$ to $\boldsymbol{v}$ is given by:

$$R(\boldsymbol{u} \rightarrow \boldsymbol{v}) = \mathbf{V}\left[\begin{array}{c|c} \mathbf{R} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{J} \end{array}\right]\mathbf{V}^*. \tag{3-44}$$

Recall that the vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ span a two-dimensional subspace. Since we have no information about the remaining $(n-2)$ dimensions, we chose not to rotate vectors that

are orthogonal to this two-dimensional subspace by taking $\mathbf{J} = \mathbf{I}_{n-2}$. The choice for β is not as obvious. We want (3-44) to be consistent with (3-38) as $|p|$ approaches one. We see that when $|p| < 1$,

$$det[R(\boldsymbol{u} \rightarrow \boldsymbol{v})] = exp(j\beta), \tag{3-45}$$

and when $|p| = 1$,

$$det[R(\boldsymbol{u} \rightarrow \boldsymbol{v})] = p. \tag{3-46}$$

Therefore, as $|p| \rightarrow 1$, $exp(j\beta) \rightarrow p$. We should thus choose $\beta = \angle p = sin^{-1}[Im(\frac{p}{|p|})]$. This choice of β also minimizes the Frobenius norm of $\mathbf{R} - \mathbf{I}_2$, which is reassuring because we expect $R(\boldsymbol{u} \rightarrow \boldsymbol{v})$ to approach the identity matrix near convergence. Substituting $\mathbf{J} = \mathbf{I}_{n-2}$ and $\beta = \angle p$ in (3-44), we find that the unitary matrix $R(\boldsymbol{u} \rightarrow \boldsymbol{v})$ reduces to:

$$R(\boldsymbol{u} \rightarrow \boldsymbol{v}) = \mathbf{I}_n + \begin{bmatrix} \boldsymbol{u} & \boldsymbol{w} \end{bmatrix} \begin{bmatrix} p-1 & -\frac{p}{|p|}\sqrt{1-|p|^2} \\ \sqrt{1-|p|^2} & |p|-1 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^* \\ \boldsymbol{w}^* \end{bmatrix}. \tag{3-47}$$

In summary, the unitary function R is defined as follows:

$$R(\boldsymbol{u} \rightarrow \boldsymbol{v}) = \mathbf{I}_n + \begin{bmatrix} \boldsymbol{u} & \boldsymbol{w} \end{bmatrix} \begin{bmatrix} p-1 & -\frac{p}{|p|}\sqrt{1-|p|^2} \\ \sqrt{1-|p|^2} & |p|-1 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^* \\ \boldsymbol{w}^* \end{bmatrix}, \tag{3-48}$$

where $\boldsymbol{u} = \frac{\hat{\boldsymbol{x}}}{\|\hat{\boldsymbol{x}}\|}$, $\boldsymbol{v} = \frac{\boldsymbol{z}}{\|\boldsymbol{z}\|}$, $p = \boldsymbol{u}^*\boldsymbol{v}$, and $\boldsymbol{w} = (\boldsymbol{v} - p\boldsymbol{u})/\sqrt{1 - |p|^2}$ if $|p| < 1$, otherwise, $\boldsymbol{w} = 0$. We should point out that when $n = 1$, the unitary matrix $R(u \rightarrow v)$ reduces to u^*v, which is the familiar single-user unitary mapping described in Section 3.2.1.

3.3.2 Loop Filter

A PLL of any order can be written in terms of the output of a first-order loop filter and a recursive update (see Section 3.2.2). The output of a first-order loop filter is generated by raising the unitary matrix $R(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)$ to a fractional power $\lambda \in (0, 1)$, where λ is the step size of the MPLL. The loop filter output, which rotates $\boldsymbol{u}_k$ a fraction λ of the way to $\boldsymbol{v}_k$, is referred to as a *partial rotation matrix*.

One method for raising a matrix to a fractional power requires calculating an eigendecomposition, which is inherently a computationally intensive task. We can avoid most of the computational complexity by symbolically determining the partial rotation matrix $R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)$. We begin by considering the two cases: $|p| = 1$ and $|p| < 1$, separately[3].

When $|p| = 1$, the partial rotation matrix $R^{\lambda}(\boldsymbol{u} \rightarrow \boldsymbol{v})$ is found by raising (3-38), with $\mathbf{J} = \mathbf{I}_{n-1}$, to a fractional power λ:

$$R^{\lambda}(\boldsymbol{u} \rightarrow \boldsymbol{v}) = \mathbf{V}\left[\begin{array}{c|c} p^{\lambda} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{I}_{n-1} \end{array}\right]\mathbf{V}^*, \tag{3-49}$$

where $\mathbf{V}$ is defined by (3-36). By expanding this matrix, we find that (3-49) reduces to:

---

3. We again suppress the dependence on time, for the rest of this section, to simplify notation.

$$R^{\lambda}(\boldsymbol{u} \rightarrow \boldsymbol{v}) = \mathbf{I}_n + (p^{\lambda} - 1)\boldsymbol{u}\boldsymbol{u}^{*}. \tag{3-50}$$

where $\boldsymbol{u} = \frac{\hat{\boldsymbol{x}}}{\|\hat{\boldsymbol{x}}\|}$, $\boldsymbol{v} = \frac{\boldsymbol{z}}{\|\boldsymbol{z}\|}$, $p = \boldsymbol{u}^{*}\boldsymbol{v}$.

When $|p| < 1$, we can find the partial rotation matrix $R^{\lambda}(\boldsymbol{u} \rightarrow \boldsymbol{v})$ by raising (3-44), with $\mathbf{J} = \mathbf{I}_{n-2}$, to a fractional power λ:

$$R^{\lambda}(\boldsymbol{u} \rightarrow \boldsymbol{v}) = \mathbf{V}\left[\begin{array}{c|c} \mathbf{R}^{\lambda} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{I}_{n-2} \end{array}\right]\mathbf{V}^{*}. \tag{3-51}$$

where $\mathbf{V}$ is defined by (3-41) and $\mathbf{R}$ is defined by (3-43). By expanding the components of this matrix, we see that (3-51) reduces to:

$$R^{\lambda}(\boldsymbol{u} \rightarrow \boldsymbol{v}) = \mathbf{I}_n + \begin{bmatrix}\boldsymbol{u} & \boldsymbol{w}\end{bmatrix}(\mathbf{R}^{\lambda} - \mathbf{I}_2)\begin{bmatrix}\boldsymbol{u}^{*} \\ \boldsymbol{w}^{*}\end{bmatrix}. \tag{3-52}$$

We can simplify (3-52) even further by symbolically calculating $\mathbf{R}^{\lambda}$.

Since $\mathbf{R}$ is a unitary matrix, it has the unique eigendecomposition: $\mathbf{R} = \mathbf{W}\mathbf{D}\mathbf{W}^{*}$ [106]. The $n \times n$ diagonal unitary matrix $\mathbf{D}$ is given by:

$$\mathbf{D} = \begin{bmatrix} e^{j\left(\frac{\beta}{2} - \Delta\right)} & & \\ & 0 & \\ & & e^{j\left(\frac{\beta}{2} + \Delta\right)} \end{bmatrix}, \tag{3-53}$$

where $\beta = \angle p$ and $\Delta = sin^{-1}[\,|p|cos(\beta/2)] - \pi/2$. The $n \times n$ unitary matrix $\mathbf{W}$ is given by:

$$\mathbf{W} = \begin{bmatrix} \alpha & -j\gamma e^{j\frac{\beta}{2}} \\ -j\gamma e^{-j\frac{\beta}{2}} & \alpha \end{bmatrix}, \tag{3-54}$$

where $\alpha = \sqrt{\frac{1}{2} - \frac{|p|sin(\beta/2)}{2sin(\Delta)}}$ and $\gamma = \sqrt{\frac{1}{2} + \frac{|p|sin(\beta/2)}{2sin(\Delta)}}$. Using the eigendecomposition, we can easily determine $\mathbf{R}^\lambda = \mathbf{W}\mathbf{D}^\lambda\mathbf{W}^*$:

$$\mathbf{R}^\lambda = e^{j\lambda\frac{\beta}{2}} \begin{bmatrix} cos(\lambda\Delta) + j\zeta|p|sin(\beta/2) & -\zeta e^{j\frac{\beta}{2}}\sqrt{1-|p|^2} \\ \zeta e^{-j\frac{\beta}{2}}\sqrt{1-|p|^2} & cos(\lambda\Delta) - j\zeta|p|sin(\beta/2) \end{bmatrix}, \tag{3-55}$$

where $\zeta = \frac{sin(\lambda\Delta)}{sin(\Delta)}$. Using the fact that $cos(\Delta) = |p|cos(\beta/2)$, it is easy to show that (3-55) reduces to (3-43) when $\lambda = 1$.

In summary, the unitary function R is defined as follows:

$$R^\lambda(\boldsymbol{u} \rightarrow \boldsymbol{v}) = \mathbf{I}_n + \begin{bmatrix} \boldsymbol{u} & \boldsymbol{w} \end{bmatrix}(\mathbf{R}^\lambda - \mathbf{I}_2)\begin{bmatrix} \boldsymbol{u}^* \\ \boldsymbol{w}^* \end{bmatrix}, \tag{3-56}$$

where $\boldsymbol{u} = \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|}$, $\boldsymbol{v} = \frac{\boldsymbol{z}}{\|\boldsymbol{z}\|}$, $p = \boldsymbol{u}^*\boldsymbol{v}$, $\boldsymbol{w} = (\boldsymbol{v} - p\boldsymbol{u})/\sqrt{1-|p|^2}$ if $|p| < 1$, otherwise, $\boldsymbol{w} = 0$, and $\mathbf{R}^\lambda$ is defined by (3-55). In a practical application, the partial rotation matrix specified by (3-56) should be used instead of actually raising the output of the rotation detector to a fractional power.

3.3.3 First-Order MPLL

A first-order MPLL is sufficient to track a constant rotation, where $\mathbf{U}_k = \mathbf{U}$. The output of a first-order DD MPLL is given by:

$$\hat{\mathbf{U}}_k = \prod_{i=0}^{k-1} R^{\lambda}(\boldsymbol{u}_i \rightarrow \boldsymbol{v}_i), \tag{3-57}$$

where $R(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)$ is the output of the rotation detector, $\boldsymbol{u}_k = \frac{\hat{\boldsymbol{x}}_k}{\|\hat{\boldsymbol{x}}_k\|}$, $\boldsymbol{v}_k = \frac{\boldsymbol{z}_k}{\|\boldsymbol{z}_k\|}$, and λ is some positive constant. As mentioned earlier, the natural choice is to accumulate the partial rotation matrices on the right. Hence, a recursive update equation for (3-57) is given by:

$$\hat{\mathbf{U}}_{k+1} = \hat{\mathbf{U}}_k R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k), \tag{3-58}$$

where $\hat{\mathbf{U}}_0 = \mathbf{I}$ is the initial condition.

We see that the recursive update specified by (3-58) requires the multiplication of two $n \times n$ matrices at each iteration. We can reduce the complexity of this update by manipulating (3-58) into the following form:

$$\hat{\mathbf{U}}_{k+1} = \hat{\mathbf{U}}_k + \hat{\mathbf{U}}_k\left(R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k) - \mathbf{I}\right). \tag{3-59}$$

While this update seems similar to the update given by (3-58), it is, in fact, less complex because the term in the parenthesis of (3-59) is at most a rank-two matrix. If $|p| = 1$, this term is the product of an $n \times 1$ vector with a $1 \times n$ vector. If the product in (3-59) is carried

out from left to right, the number of multiplications, for large values of n, is significantly less than that required to multiply two $n \times n$ matrices, as in (3-58). If $|p| < 1$, then the term in the parenthesis is the product of three matrices: an $n \times 2$ matrix, a 2×2 matrix, and a $2 \times n$ matrix. Again, if the product in (3-59) is carried out from left to right, then we realize the same computational saving in this case as well when n is large. In the following experiment, we compare the total number of multiplications that are used to generate $\hat{\mathbf{U}}_{k+1}$ in (3-58) and (3-59).

Experiment 3-5. In (3-58), n^3 multiplications are needed to generate $\hat{\mathbf{U}}_{k+1}$, while in (3-59), it takes $2n^2+n$ and $4n^2+4n$ multiplications to generate the update for the $|p| = 1$ and $|p| < 1$ cases, respectively. In Fig. 3-10, we plot the total of multiplications for each update versus the number of users. From this figure, we see that when $n \geq 5$, the update given by (3-59) requires fewer total number of multiplications (for both $|p| = 1$ and $|p| < 1$) than the update given by (3-58).

In summary, a first-order MPLL is defined by (3-55), (3-56) and (3-59).

3.3.4 Second-Order MPLL

A second-order MPLL is necessary to track an angular rotation: $\mathbf{U}_k = \mathbf{U}\mathbf{W}^k$. The output of a second-order DD MPLL output is given by:

$$\hat{\mathbf{U}}_k = \prod_{i=0}^{k-1} \left[R^{\lambda_1}(\boldsymbol{u}_i \rightarrow \boldsymbol{v}_i) \left[\prod_{l=0}^{i} R^{\lambda_2}(\boldsymbol{u}_l \rightarrow \boldsymbol{v}_l) \right] \right], \tag{3-60}$$

where $R(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)$ is the output of the rotation detector, $\boldsymbol{u}_k = \frac{\hat{\boldsymbol{x}}_k}{\|\hat{\boldsymbol{x}}_k\|}$, $\boldsymbol{v}_k = \frac{\boldsymbol{z}_k}{\|\boldsymbol{z}_k\|}$, and λ_1 and λ_2 are small positive constants. We can also write (3-60) in terms of a recursive update:

$$\mathbf{W}_{k+1} = \mathbf{W}_k R^{\lambda_2}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k), \tag{3-61}$$

$$\hat{\mathbf{U}}_{k+1} = \hat{\mathbf{U}}_k R^{\lambda_1}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)\mathbf{W}_{k+1}, \tag{3-62}$$

where $\hat{\mathbf{U}}_0 = \mathbf{I}$ and $\mathbf{W}_0 = \mathbf{I}$ are the initial conditions for the second-order MPLL.

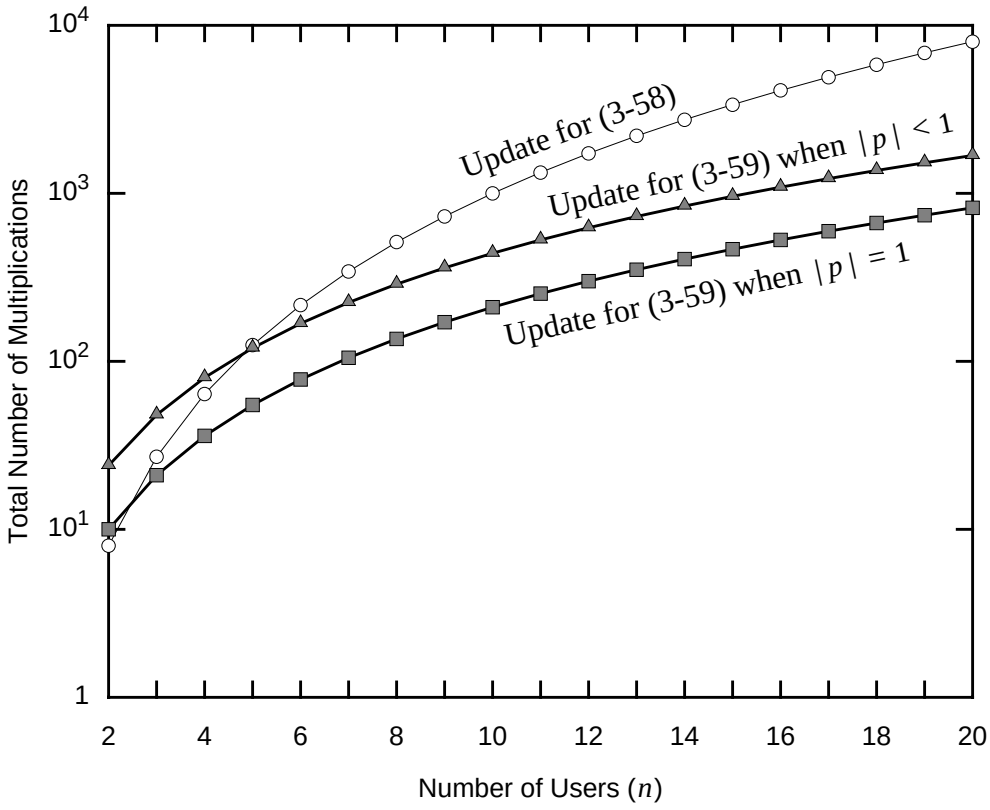


Fig. 3-10. A comparison of the total number of multiplications that are needed to generate the update in (3-58) and (3-59).

Again, we can reduce the complexity of the update equations given by (3-61) and (3-62) by manipulating them into the following form:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mathbf{W}_k\left(R^{\lambda_2}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k) - \mathbf{I}\right), \tag{3-63}$$

$$\hat{\mathbf{U}}_{k+1} = \hat{\mathbf{U}}_k\mathbf{W}_{k+1} + \hat{\mathbf{U}}_k\left(R^{\lambda_1}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k) - \mathbf{I}\right)\mathbf{W}_{k+1}. \tag{3-64}$$

The reason that the update equations given by (3-63) and (3-64) are less complex, when n is large, is the same as the one given for a first-order MPLL. We should point out that in order to obtain the reduction in complexity, the products in (3-63) and (3-64) must be carried out from left to right.

In summary, the second-order MPLL is defined by the equations (3-55), (3-56), (3-63), and (3-64).

3.4 Alternative Model for the MPLL

In the previous section, we showed that the output of the combined rotation detector and loop filter is a partial rotation matrix $R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)$ that rotates $\boldsymbol{u}_k$ to some intermediate point on the unit hypersphere between $\boldsymbol{u}_k$ and $\boldsymbol{v}_k$. In this section, we develop an alternative model for the MPLL by switching the order of the rotation detector and the loop filter. We first select an intermediate point between $\boldsymbol{u}_k$ and $\boldsymbol{v}_k$, and then we project this point on the unit hypersphere. Finally, we use the rotation detector to determine the unitary matrix that rotates $\boldsymbol{u}_k$ to the normalized intermediate point. The unitary matrix that is obtained in this case is also a partial rotation matrix.

The output of the combined loop filter and rotation detector is derived below[4]. Let $\tilde{\boldsymbol{v}}$ be some intermediate point between $\boldsymbol{u}$ and $\boldsymbol{v}$:

$$\tilde{\boldsymbol{v}}_{\mu} = (1-\mu)\boldsymbol{u} + \mu\boldsymbol{v}, \tag{3-65}$$

where μ is the step size of the alternative-model MPLL. Define $\boldsymbol{v}_{\mu} = \frac{\tilde{\boldsymbol{v}}_{\mu}}{\|\tilde{\boldsymbol{v}}_{\mu}\|}$ to be a point on the unit hypersphere. We can now use the rotation detector defined in Section 3.3.1 to determine the unitary matrix that rotates $\boldsymbol{u}$ to $\boldsymbol{v}_{\mu}$. In terms of the notation for the rotation detector, the partial rotation matrix is given by $R(\boldsymbol{u} \rightarrow \boldsymbol{v}_{\mu})$.

We have shown that both the conventional-model MPLL and the alternative-model MPLL generate a partial rotation matrix which rotates $\boldsymbol{u}$ to some intermediate point. If the input alphabet is real, then it is possible, for a given λ, to select a μ such that the intermediate points from the two MPLLs are the same. We prove this result in the following theorem.

Theorem 3-5. Let each component of the channel input $\boldsymbol{x}$ draw symbols from a real discrete-input alphabet. For a given λ, the conventional-model and the alternative-model MPLL will generate the same partial rotation matrix if

$$\mu = \frac{sin(\lambda\Delta)}{sin(\lambda\Delta) + sin((1-\lambda)\Delta)}, \tag{3-66}$$

where $\Delta = sin^{-1}(\sqrt{1-|p|^2})$.

4. We again suppress the dependence on time, for the rest of this section, to simplify notation.

Proof: See Appendix 3.1.

We should point out that this result is independent of the number of users n. However, this result does not extend to the case of complex input alphabets. Consider the following counter-example:

Example 3-1. We assume that each component of the channel $\boldsymbol{x}$ draws symbols from a 16-QAM input alphabet. Let the input to the decision device be given by $\boldsymbol{z} = [-0.7 - j0.2, -0.1 + j1.8]^T$ and let $\lambda = 0.5$. The corresponding output of the decision device is given by $\hat{\boldsymbol{x}} = [-1 - j, -1 + j]^T$. Since $\boldsymbol{z}_k$, $\hat{\boldsymbol{x}}_k$, and λ are fixed quantities, the conventional-model MPLL rotates the $\boldsymbol{u} = \frac{\hat{\boldsymbol{x}}}{\|\hat{\boldsymbol{x}}\|}$ to a fixed intermediate point $\boldsymbol{v}_\lambda = R^\lambda(\boldsymbol{u} \to \boldsymbol{v})\boldsymbol{u}$, where $R^\lambda(\boldsymbol{u} \to \boldsymbol{v})$ is defined by (3-55) and (3-56). If, on the other hand, we allow μ to vary between 0 and 1, then the alternative-model MPLL rotates $\boldsymbol{u}$ to the intermediate point $\boldsymbol{v}_\mu = \frac{\tilde{\boldsymbol{v}}_\mu}{\|\tilde{\boldsymbol{v}}_\mu\|}$, where $\tilde{\boldsymbol{v}}_\mu$ is defined by (3-65). We observe that $\boldsymbol{v}_\mu$ is a function of μ. The squared error between the two intermediate points $\varepsilon_\mu = \|\boldsymbol{v}_\lambda - \boldsymbol{v}_\mu\|^2$ versus μ is plotted in Fig. 3-11. It is seen that the error never goes to zero, which implies that there does not exist a μ such that $\boldsymbol{v}_\lambda = \boldsymbol{v}_\mu$. However, it is important to point out that for $\mu \cong 0.5071$, the Euclidean distance between the two intermediate points, $\boldsymbol{v}_\lambda$ and $\boldsymbol{v}_\mu$, is found to be very small.

As this example demonstrates, for a given λ, it is possible that there does not exist a μ for which $\boldsymbol{v}_\lambda = \boldsymbol{v}_\mu$. However, it is very likely that there does exists a μ such that the Euclidean distance between $\boldsymbol{v}_\mu$ and $\boldsymbol{v}_\lambda$ is very small. We give the following heuristic argument to support this statement. The distance between the decision $\hat{\boldsymbol{x}}$ and the decision

device $\boldsymbol{z}$ is generally small and projecting these two points onto the unit hypersphere only reduces the distance between them. Since the distance between $\boldsymbol{u}$ and $\boldsymbol{v}$ is very small, all convex paths between them on the unit hypersphere, though generally different for the two MPLLs, would also lie very close to each other. Hence, it is possible, for a given λ, to find a μ such that the intermediate points produced by the conventional-model MPLL and the alternative-model MPLL are nearly identical, and the performance for these two MPLLs should therefore be similar. This conclusion is supported by the following experiment:

Experiment 3-6. In this experiment, we compare the performance of the convention-model MPLL with that of the alternative-model MPLL. We assume that the input alphabet for both models is 16-QAM and that there is no noise.

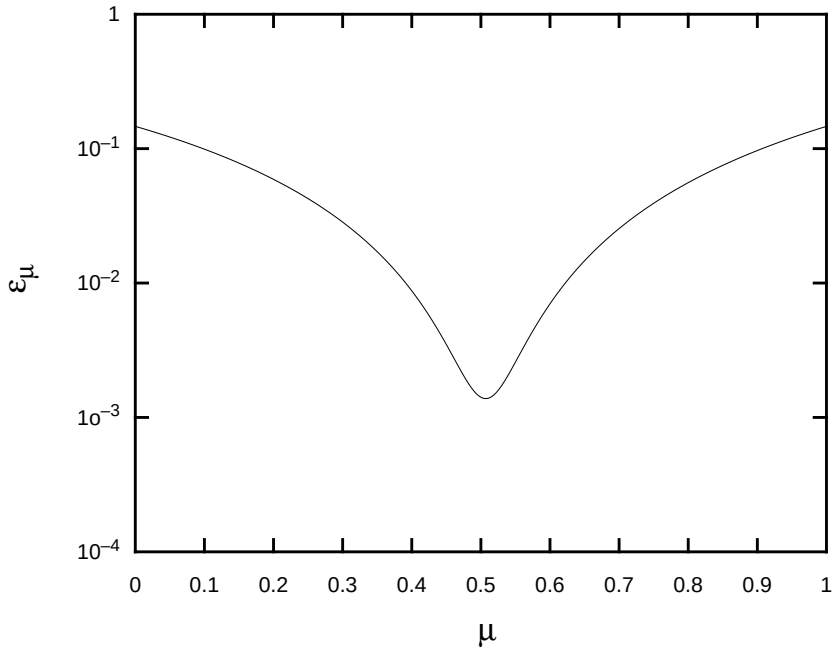


Fig. 3-11. The squared error between the two intermediate points $\boldsymbol{v}_\lambda$ and $\boldsymbol{v}_\mu$ as a function of μ.

The step sizes for the conventional-model MPLL and the alternative-model MPLL were $\lambda = 0.8$ and $\mu = 0.8$, respectively. In Fig. 3-12, we plot the $\mathrm{MSE}_1 = E[\,|\,\bar{x}_k^{(1)} - z_k^{(1)}\,|^2]$, where $\bar{\boldsymbol{x}}_k = \mathbf{P}\boldsymbol{x}_k$ is a permutation of the channel input vector and $\mathbf{P}$ is a complex permutation matrix that accounts for the inherent ambiguity associated with a blind detection problem and a $\frac{\pi}{2}$-symmetric input alphabet, versus time. The curves displayed therein represent data that was averaged over 10,000 random 3×3 memoryless unitary channels. We conclude from the two curves that the performance for the two models is essentially indistinguishable.

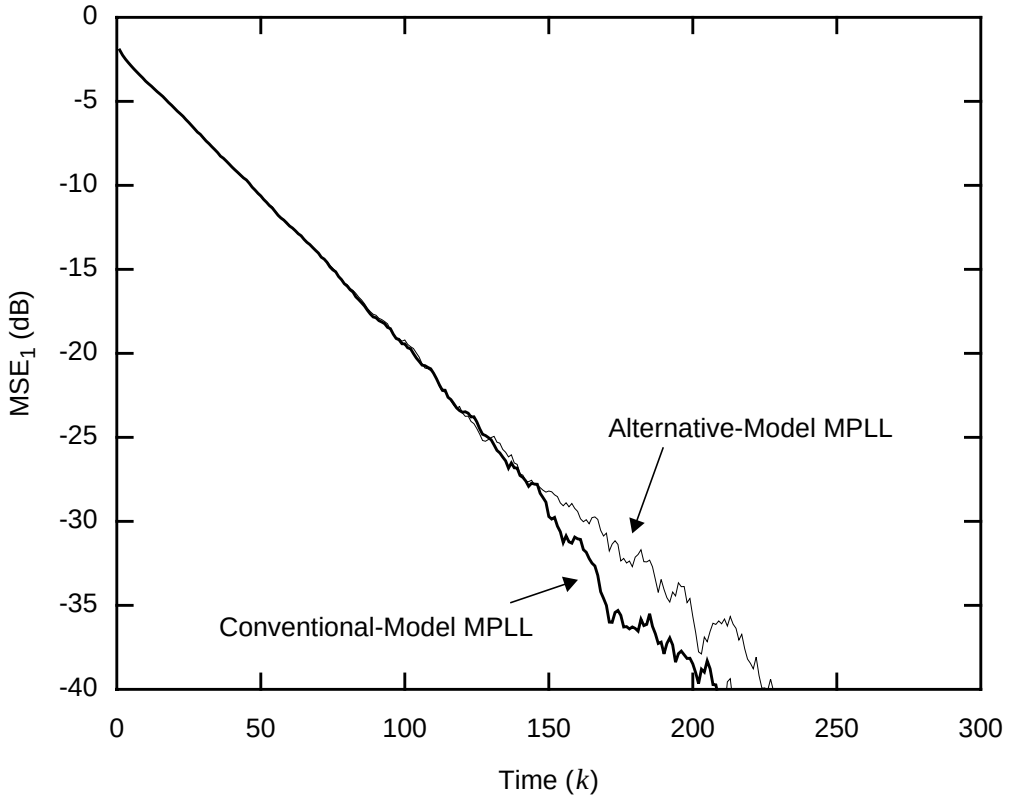


Fig. 3-12. Comparison of the conventional-model MPLL and the alternative-model MPLL in terms of MSE_1 versus time.

The conventional-model MPLL differs from the alternative-model MPLL in the degree of computational complexity. The alternative-model MPLL is inherently less complex, because it does not require an eigendecomposition. In the following experiment, we calculate the number of floating point operation (FLOPS) that are needed to generate each of the two partial rotation matrices.

Experiment 3-7. Consider the channel model described by (3-30). In Fig. 3-13, we plot the number of floating point operations (FLOPS) that are needed per iteration to generate an estimate of the rotation for each MPLL versus the

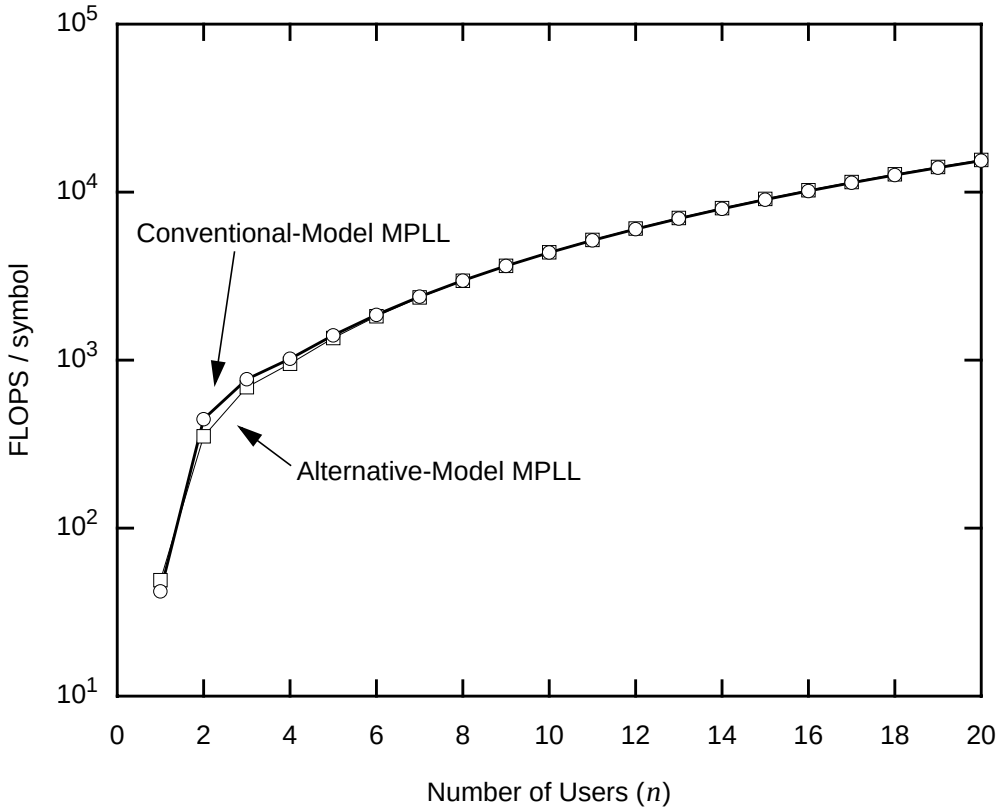


Fig. 3-13. Comparison of computational complexity for the conventional-model MPLL and the alternative-model MPLL.

number of users n. We observe that both the MPLLs have essentially the same complexity $O(n^3)$.

From these curves, we observe that the difference in computational complexity for the two models is negligible for large values of n.

The implementation of both the conventional and the alternative models of the MPLL are equally valid and interesting in their own right. However, in the remainder of this chapter, we will focus our attention on the implementation of the conventional model MPLL (see Section 3.3). Since its structure closely parallels that of the PLL, the insight gained from the analysis of the PLL could prove useful while analyzing the MPLL.

3.5 Convergence Analysis for a First-Order MPLL

In this section we examine the convergence behavior of a first-order MPLL both in the presence and absence of noise.

3.5.1 In the Absence of Noise

Since the MPLL is a decision-directed algorithm, errors in the decision will produce errors at the rotation detector. Proper operation of the MPLL depends on the decisions being correct. We therefore need to verify that errors at the rotation detector do not prevent the MPLL from estimating the rotation in the input signal correctly.

The following convergence analysis for a first-order MPLL is an extension of the analysis presented earlier for a first-order PLL. We assume that each component of the channel input vector $\boldsymbol{x}_k$ is drawn uniformly from a discrete-input alphabet and that the rotation in the input signal $\boldsymbol{y}_k$ is a constant, $\mathbf{U}_k = \mathbf{U}$, where $\mathbf{U}$ is an $n \times n$ unitary matrix. We also

assume that the noise vector $\boldsymbol{n}_k$ in (3-30) is zero. (Later in this section, the effect of noise on the convergence of a first-order MPLL is considered.)

The update equation for a first-order MPLL is given by:

$$\hat{\mathbf{U}}_{k+1} = \hat{\mathbf{U}}_k R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k), \tag{3-67}$$

where $\hat{\mathbf{U}}_k$ is an estimate of the rotation in the input signal, $R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)$ is the partial rotation matrix, and λ is the step size. Since $\boldsymbol{z}_k = \hat{\mathbf{U}}_k^{*}\mathbf{U}\boldsymbol{x}_k$ in the absence of noise, the partial rotation matrix $R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)$ is a function of both $\boldsymbol{x}_k$ and $\hat{\mathbf{U}}_k$. If the step size λ is small, then $\hat{\mathbf{U}}_k$ will vary slowly when compared to $R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)$. Hence, when we take the expectation of (3-67) with respect to the statistics of $R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)$, we can assume that the estimate of the rotation $\hat{\mathbf{U}}_k$ is a constant with respect to this expectation [1]:

$$\hat{\mathbf{U}}_{k+1} = \hat{\mathbf{U}}_k E[R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)], \tag{3-68}$$

where the expectation is taken with respect to $\boldsymbol{x}_k$. Even though $\hat{\mathbf{U}}_k$ is varying slowly, it is still a random process and therefore, we must take the expectation of both side of (3-68):

$$E[\hat{\mathbf{U}}_{k+1}] = E[\hat{\mathbf{U}}_k]E[R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)]. \tag{3-69}$$

The MPLL converges in the mean when $E[\hat{\mathbf{U}}_{k+1}] = E[\hat{\mathbf{U}}_k]$, or equivalently when,

$$E[R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)] = \mathbf{I}. \tag{3-70}$$

Recall that $\boldsymbol{z}_k = \mathbf{F}_k \boldsymbol{x}_k$ where $\mathbf{F}_k = \hat{\mathbf{U}}_k^* \mathbf{U}_k$ is the overall transfer function of the MPLL. Since the partial rotation matrix $R^{\lambda}(\boldsymbol{u}_k \rightarrow \boldsymbol{v}_k)$ is a function of both $\mathbf{F}_k$ and $\boldsymbol{x}_k$, we can define the following function:

$$S(\mathbf{F}) = E[R^{\lambda}(\boldsymbol{u} \rightarrow \boldsymbol{v}) - \mathbf{I} \,|\, \mathbf{F}], \tag{3-71}$$

where again the expectation is taken over the channel input vector. We refer to this function as the multidimensional S-curve (MS-curve) for a first-order MPLL. This function can be used to determine the stable points of a first-order MPLL. The matrix $\mathbf{F}_s$ is a stable point when $S(\mathbf{F}_s) = \mathbf{0}$, or equivalently when,

$$\frac{1}{|\chi|} \sum_{\boldsymbol{u} \in \chi} [R^{\lambda}(\boldsymbol{u} \rightarrow \mathbf{F}_s \boldsymbol{u}) - \mathbf{I}] = \mathbf{0}, \tag{3-72}$$

where χ is the set of all channel input vectors, which have been normalized to have unit length, and $|\chi|$ denotes the number of elements in the set χ. Using (3-72), we can define a stable point of a first-order MPLL as follows:

Definition 3-5. An $n \times n$ unitary matrix $\mathbf{F}_s$ is a *stable point* of a first-order MPLL if it satisfies the following relationship:

$$E[R^{\lambda}(\boldsymbol{u} \rightarrow \boldsymbol{v}) - \mathbf{I} \,|\, \mathbf{F}_s] = \frac{1}{|\chi|} \sum_{\boldsymbol{u} \in \chi} [R^{\lambda}(\boldsymbol{u} \rightarrow \mathbf{F}_s \boldsymbol{u}) - \mathbf{I}] = \mathbf{0}, \tag{3-73}$$

where χ is the set of all channel input vectors that have been normalized to have unit length, and $|\chi|$ denotes the number of elements in the set χ.

We can derive an alternative expression for (3-73) by using the binomial expansion of $R^{\lambda}(\boldsymbol{u} \rightarrow \mathbf{F}_s\boldsymbol{u})$. For a very small λ, we find that $R^{\lambda}(\boldsymbol{u} \rightarrow \mathbf{F}_s\boldsymbol{u}) \cong \mathbf{I} + \lambda(R(\boldsymbol{u} \rightarrow \mathbf{F}_s\boldsymbol{u}) - \mathbf{I})$. Substituting this approximation into (3-72), we obtain

$$\frac{1}{|\chi|} \sum_{\boldsymbol{u} \in \chi} [R^{\lambda}(\boldsymbol{u} \rightarrow \mathbf{F}_s\boldsymbol{u}) - \mathbf{I}] \cong \frac{1}{|\chi|} \sum_{\boldsymbol{u} \in \chi} [\lambda(R(\boldsymbol{u} \rightarrow \mathbf{F}_s\boldsymbol{u}) - \mathbf{I})], \tag{3-74}$$

$$= \mathbf{I} + \sum_{\boldsymbol{u} \in \chi} \left(\frac{\lambda}{|\chi|} (R(\boldsymbol{u} \rightarrow \mathbf{F}_s\boldsymbol{u}) - \mathbf{I}) \right) - \mathbf{I}, \tag{3-75}$$

$$\cong \prod_{\boldsymbol{u} \in \chi} \left(\mathbf{I} + \frac{\lambda}{|\chi|} (R(\boldsymbol{u} \rightarrow \mathbf{F}_s\boldsymbol{u}) - \mathbf{I}) \right) - \mathbf{I}, \tag{3-76}$$

$$\cong \left[\prod_{\boldsymbol{u} \in \chi} R^{\frac{\lambda}{|\chi|}}(\boldsymbol{u} \rightarrow \mathbf{F}_s\boldsymbol{u}) \right] - \mathbf{I}. \tag{3-77}$$

Relating (3-73) to (3-77), we find that

$$E[R^{\lambda}(\boldsymbol{u} \rightarrow \boldsymbol{v}) - \mathbf{I} \mid \mathbf{F}_s] \cong \left[\prod_{\boldsymbol{u} \in \chi} R^{\frac{\lambda}{|\chi|}}(\boldsymbol{u} \rightarrow \mathbf{F}_s\boldsymbol{u}) \right] - \mathbf{I}. \tag{3-78}$$

Hence, an $n \times n$ unitary matrix $\mathbf{F}_s$ is a *stable point* of a first-order MPLL if it satisfies the following approximate relationship:

$$\prod_{\boldsymbol{u} \in \chi} R^{\frac{\lambda}{|\chi|}}(\boldsymbol{u} \rightarrow \mathbf{F}_s\boldsymbol{u}) = \mathbf{I}, \tag{3-79}$$

which implies that the average geometric rotation produced by the rotation detector and loop filter together is the identity matrix for a stable point. We should point out that the identity matrix represents the case of zero rotation.

An example of typical unitary matrices that satisfy either (3-73) or (3-79) is given below:

Example 3-2. Two stable points for a first-order MPLL are given by the following matrices:

$$\mathbf{F}_s = \mathbf{P} \text{ and } \mathbf{F}_s = \mathbf{P}\begin{bmatrix} e^{j\theta} & 0 \\ 0 & e^{-j\theta} \end{bmatrix}, \tag{3-80}$$

where $\mathbf{P}$ is a complex permutation matrix and $\theta = 25.43°$ was found through computer simulations. The first stable point given in (3-80) is desirable because it implies that the channel input vector has been recovered up to a permutation of the users and up to a rotation of 90°. The last stable point is undesirable because it leaves a non-zero residual rotation error in the MPLL. This rotation error implies that the channel input vector cannot be recovered with a simple decision device.

Definition 3-6. The stable point of $\mathbf{F}_s = \mathbf{P}$ is a *desirable stable point*. Any other stable point is *undesirable*.

It is seen that the definition for the stable point of a first-order MPLL is dependent on the magnitude of the step size λ. For example, the second stable point given in the previous

example satisfies (3-73) when $\lambda = 0.001$, but it does not when $\lambda = 0.1$. The only stable point that satisfies (3-73) for all step sizes is $\mathbf{F}_s = \mathbf{P}$.

Unlike the S-curve for the PLL, the MS-curve for the MPLL defined in (3-71) is not as useful for several reasons. First, the zero points of the function are dependent upon the value of the step size. Second, we know of no graphical approach that would enable us to display the results of the MS-curve. Finally, it is almost impossible to generate an MS-curve, because it would require calculating the function for a set of infinite $n \times n$ unitary matrices. In spite of these limitations which will not allow us to determine all of the stable points for a first-order MPLL, the MS-curve is important because it predicts that a first-order MPLL has both desirable and undesirable stable points.

For a first-order MPLL, it is important to know whether the MPLL can converge to an undesirable stable point. In Section 3.1.2, it was seen that a first-order PLL converges to an undesirable stable point only when the step size is small. We believe this result to be true for the MPLL as well. In order to determine the conditions that guarantee convergence for a first-order MPLL, we consider a set of experiments, given below, that examines the effect of the step size on convergence of a first-order MPLL.

We begin by examining a first-order MPLL with a 16-QAM input alphabet. In the following experiments, we vary both the step size λ and the number of users n.

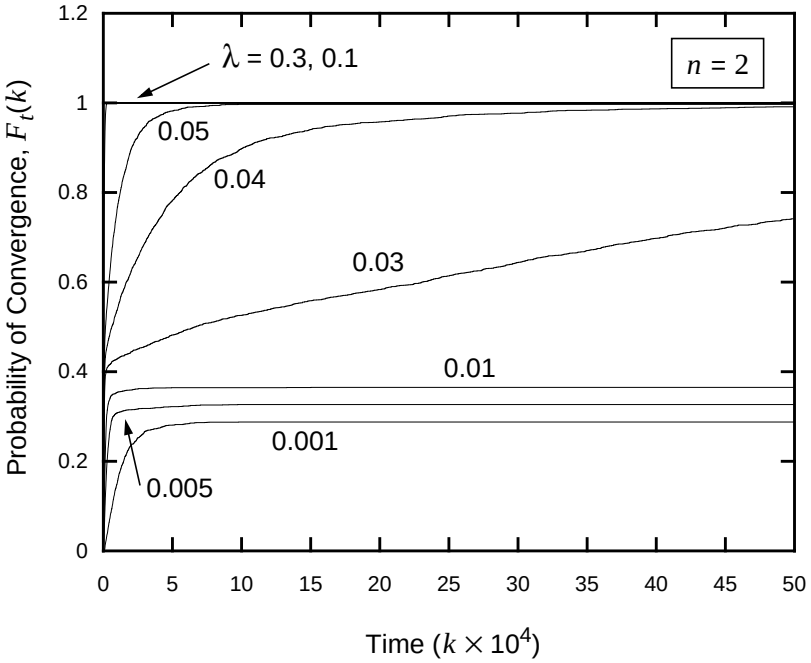
> **Experiment 3-8.** Consider a two-user ($n = 2$) MPLL. Suppose that the 2×1 input signal to the MPLL is given by $\boldsymbol{y}_k = \mathbf{W}\mathbf{H}\boldsymbol{x}_k$, where $\mathbf{H}$ is an $n \times n$ memoryless Gaussian channel matrix whose coefficients are drawn independently from a zero-mean, unit-variance, complex Gaussian distribution, and $\mathbf{W} = \Sigma^{-}$

$^{1}\mathbf{V}^{*}$ is an $n \times n$ ideal whitening matrix that is defined by the singular-value decomposition of $\mathbf{H} = \mathbf{U}\Sigma\mathbf{V}^{*}$. The cascade of the whitening matrix and the memoryless channel matrix yields a unitary matrix $\mathbf{U} = \Sigma^{-1}\mathbf{V}^{*}\mathbf{H}$; hence, $\boldsymbol{y}_k = \mathbf{U}\boldsymbol{x}_k$. Each of the components of the 2×1 channel input vector $\boldsymbol{x}_k$ is drawn uniformly from a 16-QAM input alphabet. For a given unitary channel, we implement a first-order MPLL for 500,000 random symbols and for several different step sizes $\lambda \in \{0.3, 0.1, 0.05, 0.04, 0.03, 0.01, 0.005, 0.001\}$. For each step size, the minimum number of symbols that are required for the MPLL to converge to the desired stable point is determined. Convergence to a stable point is defined to have been achieved when the transfer function $\mathbf{F}_k = \hat{\mathbf{U}}_k{}^{*}\mathbf{U}$ satisfies:

$$\|\mathbf{F}_k - \mathbf{P}\|_F^2 \leq n10^{-3}, \tag{3-81}$$

where $\mathbf{P}$ is a complex permutation matrix that accounts for the inherent ambiguity associated with a blind detection problem and a $\frac{\pi}{2}$-symmetric constellation. In all, we considered 3000 different unitary matrices. In Fig. 3-14, $F_t(k)$, the fraction of trials that converged within k symbols, is plotted versus the number of the symbols. From these curves, it is observed that the MPLL always converges within 500,000 symbols to the desired stable point when the step size $\lambda \geq 0.04$. On the other hand, if $\lambda \leq 0.01$, the MPLL only converges for a small fraction of the 3000 trials; in fact, for majority of the phase offsets, the MPLL remains trapped around an undesirable stable point even after 500,000 symbols.

(a)



(b)

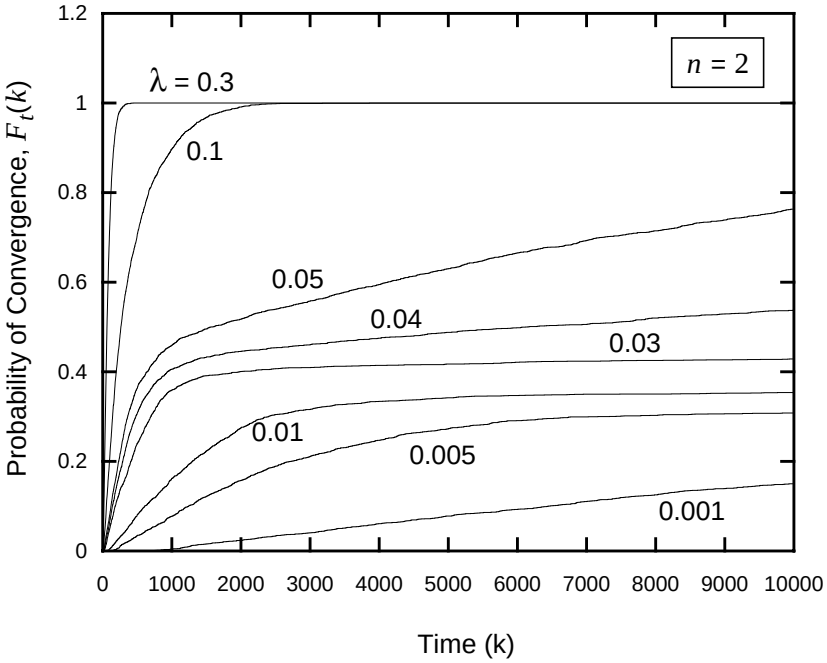


Fig. 3-14. The fraction of trials that converge within k symbols is plotted versus the number of symbols for a noiseless two-user first-order MPLL with 16-QAM input alphabet and various step sizes: (a) the number of symbols ranges from 0 to 500,000; (b) an expanded view of the first 10,000 symbols.

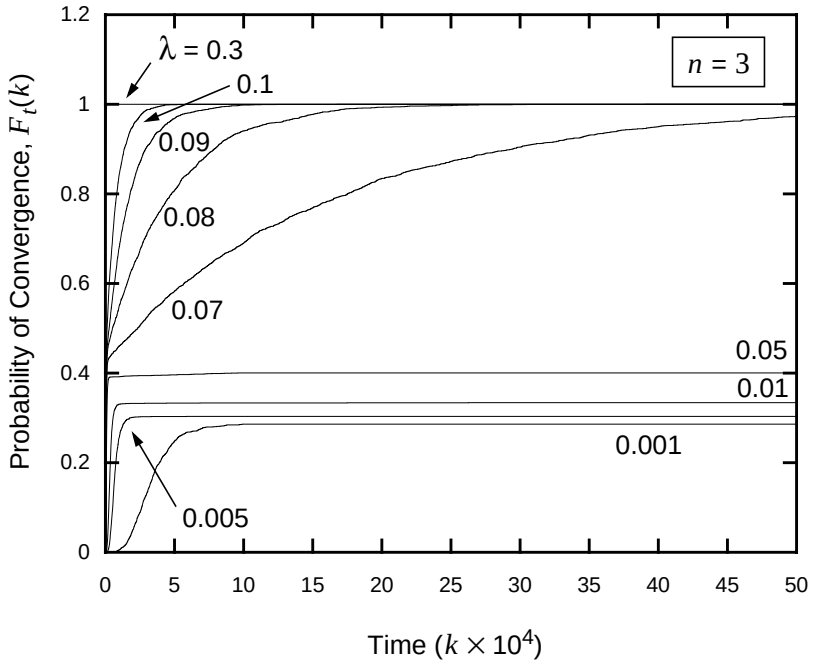
Experiment 3-9. In this experiment we essentially repeat Experiment 3-8 for a three-user (n = 3) MPLL and several different step sizes $\lambda \in \{0.3, 0.1, 0.09, 0.08, 0.07, 0.05\ 0.01, 0.005, 0.001\}$. In Fig. 3-15, $F_t(k)$ is plotted versus the number of the symbols. It is observed from these curves that the MPLL always converges to the desired stable point when the step size $\lambda \geq 0.075$. On the other hand, if $\lambda \leq 0.05$, the MPLL remains trapped around an undesirable stable point for the majority of the unitary matrices.

Experiment 3-10. Again, we repeat Experiment 3-8 for a four-user (n = 4) MPLL and several different step sizes $\lambda \in \{0.3, 0.11, 0.07, 0.05, 0.01, 0.005\}$. In Fig. 3-16, we plot $F_t(k)$ versus the number of the symbols. From these curves, we observe that the minimum step size that guarantees convergence within 500,000 symbols is given by $\lambda_{min} = 0.10$. If $\lambda \leq 0.06$, then it is possible for the MPLL to converge to an undesirable stable point.

These experiments confirm that it is possible for the MPLL to converge to an undesirable stable point for a sufficiently small step size. We do not know if the MPLL will ever escape the undesirable stable point even for an infinite number of symbols. Since the speed of convergence is of utmost importance in a practical implementation, the result would only be of an academic interest.

These experiments also indicate that the minimum step size λ_{min} that guarantees convergence within 500,000 symbols increases as a function of the number of users n. Since $\lambda \in (0,1)$, λ_{min} must eventually approach a constant value as n becomes large.

(a)



(b)

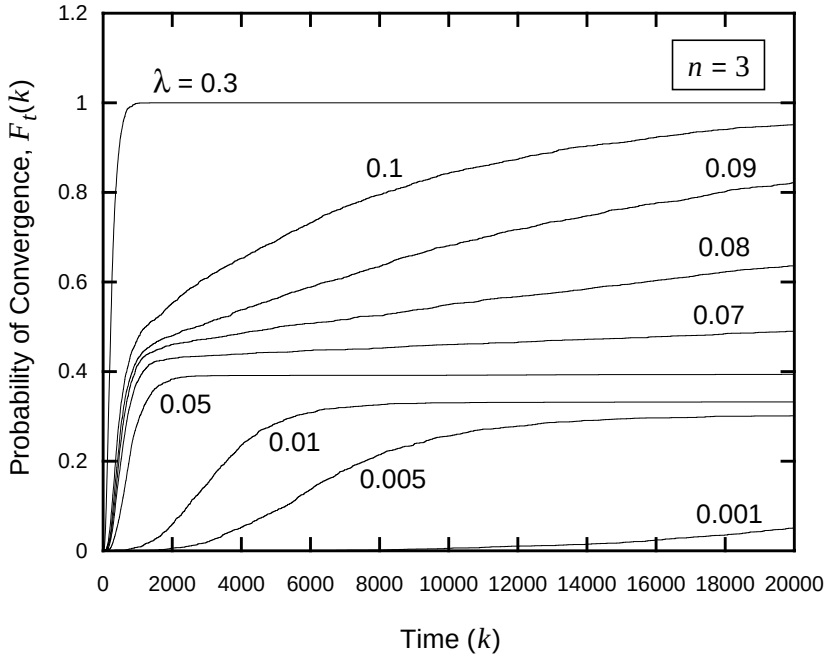
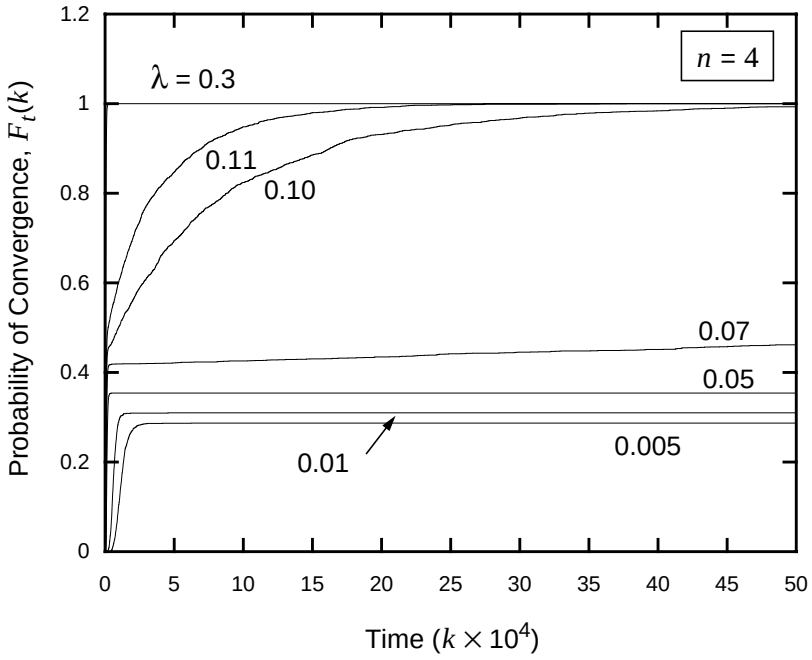


Fig. 3-15. The fraction of trials that converge within k symbols is plotted versus the number of symbols for a noiseless three-user first-order MPLL with 16-QAM input alphabet and various step sizes: (a) the number of symbols ranges from 0 to 500,000; (b) an expanded view of the first 20,000 symbols.

(a)



(b)

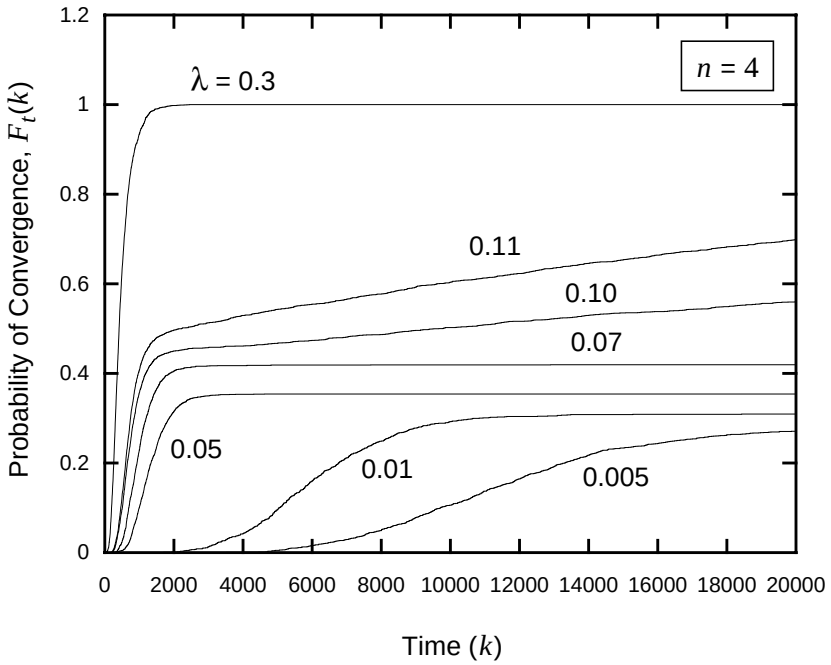


Fig. 3-16. The fraction of trials that converge within k symbols is plotted versus the number of symbols for a noiseless four-user first-order MPLL with 16-QAM input alphabet and various step sizes: (a) the number of symbols ranges from 0 to 500,000; (b) an expanded view of the first 20,000 symbols.

Experiment 3-11. In Fig. 3-17, we plot λ_{min} versus n. The values of $\lambda_{min}$for $n \in \{6, 8, 10, 12\}$ were found in a manner similar to that in Experiments 3-8 – 3-10. This curve shows that, as n becomes large, λ_{min} does indeed start to approach a constant value.

In summary, these experiments verify that there exists a minimum step size that would guarantee convergence of a first-order MPLL within 500,000 symbols.

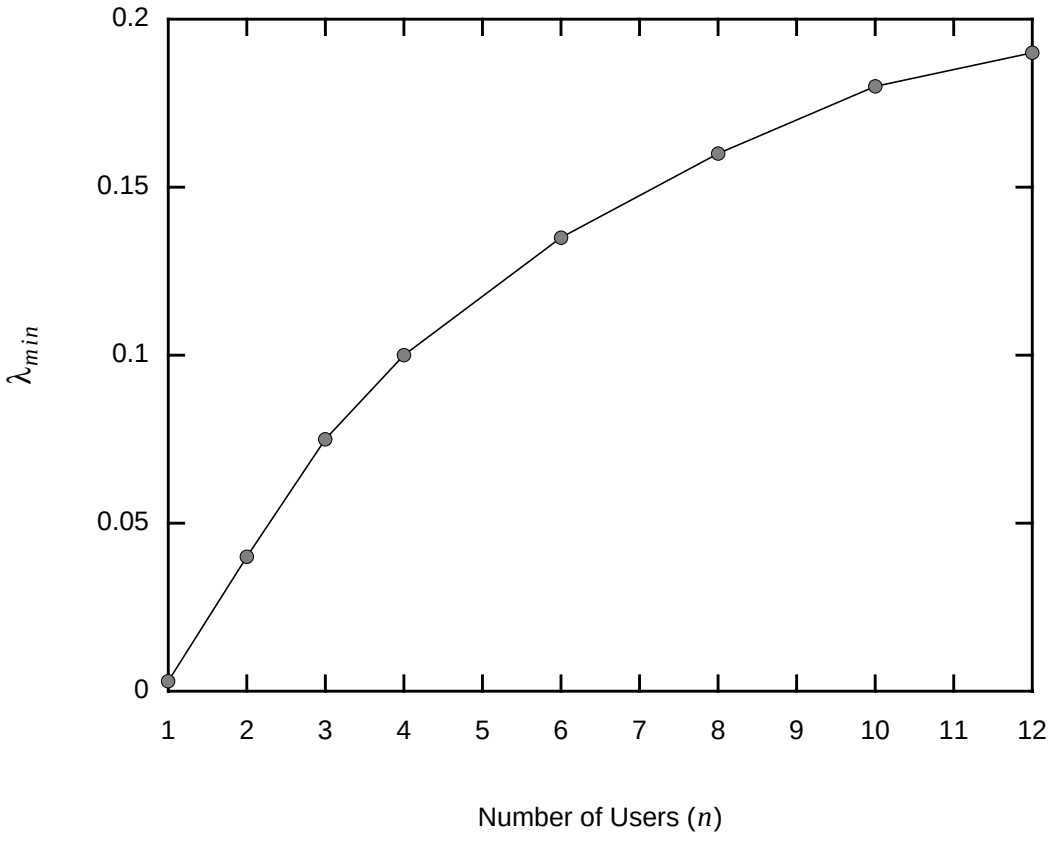


Fig. 3-17. The minimum step size that guarantees convergence within 500,000 symbols λ_{min} is plotted versus the number of users n.

3.5.2 In the Presence of Noise

So far we have only considered the convergence of a first-order MPLL in the absence of noise. We now examine the effect of noise on convergence. We begin by assuming that the noise vector $\boldsymbol{n}_k$ in (3-30) is a complex Gaussian random vector with power spectral density $\sigma^2\mathbf{I}$. If the noise term is non-zero, the received signal $\boldsymbol{z}_k$ can written as follows:

$$\boldsymbol{z}_k = \mathbf{F}_k\boldsymbol{z}_k + \boldsymbol{w}_k, \tag{3-82}$$

where $\boldsymbol{w}_k = \hat{\mathbf{U}}_k^*\boldsymbol{n}_k$ is also a complex Gaussian random vector with power spectral density $\sigma^2\mathbf{I}$.

As before, the MS-curve is not of much use for determining the noisy stable points of a first-order MPLL, but we can show that $\mathbf{F}_s = \mathbf{P}$, where $\mathbf{P}$ is a complex permutation matrix, is a noisy stable point of the algorithm. We need to investigate here if there are any other stable points. For the PLL, we have shown that as the SNR decreases, the undesirable stable points disappear, leaving the desired stable point as the only remaining stable point. We believe this result to be true for the MPLL as well. We describe the following experiment to support our intuition. We examine a first-order MPLL with a 16-QAM input alphabet.

> **Experiment 3-12.** Consider a three-user MPLL. Suppose that the 3×1 input signal to the MPLL is given by $\boldsymbol{y}_k = \mathbf{U}\boldsymbol{x}_k + \boldsymbol{n}_k$, where $\mathbf{U}$ is a randomly generated 3×3 unitary matrix, $\boldsymbol{x}_k$ is the 3×1 channel input vector whose components are drawn uniformly from a 16-QAM input alphabet, and $\boldsymbol{n}_k$ is a zero-mean white complex Gaussian noise vector with power spectral density

$E[\boldsymbol{n}_k \boldsymbol{n}_k^*] = \sigma^2 \mathbf{I}$. We assume that $\mathrm{SNR}_1 = 1/\sigma^2 = 20$ dB. For a given unitary channel, we implement a first-order MPLL for 100,000 random symbols and for several different step sizes $\lambda \in \{0.3, 0.1, 0.05, 0.01, 0.005, 0.001\}$. For each step size, the minimum number of symbols that are required for the PLL to converge to the desired stable point. Convergence to a stable point is defined to have been achieved when the transfer function $\mathbf{F}_k = \hat{\mathbf{U}}_k^* \mathbf{U}$ satisfies:

$$\|\mathbf{F}_k - \mathbf{P}\|_F^2 \leq n10^{-3}, \tag{3-83}$$

where $\mathbf{P}$ is a complex permutation matrix that accounts for the inherent ambiguity associated with a blind detection problem and a $\frac{\pi}{2}$-symmetric constellation. In all, we considered 3000 different unitary matrices. In Fig. 3-18, $F_t(k)$, the fraction of trials that converged within k symbols, is plotted versus the number of the symbols. It is observed from these curves that the MPLL converges within 100,000 symbols to the desired stable point for all step sizes, even for those smaller step sizes for which the noiseless MPLL did not always converge.

These curves clearly show that a noisy signal can prevent a first-order MPLL from converging to an undesirable stable point, irrespective of the step size. In fact, the noise provides the necessary perturbation to enable the MPLL to escape from any undesirable stable point.

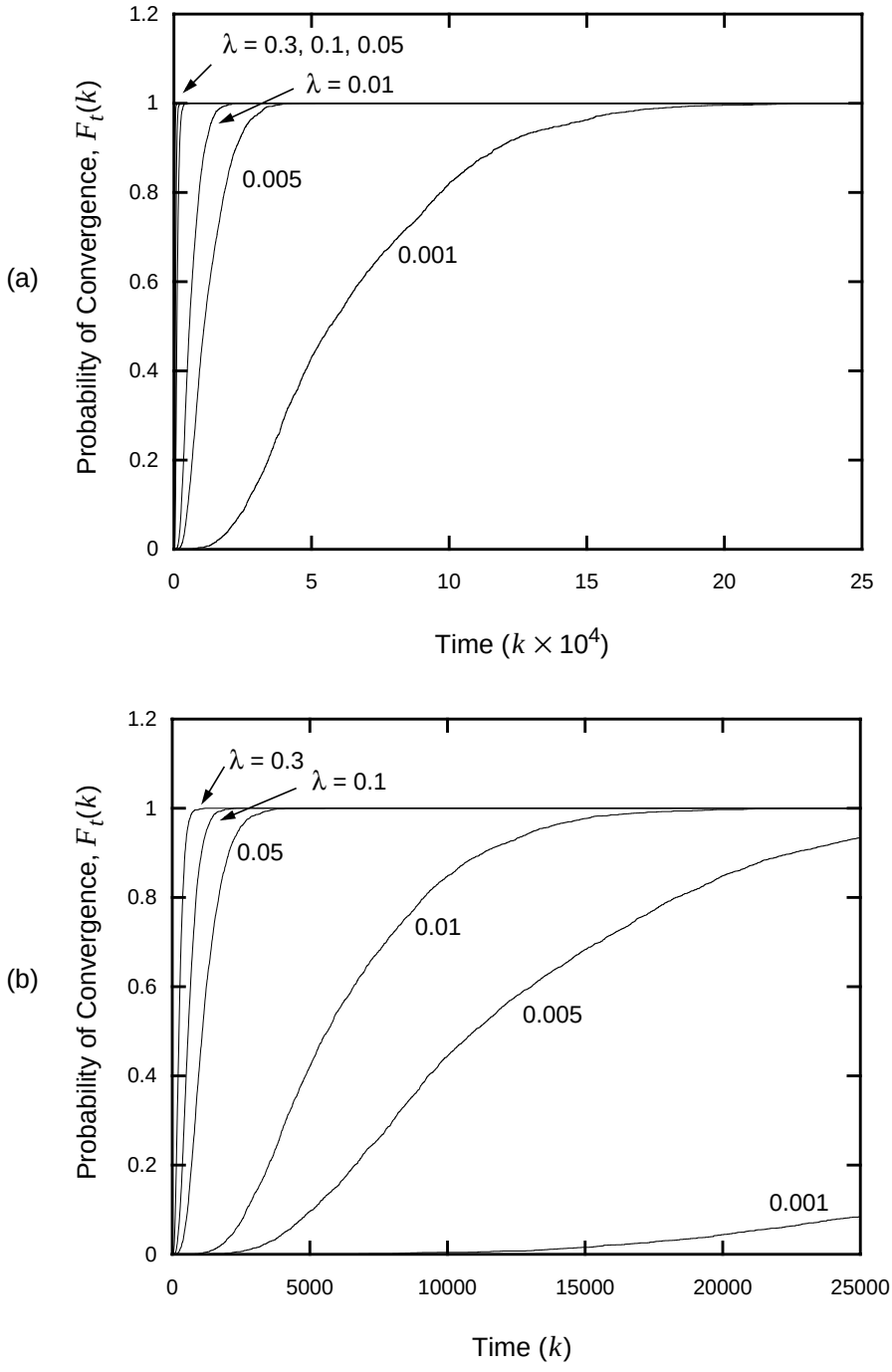


Fig. 3-18. The fraction of trials that converge within k symbols is plotted versus the number of symbols for a noisy three-user first-order MPLL with 16-QAM input alphabet, various step sizes, and SNR_1 = 20 dB: (a) the number of symbols ranges from 0 to 250,000; (b) an expanded view of the first 25,000 symbols.

3.6 CONVERGENCE ANALYSIS FOR A SECOND-ORDER MPLL

In this section, we study the convergence of a second-order MPLL in the absence of noise. We assume that each component of the channel input vector $\boldsymbol{x}_k$ is drawn uniformly from a discrete-input alphabet and that the only impairment in (3-30) is an angular rotation: $\mathbf{U}_k = \mathbf{U}\mathbf{W}^k$, where $\mathbf{U}$ is an $n \times n$ unitary matrix and $\mathbf{W}$ is an $n \times n$ diagonal unitary matrix. Typically, an angular rotation arises because of a difference between the frequencies of the n transmitter oscillators and that of a single receiver oscillator. The diagonal elements of $\mathbf{W}$ represent these frequency differences.

As before with the case for the second-order PLL, a theoretical analysis of the dynamics of a second-order MPLL is intractable. We can however, determine the range of frequency offsets that a second-order PLL could resolve experimentally. In the following experiment, we determine this frequency range for fixed values of λ_1 and λ_2.

Experiment 3-13. This experiment determines the ranges of frequencies that a two-user second-order MPLL is able to resolve. Suppose that the 2×1 input signal to the PLL is given by $\boldsymbol{y}_k = \mathbf{U}\mathbf{W}^k\boldsymbol{x}_k$, where $\mathbf{U}$ is a random 2×2 unitary matrix, $\mathbf{W} = diag(exp(j2\pi f_1), exp(j2\pi f_2))$ is a 2×2 diagonal unitary matrix with frequency offsets of f_1 and f_2, and $\boldsymbol{x}_k$ is the 2×1 channel input vector whose components are drawn uniformly from a 16-QAM input alphabet. For a given set of frequency offsets $f_1 = \frac{\varphi_1}{360}$ and $f_2 = \frac{\varphi_2}{360}$, where φ_1 and ϕ_2 are integers in the range [–25, 25], and a given random unitary channel, we implemented a second-order MPLL with parameters $\lambda_1 = 0.3$ and $\lambda_2 = 0.1$ for 500,000 symbols and determine whether or not the MPLL converged to the

desired stable point. Convergence to a stable point is defined to have been achieved when the transfer function $\mathbf{F}_k = \hat{\mathbf{U}}_k^* \mathbf{U}$ satisfies the following relationship for 15 consecutive symbols:

$$\|\mathbf{F}_k - \mathbf{P}\|_F^2 \leq n10^{-3}, \tag{3-84}$$

where $\mathbf{P}$ is a complex permutation matrix that accounts for the inherent ambiguity associated with a blind detection problem and a $\frac{\pi}{2}$-symmetric constellation. The shaded region in the plot of the first frequency offset f_1 versus the second frequency offset f_2, shown in Fig. 3-19, represents the values for which

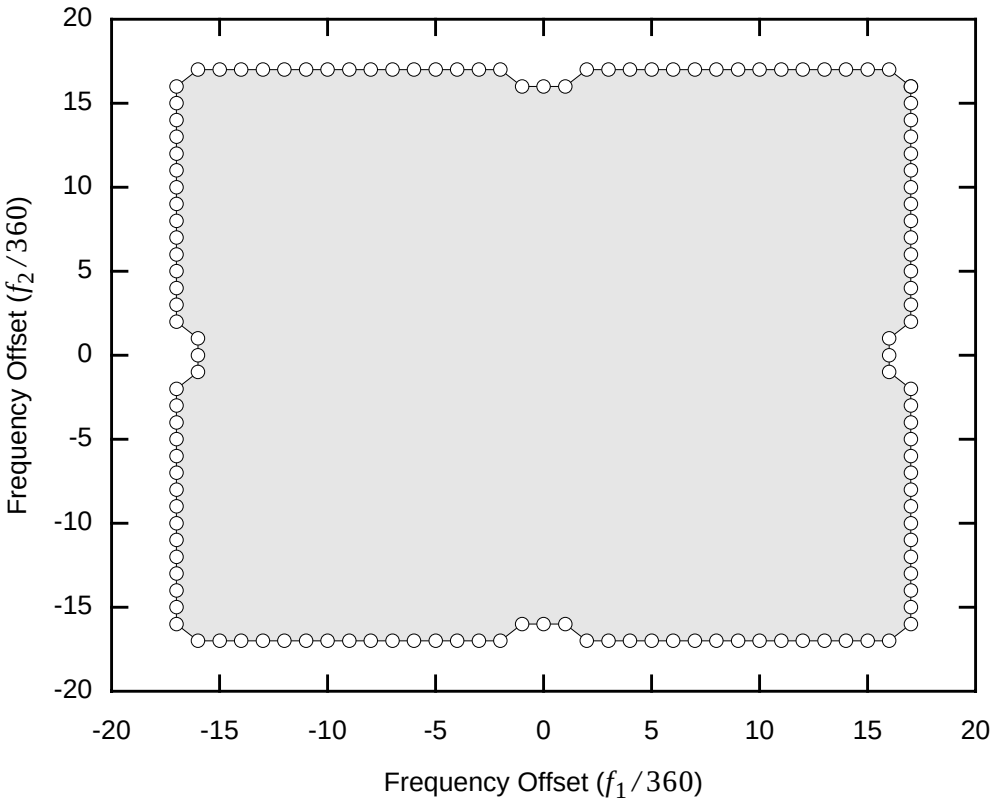


Fig. 3-19. A phase-plane portrait for a second-order MPLL with parameters $\lambda_1 = 0.3$ and $\lambda_2 = 0.1$.

the second-order DD MPLL converged. Each point in the plot was averaged over 1000 different random unitary channels. From this figure, we observe that a second-order MPLL with parameters $\lambda_1 = 0.3$ and $\lambda_2 = 0.1$ always converges to the desired stable point if $|f_i| \leq \frac{16}{360}\ \forall\ i$, regardless of the random unitary matrix. Therefore, a second-order MPLL can resolve a frequency offset of up to 16 degrees per baud.

This experiment shows, for $\lambda_1 = 0.3$ and $\lambda_2 = 0.1$, that it is possible for a second-order MPLL to resolve am angular rotation. In general, the range of frequency offsets that a second-order PLL can resolve will depend upon the choice of λ_1 and λ_2. As discussed earlier, a noisy input signal should assist in the convergence of a second-order MPLL.

3.7 Experimental Results

We conclude this chapter with several computer experiments. In each of the experiments, we consider the channel model depicted in Fig. 3-20:

$$\boldsymbol{y}_k = \mathbf{W}\mathbf{H}\boldsymbol{x}_k + \mathbf{W}\boldsymbol{n}_k, \tag{3-85}$$

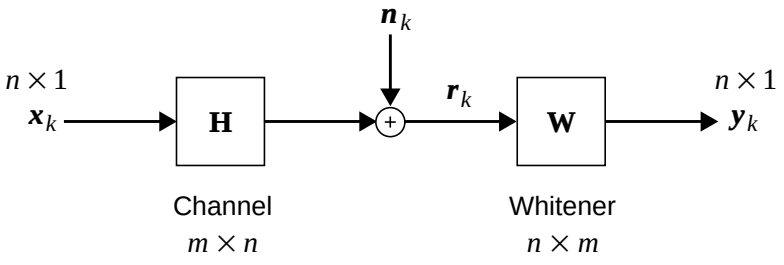


Fig. 3-20. A block diagram of an $m \times n$ memoryless channel followed by an $n \times m$ ideal whitener. This model is used to generate the input signal for all computer simulations.

where $\mathbf{H}$ is an $m \times n$ memoryless Gaussian channel matrix with coefficients that are drawn independently from a zero-mean, unit-variance, complex Gaussian distribution, and $\mathbf{W} = \mathbf{\Sigma}^{-1}\mathbf{U}^*$ is an $n \times m$ matrix that whitens the signal component of $\boldsymbol{y}_k$ and is defined by the truncated singular-value decomposition of $\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$. The cascade of the whitening matrix and the memoryless channel matrix yields a unitary matrix. The channel input vector $\boldsymbol{x}_k$ is an $n \times 1$ vector consisting of the symbols sent by the n independent users. The whitened vector $\boldsymbol{y}_k$ is an $n \times 1$ vector and $\boldsymbol{n}_k$ is an $m \times 1$ complex Gaussian noise vector with $E[\boldsymbol{n}_k \boldsymbol{n}_k^*] = \sigma^2\mathbf{I}$.

In these experiments, we compare the performance and complexity of the MPLL to that of JADE [47-49] and EASI (with $\mathbf{G}(\boldsymbol{z}) = \boldsymbol{z}\boldsymbol{z}^* - \mathbf{I} + g(\boldsymbol{z})\boldsymbol{z}^* - \boldsymbol{z}g(\boldsymbol{z})^*$ and $g(\boldsymbol{z}) = \boldsymbol{z} \otimes \boldsymbol{z}^* \otimes \boldsymbol{z}$, where $\otimes$ indicates a component-wise product) [50]. Unlike the MPLL, both JADE and EASI leave a residual phase error on each component of the detector output; in other words, both JADE and EASI are invariant to a diagonal unitary ambiguity. In practice, this ambiguity can be resolved by filtering the detector output through a bank of independent single-user PLLs. The performance criteria used in this comparison is the mean-squared error for the i-th user:

$$\mathrm{MSE}_i = E[\,|\,\bar{x}_k^{(i)} - z_k^{(i)}\,|^2], \tag{3-86}$$

where $\bar{\boldsymbol{x}}_k = \mathbf{P}\boldsymbol{x}_k$ is a permutation of the channel input vector and $\mathbf{P}$ is a complex permutation matrix (a matrix consisting of only one nonzero value from the set $\{1, j, -1, -j\}$ in each row and each column) that reorders the sources and rotates the phase of each source by a multiple of 90˚. The permutation is necessary to account for the inherent ambiguity that exists in any blind detection problem where the input alphabet is $\frac{\pi}{2}$-symmetric.

3.7.1 Noiseless Memoryless Unitary Channel

In the following two experiments, we consider the speed of convergence of JADE, EASI, and MPLL detectors in the absence of noise.

Experiment 3-14. Consider a 2×2 noiseless channel model described by (3-85), where $\boldsymbol{y}_k = \mathbf{U}\boldsymbol{x}_k$ and $\mathbf{U}$ is a 2×2 random unitary channel. We assume that each user draws symbols independently and uniformly from a 16-QAM input alphabet. In Fig. 3-21, we plot the MSE_1 versus time for JADE, EASI, and

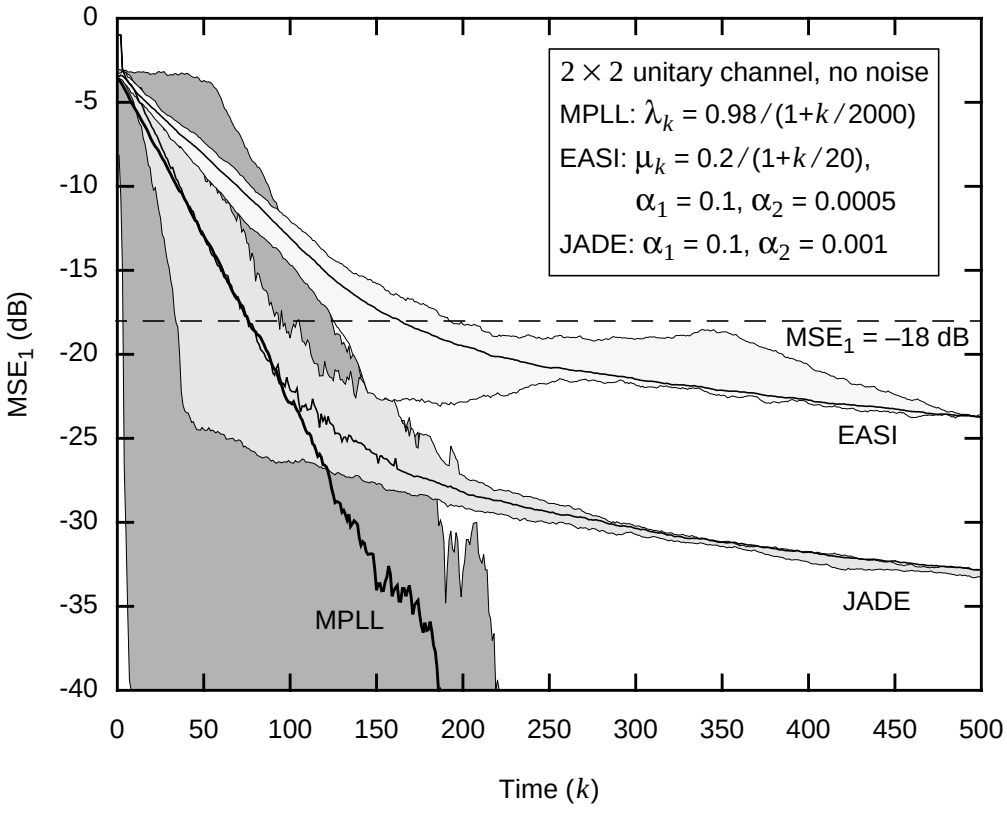


Fig. 3-21. Comparison of the MPLL, JADE, and EASI, in terms of MSE_1 versus time, for a 2×2 noiseless unitary channel. The shaded regions represent the variation in convergence time for each of the detector. For each detector, the lower curve is an average of the fastest 10% of the trials, the middle represents the average MSE, and the upper curve is an average of the slowest 10% of the trials.

MPLL detectors. Since the MSE is biased towards the worst-case performance, *i.e.*, towards the trials that converge the slowest, we display three curves for each detector: the middle curve represents the average over 5,000 random unitary channels and input realizations, the lower curve represents the average of the fastest 10% of the trials (best 10%), and the upper curve represents the average of the slowest 10% of the trials (worst 10%). The parameters for each detector were optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram, or equivalently an $\mathrm{MSE}_1 = -18$ dB. The step sizes for EASI and MPLL decrease with time according to $\mu_k = 0.2/(1 + k/20)$ and $\lambda_k = 0.98/(1 + k/2000)$, respectively. The parameters for the second-order PLLs were $\alpha_1 = 0.1$ and $\alpha_2 = 0.0005$ for EASI, and $\alpha_1 = 0.1$ and $\alpha_2 = 0.001$ for JADE. Fig. 3-21 shows that on average, both the MPLL and JADE converge much faster than EASI. We also observe that the worst 10% for the MPLL and JADE converge faster than the best 10% for EASI. Hence, we conclude for this channel that the MPLL and JADE clearly outperform EASI in terms of speed of convergence. One nice feature about both EASI and JADE is that they have near-uniform convergence, *i.e.*, there is very little difference between the best 10% and the worst 10%. In contrast, there is a large differences in the speed of convergence between the best 10% and the worst 10% for the MPLL. In fact, the best 10% can open the eye diagram in less than 4 symbols, while for the worst 10%, it takes slightly less than 75 symbols. Since the average MSE curve is very close to the curve for the worst 10%, we verify that the average MSE is dominated by the slowest trials. Finally, we should point out that the steady-state performance of the

MPLL is far superior to that of the other blind unitary estimators because of its decision-directed nature.

For each trial, we also calculated the convergence time of each detector. The *convergence time* is defined as the number of symbols it takes for a detector to reach an $\text{MSE}_1 \leq -18$ dB for 100 consecutive symbols. This value is a measure of how many symbols it takes to open the eye diagram. The average convergence time of the MPLL, JADE, and EASI are found to be 26.7, 55.6, and 237.3 symbols, respectively. Hence, we conclude that the MPLL is twice as fast as JADE and nine times as fast as EASI. From a histogram of the convergence time for each detector, plotted in Fig. 3-22, it is seen that the distribution for the MPLL and JADE are skewed towards fast convergence times, while the distribution for EASI is more evenly distributed about its mean. In fact, 96% trials for the MPLL converge within 75 symbols, while it takes JADE nearly 110 symbols and EASI almost 390 symbols to reach that same percentage. Hence, we conclude that, on average, the MPLL will converge much faster than either JADE or EASI.

This experiment shows that for a 2×2 noiseless unitary channel the MPLL is undoubtedly a better algorithm. In the next experiment, we consider a unitary channel with a slightly higher dimension.

Experiment 3-15. Consider a 3×3 noiseless channel model described by (3-85), where $\boldsymbol{y}_k = \mathbf{U}\boldsymbol{x}_k$ and $\mathbf{U}$ is a 3×3 random unitary channel. Again, we assume that each user draws symbols independently and uniformly from a 16-

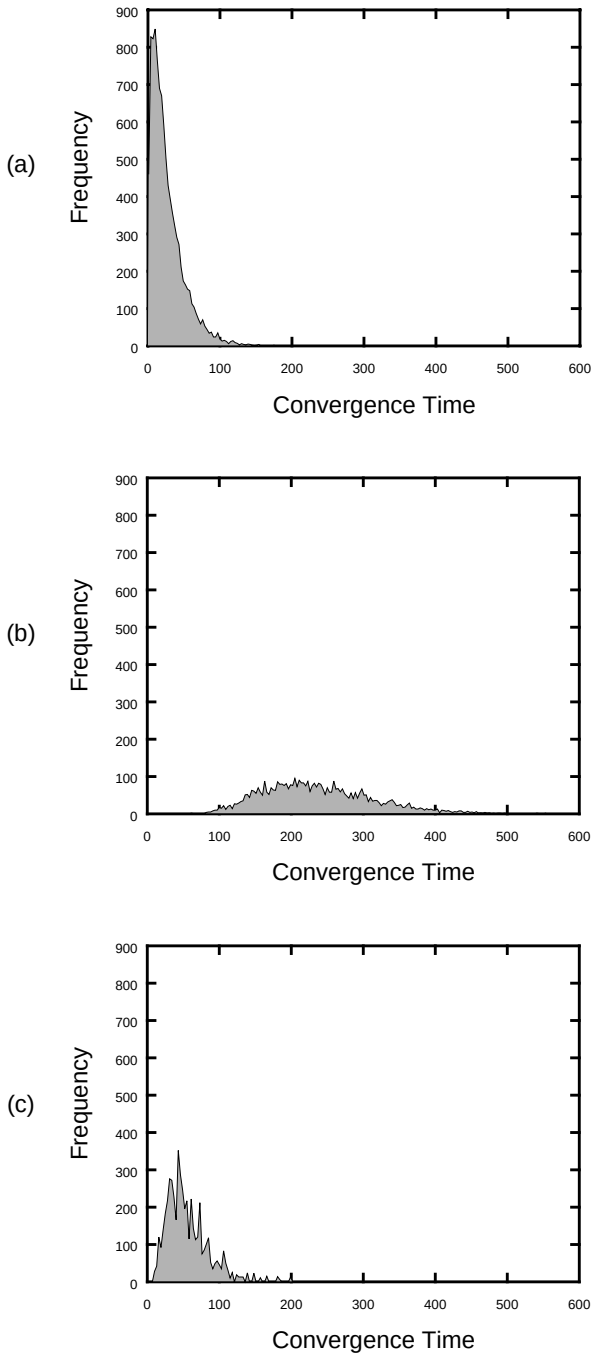


Fig. 3-22. Histogram of convergence times for a 2×2 unitary channel: (a) MPLL; (b) EASI; (c) JADE.

QAM input alphabet. In Fig. 3-23, we plot MSE_1 versus time for JADE, EASI, and MPLL. Each of the three curves is generated by averaging over 5,000 random unitary channels and input realizations. The parameters for each algorithm were again optimized for fast convergence. The step sizes for EASI and MPLL decrease with time according to $\mu_k = 0.1/(1 + k/50)$ and $\lambda_k = 0.66/(1 + k/3000)$, respectively. The parameters for the second-order PLLs were $\alpha_1 = 0.1$ and $\alpha_2 = 0.0005$ for EASI, and $\alpha_1 = 0.1$ and $\alpha_2 = 0.001$ for JADE. From Fig. 3-23, it appears that, on average, both JADE and EASI can

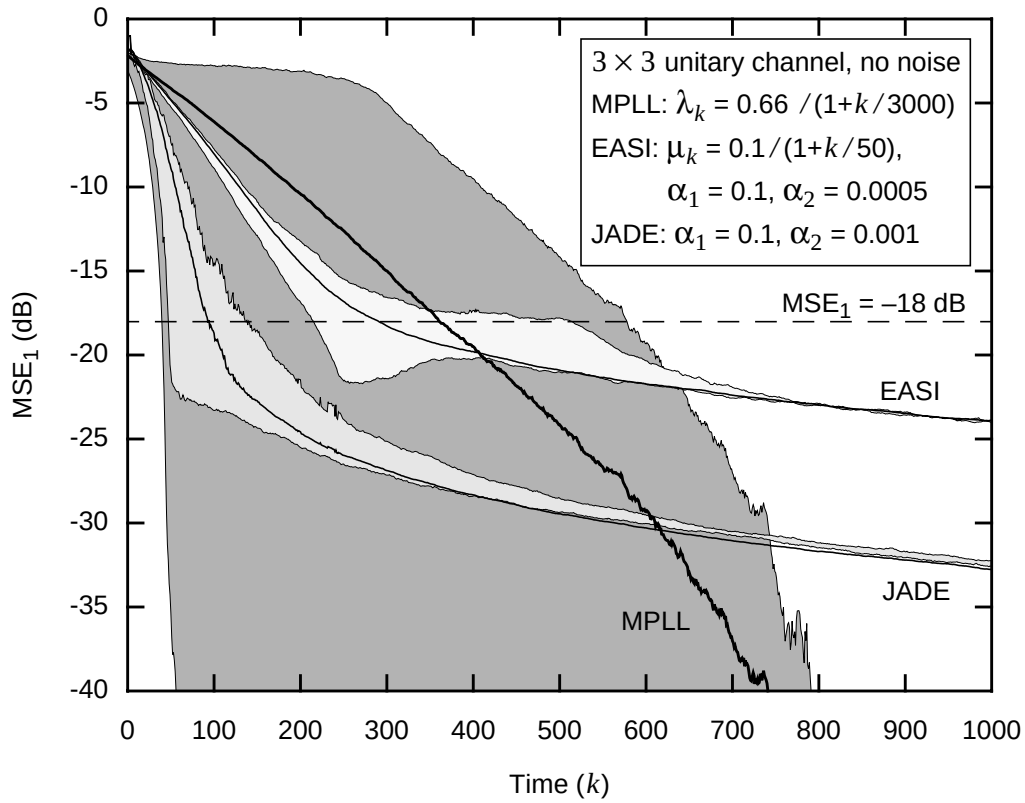


Fig. 3-23. Comparison of the MPLL, JADE, and EASI, in terms of MSE_1 versus time, for a 3×3 noiseless unitary channel. The shaded regions represent the variation in convergence time for each of the detector. For each detector, the lower curve is an average of the fastest 10% of the trials, the middle represents the average MSE, and the upper curve is an average of the slowest 10% of the trials.

open the eye diagram more quickly than the MPLL. Again, we reiterate that the average MSE curve is dominated by the slowest converging trials. The worst trials for the MPLL may converge more slowly than JADE and EASI, but the best 10% clearly outperform these detectors. From a histogram of the convergence time for each detector, plotted in Fig. 3-24, it is seen that the distribution for the MPLL and JADE are skewed towards fast convergence times, while EASI is more evenly distributed about its mean, indicating a more uniform rate of convergence. In fact, the MPLL is more likely to have a fast convergence time than a slow convergence time, but the average MSE curve, unfortunately, is dominated by the slow convergence times. The average convergence time of JADE, MPLL, and EASI are 85.0, 151.5, and 366.6 symbols, respectively. So even though, in Fig. 3-23, the MPLL appears to be slower than EASI, it is in fact more than two times faster. Again, it is important to note that the steady-state performance of the MPLL is far superior to that of JADE and EASI.

The speed of convergence of the MPLL is slower than that of JADE due to the fact that, at each iteration, it is only able to compensate for a rotation in two dimensions, whereas the unitary ambiguity represents a three-dimensional rotation.

3.7.2 Noisy Memoryless Gaussian Channel

In the following four experiments, we consider the effects of noise and the input alphabet on the speed of convergence of JADE, EASI, and the MPLL. We begin by looking at a 4-QAM input alphabet.

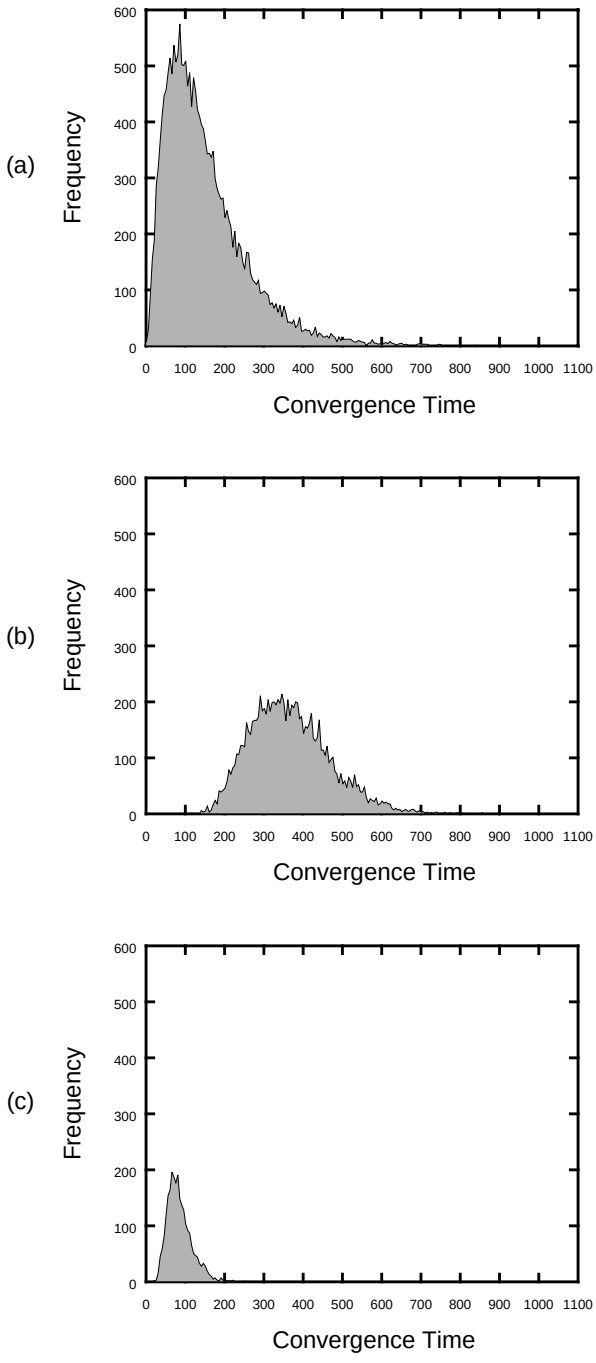


Fig. 3-24. Histogram of convergence times for a 3×3 unitary channel: (a) MPLL; (b) EASI; (c) JADE.

Experiment 3-16. Consider a 3×2 noiseless channel model described by (3-85). We assume that each user draws symbols independently and uniformly from a 4-QAM input alphabet. For a 4-QAM input alphabet, we define *convergence time* as the number of symbols it takes for each detector to reach an $\text{MSE}_1 \leq -12$ dB for 100 consecutive symbols. In Fig. 3-25, we plot the convergence time of each detector versus $\text{SNR}_1 = 1/\sigma^2$. Each curve was generated by averaging over 1000 different random unitary channels and input and noise realizations. The step sizes for EASI and MPLL decrease with time

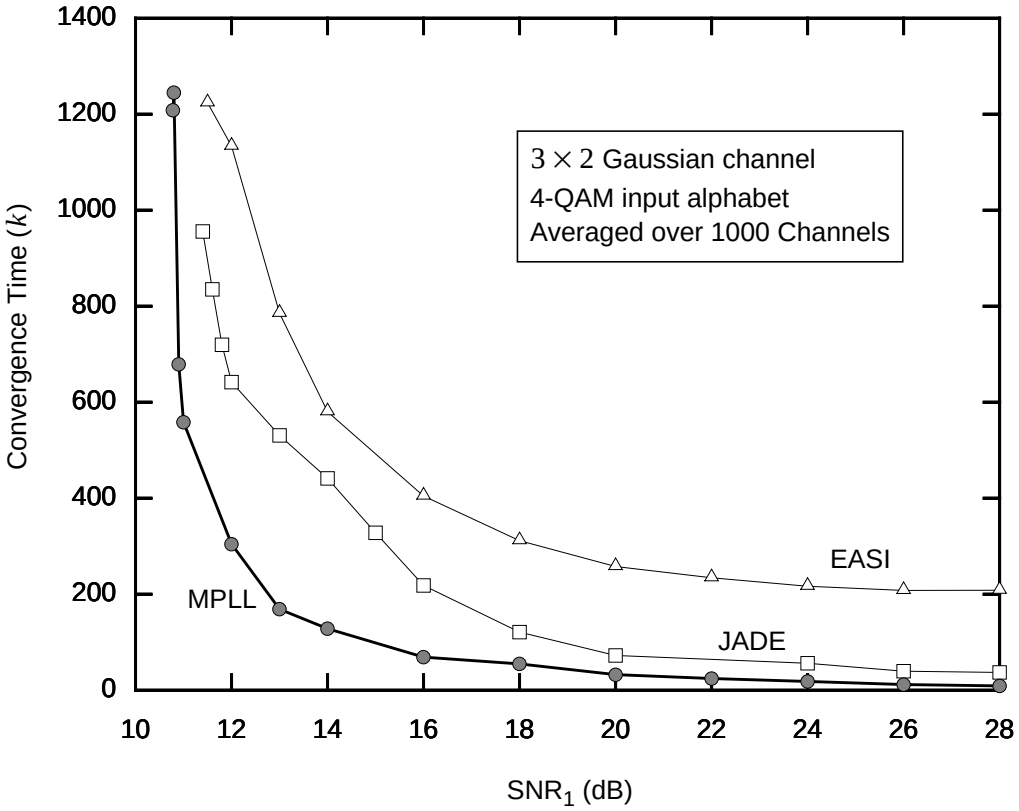


Fig. 3-25. The convergence time for each detector is plotted versus SNR_1 for a 3×2 noisy complex Gaussian channel followed by a 2×3 ideal whitener and a 4-QAM input alphabet. The optimal parameters for each detector can be found in Table 3-2.

according to $\mu_k = \mu_0/(1 + k/k_\mu)$and $\lambda_k = \lambda_0/(1 + k/k_\lambda)$, respectively. The single-user PLL bank, used by both JADE and EASI, has parameters α_1 and α_2. The parameters for each detector were optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram at each SNR; the values for the optimal parameters are listed in Table 3-2. These curves show that the MPLL provides the fastest convergence time, followed by JADE and then by EASI. At high SNR, the convergence time of the MPLL is only slightly better

TABLE 3-2: Optimal Parameters for a 3×2 Gaussian Channel with 4-QAM input alphabet.

SNR	MPLL	EASI $\alpha_1 = 10^{-2}$, $\alpha_2 = 10^{-4}$	JADE
10.78	$\lambda_0 = 0.98$, $k_\lambda = 3$	—	—
10.80 – 11.0	0.98, 3	—	—
11.4	—	—	$\alpha_1 = 10^{-2}$, $\alpha_2 = 10^{-4}$
11.5	0.98, 3	$\mu_0 = 0.05$, $k_\mu = 30$	—
11.6 – 11.8	—	—	10^{-2}, 10^{-4}
12.0	0.98, 3	0.05, 30	10^{-2}, 10^{-4}
13.0	0.98, 3	0.08, 20	10^{-2}, 10^{-4}
14.0	0.98, 5	0.08, 20	10^{-2}, 10^{-4}
16.0	0.98, 5	0.10, 20	10^{-2}, 10^{-4}
18.0	0.98, 5	0.10, 20	4×10^{-2}, 10^{-4}
20.0 – 22.0	0.98, 10	0.10, 20	10^{-1}, 10^{-3}
24.0	0.98, 20	0.10, 20	10^{-1}, 10^{-3}
26.0	0.98, 30	0.10, 20	10^{-1}, 10^{-3}
28.0	0.98, 40	0.10, 20	10^{-1}, 10^{-3}

than that of JADE. At low SNR, all three detectors start to break down and the convergence time becomes very large.

For a 4-QAM input alphabet, the MPLL clearly outperforms both JADE and EASI for all SNR. In the next experiment, we consider a 16-QAM input alphabet.

Experiment 3-17. Consider a 3×2 noiseless channel model described by (3-85). We assume that each user draws symbols independently and uniformly from a 16-QAM input alphabet. For a 16-QAM input alphabet, we define *convergence time* to be the number of symbols it takes for each detector to reach an $\text{MSE}_1 \leq -18$ dB for 100 consecutive symbols. In Fig. 3-26, we plot the convergence time of each detector versus $\text{SNR}_1 = 1/\sigma^2$. Each curve was generated by averaging over 1000 different random unitary channels and input and noise realizations. The parameters for each detector were optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram at each SNR; the values for the optimal parameters are listed in Table 3-3. These curves show that the MPLL provides the fastest convergence time, followed by JADE and then by EASI. At high SNR, the convergence time of the MPLL is only slightly better than that of JADE. At low SNR, JADE and EASI start to break down first, and the convergence time for all three detectors becomes very large.

Again, the MPLL clearly outperforms JADE and EASI for a 16-QAM input alphabet. In the next experiment, we demonstrate that the MPLL is compatible with shaped-input alphabets that are necessary for capacity achieving applications.

Experiment 3-18. Consider a 3×2 noiseless channel model described by (3-85). We assume that each user draws symbols independently and uniformly from a shaped 16-QAM input alphabet, which is defined as follows:

$$x_k^{(i)} = \begin{cases} \{\pm 1 \pm j\} & \text{each with probability } p(x_k) = 5/32 \\ \{\pm 3 \pm j, \pm 1 \pm 3j\} & \text{each with probability } p(x_k) = 1/32\,. \\ \{\pm 3 \pm 3j\} & \text{each with probability } p(x_k) = 1/32 \end{cases} \tag{3-87}$$

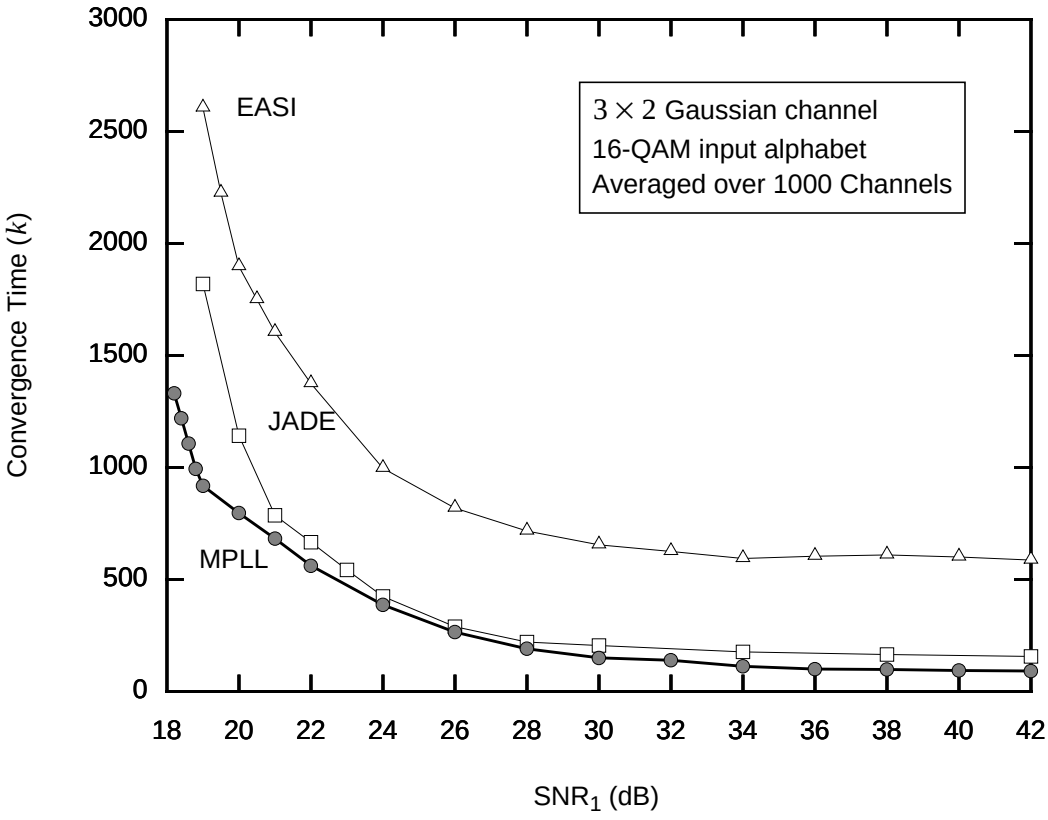


Fig. 3-26. The convergence time for each detector is plotted versus SNR_1 for a 3×2 noisy complex Gaussian channel followed by a 2×3 ideal whitener and a 16-QAM input alphabet. The optimal parameters for each detector can be found in Table 3-3.

It is easily verified that the kurtosis of this constellation is $\kappa = 1.89$, which is very close to the kurtosis of a complex Gaussian distribution. For a shaped 16-QAM input alphabet, we define *convergence time* to be the number of symbols it takes for each detector to reach an $\mathrm{MSE}_1 \leq -18$ dB for 100 consecutive symbols. In Fig. 3-27, we plot the convergence time of each detector versus $\mathrm{SNR}_1 = 1/\sigma^2$. Each curve was generated by averaging over 1000 different random

TABLE 3-3: Optimal Parameters for a 3×2 Gaussian Channel with 16-QAM input alphabet.

SNR	MPLL	EASI $\alpha_1 = 10^{-2}$, $\alpha_2 = 5\times10^{-4}$	JADE
18.2	$\lambda_0 = 0.98$, $k_\lambda = 25$	—	—
18.4 – 18.6	0.98, 28	—	—
18.8	0.98, 30	—	—
19.0	0.98, 30	$\mu_0 = 0.04$, $k_\mu = 40$	$\alpha_1 = 2\times10^{-2}$, $\alpha_2 = 10^{-4}$
19.5	—	0.04, 40	—
20.0 – 21.0	0.98, 30	0.04, 40	2×10^{-2}, 10^{-4}
22.0	0.98, 35	0.05, 40	2×10^{-2}, 10^{-4}
24.0	0.98, 45	0.05, 40	4×10^{-2}, 10^{-4}
26.0	0.98, 55	0.10, 20	8×10^{-2}, 5×10^{-4}
28.0	0.98, 70	0.10, 20	8×10^{-2}, 5×10^{-4}
30.0	0.98, 90	0.10, 20	8×10^{-2}, 5×10^{-4}
32.0	0.98, 100	0.10, 20	—
34.0	0.98, 120	0.10, 20	8×10^{-2}, 5×10^{-4}
36.0	0.98, 300	0.10, 20	—
38.0	0.98, 600	0.10, 20	8×10^{-2}, 5×10^{-4}
40.0	0.98, 800	0.10, 20	—
42.0	0.98, 1000	0.10, 20	10^{-1}, 10^{-3}

unitary channels and input and noise realizations. The step sizes for EASI and MPLL decrease with time according to $\mu_k = \mu_0/(1 + k/k_\mu)$and $\lambda_k = \lambda_0/(1 + k/k_\lambda)$, respectively. The single-user PLL bank, used by both JADE and EASI, has parameters α_1 and α_2. The parameters for each detector were optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram at each SNR; the values for optimal parameters are listed in Table 3-4. These curves show that the MPLL clearly outperforms both JADE and EASI. In fact, the convergence time for the MPLL with a shaped 16-QAM

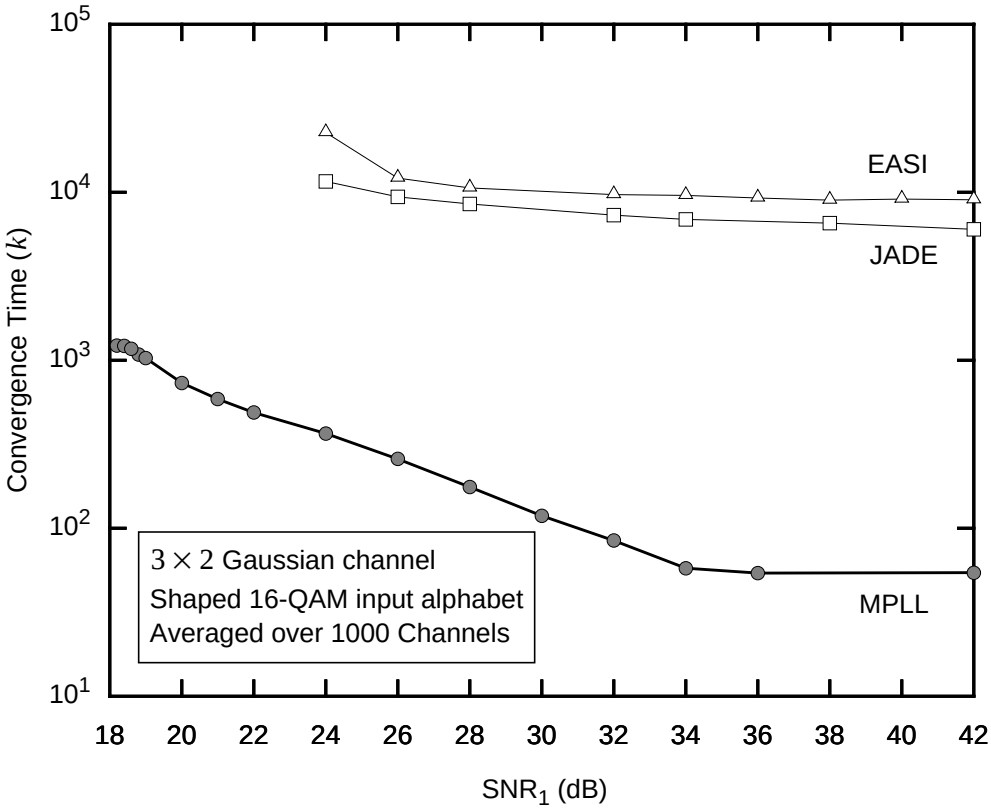


Fig. 3-27. The convergence time for each detector is plotted versus SNR_1 for a 3×2 noisy complex Gaussian channel followed by a 2×3 ideal whitener and a shaped 16-QAM input alphabet. The optimal parameters for each detector can be found in Table 3-4.

input alphabet is slightly better than that of the MPLL with a 16-QAM input alphabet. In contrast, the convergence times at high SNR for JADE and EASI are 6000 symbols and 9000 symbols, respectively. Thus for a shaped 16-QAM input alphabet, the MPLL is 150 times better than JADE and 200 times better than EASI.

This example demonstrates that the MPLL can resolve a unitary ambiguity for a shaped-input alphabet. In fact, as long as the input alphabet is discrete and the step size is

TABLE 3-4: Optimal Parameters for a 3×2 Gaussian Channel with shaped 16-QAM input alphabet.

SNR	MPLL	EASI $\alpha_1 = 10^{-2}$, $\alpha_2 = 5\times10^{-4}$	JADE
18.2	$\lambda_0 = 0.98$, $k_\lambda = 25$	—	—
18.4	0.98, 25	—	—
18.6	0.98, 28	—	—
18.8 – 22.0	0.98, 30	—	—
24.0	0.98, 45	$\mu_0 = 0.10$, $k_\mu = 12000$	$\alpha_1 = 10^{-2}$, $\alpha_2 = 10^{-4}$
26.0	0.98, 45	0.10, 8000	10^{-2}, 10^{-4}
28.0	0.98, 70	0.10, 6000	10^{-2}, 10^{-4}
30.0	0.98, 90	—	10^{-2}, 10^{-4}
32.0	0.98, 100	0.10, 4000	—
34.0	0.98, 120	0.10, 4000	10^{-2}, 10^{-4}
36.0	0.98, 300	0.10, 4000	—
38.0	0.98, 600	0.10, 4000	10^{-2}, 10^{-4}
40.0	0.98, 800	0.10, 4000	—
42.0	0.98, 1000	0.10, 4000	10^{-2}, 10^{-4}

chosen appropriately, the DD MPLL will be able to correctly resolve the unitary ambiguity. The convergence time of the MPLL for a shaped-input alphabet is found to be nearly the same as for a non-shaped input alphabet. In contrast, both JADE and EASI have large convergence times for a shaped-input alphabet, even for high values of SNR.

In the next experiment, we consider the effect that the number of users n has on the convergence time.

> **Experiment 3-19.** Consider a 5×3 noiseless channel model described by (3-85). We assume that each user draws symbols independently and uniformly from a 16-QAM input alphabet. For a 16-QAM input alphabet, we define *convergence time* to be the number of symbols it takes for each detector to reach an $\mathrm{MSE}_1 \leq -18$ dB for 100 consecutive symbols. In Fig. 3-28, we plot the convergence time of each detector versus $\mathrm{SNR}_1 = 1/\sigma^2$. Each curve was generated by averaging over 1000 different random unitary channels and input and noise realizations. The parameters for each detector were optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram at each SNR; the values for optimal parameters are listed in Table 3-5. These curves show that JADE provides the fastest convergence time, followed by the MPLL and then by EASI. At high SNR, JADE is two times faster than the MPLL and five times faster than EASI. At low SNR, all three detectors start to break down and the convergence becomes very large.

When $n > 2$, JADE provides faster convergence than either the MPLL or EASI. This result suggests that JADE should always be used in this case. This would indeed be true if

we were not concerned with the computational complexity of the detector. In the next section, we compute the computational complexity of each detector.

3.7.3 Complexity Comparison

In the following experiment, we calculate the complexity of each detector.

Experiment 3-20. Consider the channel model described by (3-30). In Fig. 3-29, we plot the number of floating point operations (FLOPS) required per iteration to generate an estimate of the rotation for each algorithm versus the

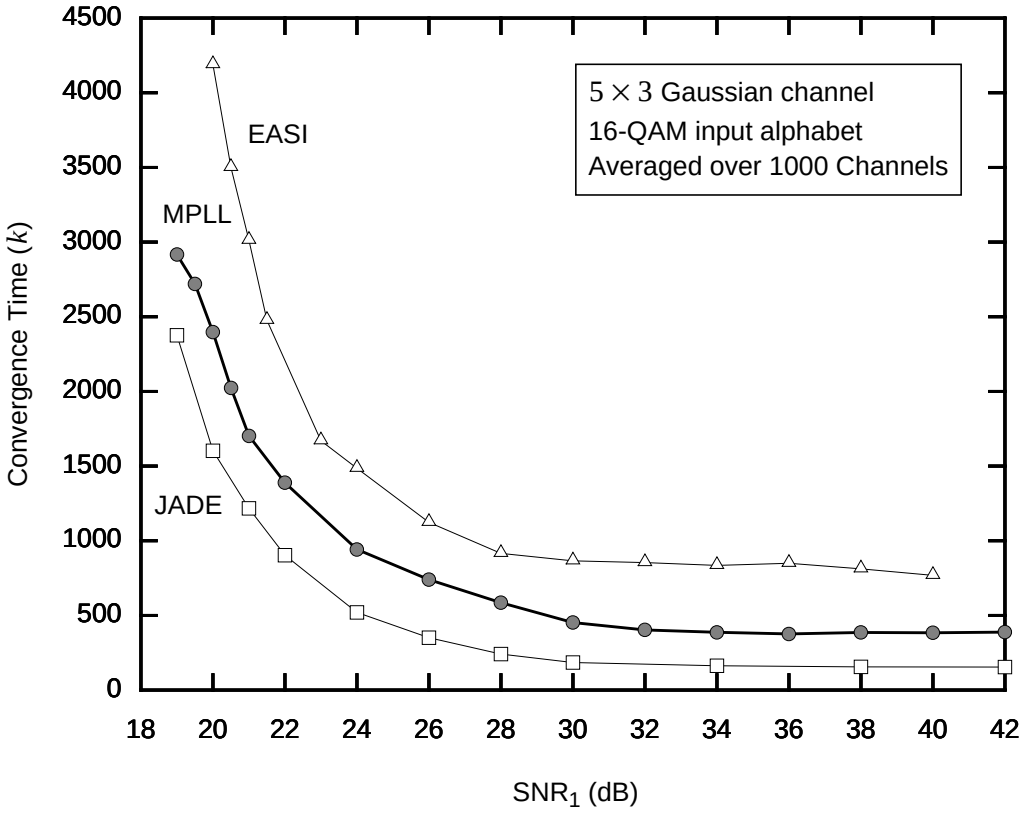


Fig. 3-28. The convergence time for each detector is plotted versus SNR_1 for a 5×3 noisy complex Gaussian channel followed by a 3×5 ideal whitener and a 16-QAM input alphabet. The optimal parameters for each detector can be found in Table 3-5.

number of users n. For both JADE and EASI, we took into account the complexity of the bank of independent single-user PLLs. These curves demonstrate the low computational complexity of the MPLL. Both the MPLL and EASI have essentially the same complexity $O(n^3)$, even though when $m = 25$, the complexity of EASI is six times greater than that of the MPLL. In contrast, the complexity of the JADE is extremely large, $O(n^6)$. It has been reported that it is possible to reduce the complexity of JADE to $O(n^5)$ [49].

The extremely large computational complexity of JADE makes this algorithm almost impractical in many real-world applications. When selecting a detector, both the computational complexity and the performance of the detector must be taken into consideration

TABLE 3-5: Optimal Parameters for a 5×3 Gaussian Channel with 16-QAM input alphabet.

SNR	MPLL	EASI $\alpha_1 = 10^{-2}$, $\alpha_2 = 5\times10^{-4}$	JADE
19.0	$\lambda_0 = 0.50$, $k_\lambda = 200$	—	$\alpha_1 = 2\times10^{-2}$, $\alpha_2 = 10^{-4}$
19.5	0.60, 200	—	—
20.0	0.60, 200	$\mu_0 = 0.10$, $k_\mu = 20$	2×10^{-2}, 10^{-4}
20.5	0.60, 200	0.10, 20	—
21.0 – 24.0	0.60, 200	0.10, 20	4×10^{-2}, 10^{-4}
26.0	0.66, 400	0.10, 20	4×10^{-2}, 10^{-4}
28.0	0.66, 400	0.10, 20	8×10^{-2}, 5×10^{-4}
30.0	0.66, 700	0.10, 20	8×10^{-2}, 5×10^{-4}
32.0	0.66, 1000	0.10, 20	—
34.0 – 42.0	0.66, 1000	0.10, 20	10^{-1}, 10^{-3}

and the trade-off must be carefully balanced. These experiments demonstrate that the MPLL provides such a balance and is therefore an excellent algorithm.

3.7.4 Trained MPLL versus Decision-Directed MPLL

In this chapter, we have focused our attention solely on the decision-directed MPLL. However, it is possible to use it also when a training sequence is available. The algorithm for a trained MPLL is the same as that of the decision-directed MPLL, except that the decision $\hat{\boldsymbol{x}}_k$ is replaced with the training sequence. In the following experiment, we com-

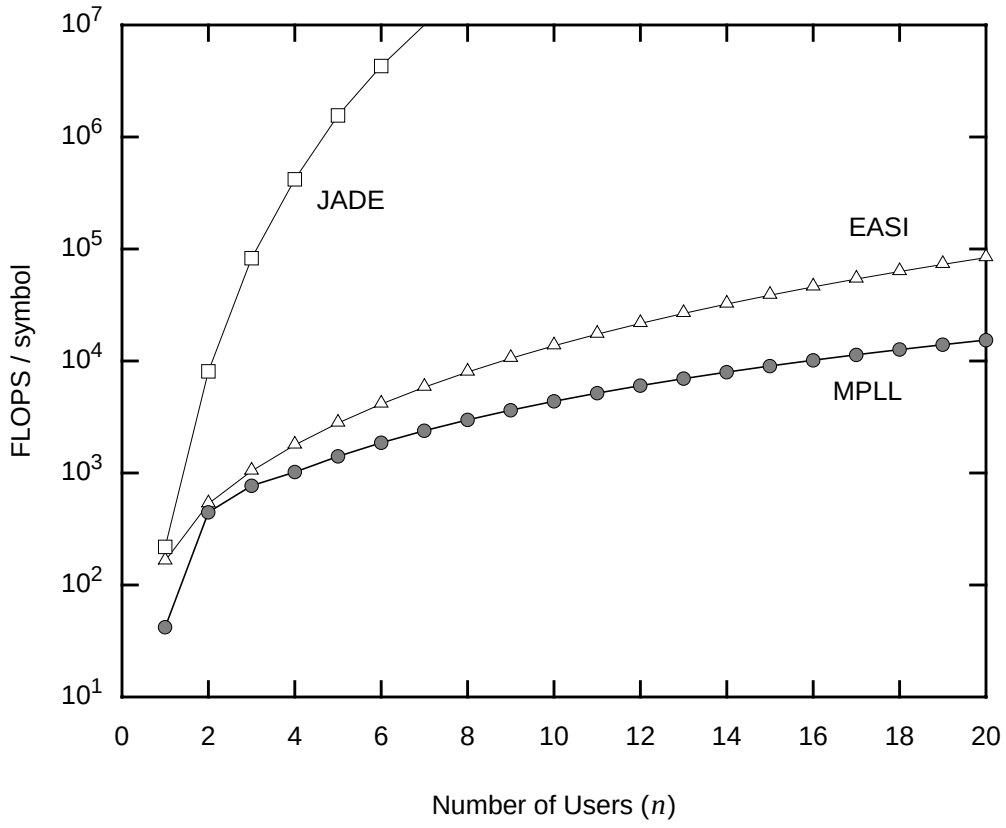


Fig. 3-29. Comparison of computational complexity of MPLL, JADE, and EASI.

pare the performance of a decision-directed MPLL with that of a trained MPLL, where the training sequence is the channel input vector $\boldsymbol{x}_k$.

Experiment 3-21. Consider a 2×2 noiseless channel model described by (3-85), where $\boldsymbol{y}_k = \mathbf{U}\boldsymbol{x}_k$ and $\mathbf{U}$ is a 2×2 random unitary channel. We assume that each user draws symbols independently and uniformly from a 16-QAM input alphabet. The step sizes for the trained MPLL and the decision-directed MPLL were the same: $\lambda = 0.8$. In Fig. 3-30, we plot the MSE_1 versus time for the DD MPLL and the trained MPLL. Each curve is an ensemble average over

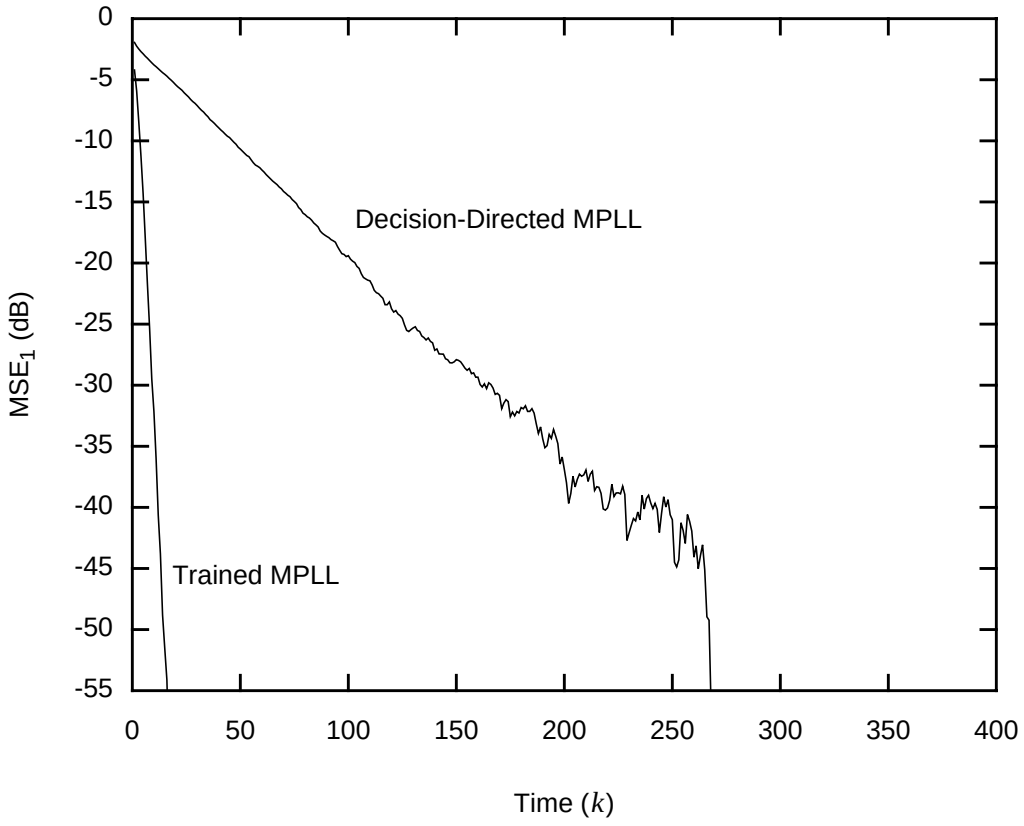


Fig. 3-30. Comparison of a trained MPLL and a decision-directed MPLL in terms of MSE_1 versus time.

10,000 random channels. We observe that the trained MPLL converges much more quickly than the decision-directed MPLL. We also see that both MPLLs can open the eye diagram within 100 symbols.

As expected, the performance of the trained MPLL is much better than that of the decision-directed MPLL, which is inherently a blind algorithm. Despite the fact that the decision-directed MPLL does not have access to a training sequence, it is still able to successfully open the eye diagram.

3.8 SUMMARY

In this chapter, we have reviewed the basic structure of a first-order and second-order phase-locked loop. In particular, we have analyzed the dynamics of these PLLs and have demonstrated that when the SNR is high and the step size is small, it is possible for a PLL to false lock, *i.e.*, to converge to an undesirable stable point. In that same analysis, we also showed that in a practical implementation of the PLL, there is a minimum step size that guarantees convergence to a desirable stable point. Unfortunately, the structure of the conventional PLL does not extend to multiple dimensions. However, we were able to manipulate the update equations and develop an alternative model for the PLL whose structure is shown in Fig. 3-7.

We have generalized the structure of the alternative-model PLL and developed the multidimensional phase-locked loop (MPLL), which is illustrated in Fig. 3-9. The MPLL is a decision-directed adaptive algorithm which exploits the discrete nature of digital communication signals. The key component of the MPLL is the rotation detector, which measures the rotation between two vectors. Using the rotation detector and extending the loop

filter to multiple dimensions, we are able to derive the update equations for both a first-order and a second-order MPLL. The computational complexity of the update equations for the MPLL can be rather large, because it requires raising a matrix to a fractional power which inherently demands an eigendecomposition. This computational burden was reduced by combining the operations of the rotation detector and the loop filter. These two components could be combined into two unique ways, one of which leads to an alternative structure for the MPLL.

Finally, we have analyzed the dynamics of both a first-order and a second-order MPLL. Since the MPLL is a generalization of the single-user PLL, it also subject to false lock when the SNR is high and the step size is small. But just as in the single-user case, the probability of false lock can be minimized through a careful choice of the step size. We have shown, through computer simulations, that the MPLL compares favorably, in terms of both performance and complexity, to both JADE and EASI. It was concluded that the MPLL offers fast convergence, excellent steady-state performance, and low complexity.

APPENDIX 3.1

PROOF OF THEOREM 3-5

For a given λ, the conventional-model MPLL rotates $\boldsymbol{u}$ to $\boldsymbol{v}_\lambda = R^\lambda(\boldsymbol{u} \rightarrow \boldsymbol{v})\boldsymbol{u}$, where $R^\lambda(\boldsymbol{u} \rightarrow \boldsymbol{v})$ is defined by (3-55) and (3-56). If each component of the $n \times 1$ channel input vector $\boldsymbol{x}$ draw symbols from a *real* discrete-input alphabet, then $R^\lambda(\boldsymbol{u} \rightarrow \boldsymbol{v})$ reduces to:

$$R^\lambda(\hat{\boldsymbol{x}}_k \rightarrow \boldsymbol{z}_k) = \mathbf{I}_n + \begin{bmatrix} \boldsymbol{u} & \boldsymbol{w} \end{bmatrix} \begin{bmatrix} cos(\lambda\Delta) - 1 & -sin(\lambda\Delta) \\ sin(\lambda\Delta) & cos(\lambda\Delta) - 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^T \\ \boldsymbol{w}^T \end{bmatrix}, \tag{3-88}$$

where $p = \boldsymbol{u}^T\boldsymbol{v} = cos(\Delta)$ and $\boldsymbol{w} = (\boldsymbol{v} - cos(\Delta)\boldsymbol{u})/sin(\Delta)$. (This analysis implicitly assumes that $|p| < 1$.) Using (3-88), we find that $\boldsymbol{v}_\lambda$ simplifies as follows:

$$\boldsymbol{z}_\lambda = \boldsymbol{u} + \begin{bmatrix} \boldsymbol{u} & \boldsymbol{w} \end{bmatrix} \begin{bmatrix} cos(\lambda\Delta) - 1 & -sin(\lambda\Delta) \\ sin(\lambda\Delta) & cos(\lambda\Delta) - 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^T \\ \boldsymbol{w}^T \end{bmatrix} \boldsymbol{u}, \tag{3-89}$$

$$= \boldsymbol{u} + \begin{bmatrix} \boldsymbol{u} & \boldsymbol{w} \end{bmatrix} \begin{bmatrix} cos(\lambda\Delta) - 1 & -sin(\lambda\Delta) \\ sin(\lambda\Delta) & cos(\lambda\Delta) - 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \tag{3-90}$$

$$= cos(\lambda\Delta)\boldsymbol{u} + sin(\lambda\Delta)\boldsymbol{w}, \tag{3-91}$$

$$= \frac{sin((1-\lambda)\Delta)}{sin(\Delta)}\boldsymbol{u} + \frac{sin(\lambda\Delta)}{sin(\Delta)}\boldsymbol{v}, \tag{3-92}$$

where the second equality arises from the fact that $\boldsymbol{w}$ is orthogonal to $\boldsymbol{u}$.

Similarly, for a given μ, the alternative-model MPLL rotates $\boldsymbol{u}$ to a fixed intermediate point $\boldsymbol{v}_\mu = \frac{\tilde{\boldsymbol{v}}_\mu}{\|\tilde{\boldsymbol{v}}_\mu\|}$, where $\tilde{\boldsymbol{v}}_\mu$ is defined by (3-65). We can expand $\boldsymbol{v}_\mu$ as:

$$\boldsymbol{v}_\mu = \frac{1-\mu}{\sqrt{\mu^2 + (1-\mu)^2 + 2\mu(1-\mu)\,p}}\boldsymbol{u} + \frac{\mu}{\sqrt{\mu^2 + (1-\mu)^2 + 2\mu(1-\mu)\,p}}\boldsymbol{v}. \tag{3-93}$$

Subtracting (3-93) from (3-92), we obtain:

$$\boldsymbol{v}_\lambda - \boldsymbol{v}_\mu = \left[\frac{sin((1-\lambda)\Delta)}{sin(\Delta)} - \frac{1-\mu}{\sqrt{\mu^2 + (1-\mu)^2 + 2\mu(1-\mu)\,p}}\right]\boldsymbol{u} +$$

$$\left[\frac{sin(\lambda\Delta)}{sin(\Delta)} - \frac{\mu}{\sqrt{\mu^2 + (1-\mu)^2 + 2\mu(1-\mu)\,p}}\right]\boldsymbol{v}. \tag{3-94}$$

One way that the two intermediate points $\boldsymbol{v}_\lambda$ and $\boldsymbol{v}_\mu$ can be the same is if both terms inside the square brackets are zero, or in other words, if both of the following equations are satisfied:

The only way that the two intermediate points can be the same is if both terms inside the square brackets are zero; in other words, if both of the following equations are satisfied:

$$\frac{sin((1-\lambda)\Delta)}{sin(\Delta)} = \frac{1-\mu}{\sqrt{\mu^2 + (1-\mu)^2 + 2\mu(1-\mu)\,p}}, \tag{3-95}$$

$$\frac{sin(\lambda\Delta)}{sin(\Delta)} = \frac{\mu}{\sqrt{\mu^2 + (1-\mu)^2 + 2\mu(1-\mu)\,p}}. \tag{3-96}$$

Dividing (3-96) by (3-95) and solving for μ, we find that, for a given λ, μ must satisfy the following equation in order for the two intermediate points to be the same:

$$\mu = \frac{sin(\lambda\Delta)}{sin(\lambda\Delta) + sin((1-\lambda)\Delta)}. \tag{3-97}$$

The only way that the two intermediate points can be the same is if the partial rotation matrix for the conventional-model and alternative-model MPLL are the same. ❑

CHAPTER 4

SPATIAL VECTOR CMA

The constant-modulus algorithm (CMA) is the most commonly implemented and studied blind equalization algorithm for single-user communication systems [79]. Several generalizations of CMA to the multiuser framework have been proposed over the last fifteen years. To our knowledge, the most common extension is the pointwise constant-modulus algorithm, which forces each component of the detector output vector to have a constant modulus. The primary drawback of this algorithm is its susceptibility to the one-to-many problem, whereby, the detector recovers the same user more than once. Most modifications of pointwise CMA that have been proposed to eliminate the one-to-many problem add an additional term to pointwise CMA cost function [100-102].

In contrast, we propose a new generalization of CMA to the multiuser framework. Instead of imposing the constant modulus constraint on each component of the detector output, we suggest a new cost function that imposes a constant modulus constraint on the entire vector-valued detector output. We will show that this cost function has properties similar to that of the scalar CMA cost function and that the resulting algorithm fits in with the general whiten-rotate structure described in Chapter 1.

In this chapter, we limit the focus of the research to a memoryless channel. The insight gained for a memoryless channel will prove valuable when extending the results to a channel with memory in Chapter 5. The results presented in this chapter are also important in their own right, because many real-world applications, such as a synchronous CDMA system and a narrowband array-processing system, can be modeled by a memoryless channel. In the following chapter, we extend the concepts and algorithms presented in this chapter to channels with memory.

In Section 4.1, we introduce the channel model and assumptions that will be used throughout the remainder of this chapter. In Section 4.2, we introduce the vector constant-modulus algorithm cost function. In Section 4.3, we determine the local minima of the cost function in the absence of noise. We show that for certain input alphabets, the cost function is minimized only by desirable local minima, while for other input alphabets, the cost function is minimized by both desirable and undesirable local minima. In Section 4.4, we consider the effects of noise on local minima of the cost function. In Section 4.5, we propose a modification to the vector CMA cost function, which can eliminate the undesirable local minima for all input alphabets. We derive a stochastic gradient-descent algorithm for both cost functions in Section 4.6. Finally, in Section 4.7, we present several simulation results which demonstrate the effectiveness, in terms of speed of convergence and complexity, of the two proposed algorithms.

4.1 CHANNEL MODEL AND ASSUMPTIONS

Consider the memoryless channel model depicted in Fig. 4-1:

$$\boldsymbol{r}_k = \mathbf{H}\boldsymbol{x}_k + \boldsymbol{n}_k, \tag{4-1}$$

where $\mathbf{H}$ is an $m \times n$ memoryless channel matrix. This memoryless channel arises in a wide-variety of real-world applications, including a narrowband m-sensor uniform linear-array application, where the columns of the $\mathbf{H}$ represents the steering vectors for the n users, and in a synchronous CDMA application, where the columns of $\mathbf{H}$ represent the m-chip length signature sequences of the n users. The transmitted vector $\boldsymbol{x}_k$ is an $n \times 1$ vector consisting of the symbols sent by the n independent users. The received vector $\boldsymbol{r}_k$ is an $m \times 1$ vector composed of the receiver observations, while $\boldsymbol{n}_k$ represents an $m \times 1$ noise vector with power spectral density $E[\boldsymbol{n}_k \boldsymbol{n}_k^*] = \sigma^2\mathbf{I}$, with $\sigma^2 > 0$.

We assume that channel $\mathbf{H}$ has full-column rank ($rank(\mathbf{H}) = n$), which implies that the channel is either square or tall ($m \geq n$), and that the channel input vector $\boldsymbol{x}_k$ can be recovered using a linear detector (see Section 2.1.2). We further assume that the signal and

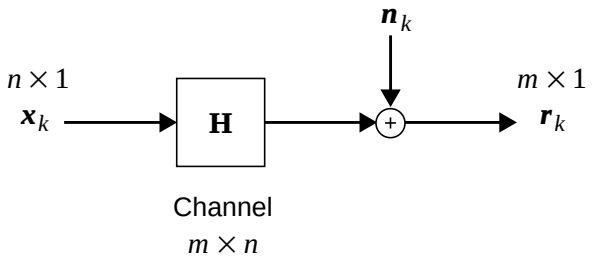


Fig. 4-1. A block diagram of an $m \times n$ noisy memoryless channel model.

noise components are independent and have zero mean. Finally, we assume that components of the channel input vector $\boldsymbol{x}_k$ are stationary, independent, and uniformly selected from a complex discrete-input alphabet. Define $x_k^{(i)}$ to be i-th component of the vector $\boldsymbol{x}_k$. The second and fourth moments of the i-th input alphabet can be defined as follows:

$$m_2^{(i)} = E[\,|\,x_k^{(i)}\,|^2], \tag{4-2}$$

$$m_4^{(i)} = E[\,|\,x_k^{(i)}\,|^4]. \tag{4-3}$$

An intrinsic property of the i-th input alphabet is the kurtosis, which is defined as:

$$\kappa_i = \frac{m_4^{(i)}}{(m_2^{(i)})^2}. \tag{4-4}$$

A lower bound for the kurtosis is given by the following theorem:

Theorem 4-1. For any given input alphabet, $\kappa \geq 1$.

Proof: This result is a direct consequence of Jensen's inequality [107], which states that $m_4 \geq (m_2)^2$.

Property 4-1. The kurtosis, as defined in (4-4), is invariant to an arbitrary scaling of the input alphabet, *i.e.*, multiplying each point in the input alphabet by a constant does not change the value of the kurtosis.

We have calculated the kurtosis for a few of the most common discrete-input alphabets.

Example 4-1. The kurtosis for M phase-shift keying (M-PSK) constellations is one, *i.e.*, $\kappa = 1$. This result is due to the fact that all of the symbols in an M-PSK constellation lie on a single circle of radius m_2.

Example 4-2. The kurtosis for the M pulse-amplitude modulation (M-PAM) alphabet $\{\pm 1, \pm 3, \ldots, \pm(M-1)\}$ is given by:

$$\kappa = \frac{9}{5} - \frac{12}{5}(M^2 - 1)^{-1}. \tag{4-5}$$

Using (4-5), we see that $\kappa = 1$ for a 2-PAM constellation, $\kappa = 1.64$ for a 4-PAM constellation, and $\kappa = 1.7314$ for a 6-PAM constellation. In the limit, as $M \to \infty$, $\kappa \to 1.8$.

Example 4-3. For the M^2 quadrature-amplitude modulation (M^2-QAM) alphabet $\{\pm 1, \pm 3, \ldots, \pm(M-1)\} \times \{\pm j, \pm 3j, \ldots, \pm(M-1)j\}$, where '$\times$' denotes the two-dimensional Cartesian product, the kurtosis is given by:

$$\kappa = \frac{7}{5} - \frac{6}{5}(M^2 - 1)^{-1}. \tag{4-6}$$

Using (4-6), we see that $\kappa = 1$ for a 4-QAM constellation, $\kappa = 1.32$ for a 16-QAM constellation, and $\kappa = 1.381$ for a 64-QAM constellation. In the limit, as $M \to \infty$, $\kappa \to 1.4$.

From these examples, we observe that the kurtosis can be used to subdivide the set of all discrete-input alphabets into two distinct and disjoint subsets.

Definition 4-1. An input alphabet is said to be *constant modulus* (CM) if $\kappa = 1$; in other words, all of the symbols in the discrete-input alphabet lie on a single circle of radius m_2.

Definition 4-2. An input alphabet is said to be *non-constant modulus* (non-CM) if $\kappa > 1$.

The importance of these two definitions will become evident later in this chapter.

4.2 VECTOR CMA COST FUNCTION

Since we have assumed that the channel has full-column rank, the transmitted vector $\boldsymbol{x}_k$ is linearly detectable and it can therefore be recovered by passing the $m \times 1$ received vector $\boldsymbol{r}_k$ through an $n \times m$ adaptive memoryless linear detector $\mathbf{C}$ as illustrated in Fig. 4-2. It can be seen from this figure that the number of outputs for the linear detector is less than the number of inputs. This detector, often referred to as a short linear detector, is the "smallest" possible detector that can be used to recover the transmitted data. The detector output is expressed as:

$$\boldsymbol{y}_k = \mathbf{F}\boldsymbol{x}_k + \mathbf{C}\boldsymbol{n}_k, \tag{4-7}$$

where $\mathbf{F} = \mathbf{CH}$ is the $n \times n$ overall transfer function matrix. It should be emphasized that the dimension of the detector output is the same as the dimension of the channel input.

We choose to adapt the memoryless linear detector $\mathbf{C}$ using a multi-dimensional generalization of the constant modulus-algorithm. The reason for choosing CMA is that it is adaptive and has low complexity. Another important advantage of CMA is that it can mitigate intersymbol interference on both minimum and non-minimum phase channels by implicitly using higher-order statistics. Further, we can build on the great deal of knowledge about the cost function in the literature.

We now propose the *vector constant-modulus algorithm* (vector CMA) cost function [20]:

$$J_v = E\left[\left(\left\|\boldsymbol{y}_k\right\|^2 - M_v\right)^2\right], \tag{4-8}$$

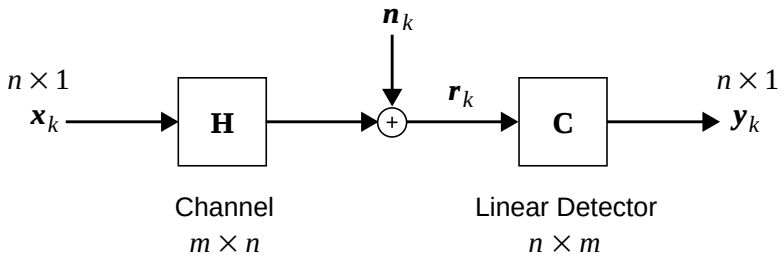


Fig. 4-2. A block diagram of an $m \times n$ memoryless channel followed by an $n \times m$ memoryless linear detector.

where $\boldsymbol{y}_k = \mathbf{C}\boldsymbol{r}_k$, and where M_v is some a constant to be specified later in this section. The cost function given by (4-8) is a special case of the combination CMA cost function with $A = 0$ and $B = 1$. Also, (4-8) reduces to the conventional single-user CMA cost function when $n = 1$.

The vector CMA cost function is a unique extension of the single-user CMA cost function to vector-valued signals. In fact, when compared to the pointwise CMA cost function, the vector CMA cost function is a natural generalization because it inherits the most important property, the invariance to an arbitrary (unitary) rotation, from the scalar CMA cost function. This is a direct consequence of the fact that the vector CMA cost function attempts to restore the modulus of the *entire* vector-valued detector output; in other words, the l_2-norm in (4-8) is invariant to an arbitrary unitary transformation. On the other hand, the pointwise CMA cost function attempts to restore the modulus of each individual component and is therefore only invariant to an arbitrary *diagonal* (unitary) rotation (see Section 2.3).

We select M_v so that in the absence of noise, the gradient is equal to the zero matrix when $\boldsymbol{y}_k = \boldsymbol{x}_k$, or equivalently, when at perfect equalization. By selecting this particular M_v, we ensure that $\mathbf{F} = \mathbf{I}$ is a local minimum of the cost function and that the detector stops updating (on average) when the transmitted data have been recovered. Expanding (4-8), the noiseless vector CMA cost function can be written in terms of the received vector $\boldsymbol{r}_k$ and the linear detector $\mathbf{C}$ as follows:

$$J_v = E\left[tr[(\mathbf{C}\boldsymbol{r}\boldsymbol{r}^*\mathbf{C}^*)^2] - 2M_v tr(\mathbf{C}\boldsymbol{r}\boldsymbol{r}^*\mathbf{C}^*) + M_v{}^2 \right]. \tag{4-9}$$

The dependence on time has been suppressed to simplify the notation. The complex gradient of (4-9) with respect to **C** is

$$\nabla_{\mathbf{C}} J_v = 4E(\|\boldsymbol{y}\|^2 \boldsymbol{y}\boldsymbol{x}^*)\mathbf{H}^* - 4M_v\mathbf{C}\mathbf{H}E(\boldsymbol{y}\boldsymbol{x}^*)\mathbf{H}^*. \tag{4-10}$$

Substituting $\boldsymbol{y} = \boldsymbol{x}$ into (4-10) and setting the gradient equal to the zero matrix, we find that M_v must satisfy:

$$\left[E(\|\boldsymbol{x}\|^2 \boldsymbol{x}\boldsymbol{x}^*) - M_v E(\boldsymbol{x}\boldsymbol{x}^*)\right]\mathbf{H}^* = \mathbf{0}. \tag{4-11}$$

Since the channel matrix **H** has full-column rank, (4-11) reduces to:

$$E(\|\boldsymbol{x}\|^2 \boldsymbol{x}\boldsymbol{x}^*) = M_v E(\boldsymbol{x}\boldsymbol{x}^*). \tag{4-12}$$

For an M_v to exist, it must simultaneously satisfy all of the equations described in (4-12). The following theorem described the conditions for which an M_v can exist.

Theorem 4-2. There exists an M_v which satisfies (4-12) if and only if

$$m_2^{(i)}(\kappa_i - 1) = K, \quad \forall\ i \tag{4-13}$$

where K is some positive constant.

Proof: Assuming that all of the users are independent, it is easy to show that the left- and right-hand sides of (4-12) are both diagonal. Hence, we can write the *i*-th equation for (4-12) as:

$$M_v \; m_2^{(i)} = m_4^{(i)} + m_2^{(i)} \sum_{j \neq i} m_2^{(j)}. \tag{4-14}$$

Dividing (4-14) by $m_2^{(i)}$ and rearranging some of the terms, we find that

$$M_v = m_2^{(i)} (\kappa_i - 1) + \sum_{j} m_2^{(j)}. \tag{4-15}$$

The last term in (4-15) is the same for all n users. Therefore, (4-15) holds if and only if $m_2^{(i)} (\kappa_i - 1)$ is a constant for all n users. ❑

Corollary 4-1. If (4-13) holds, then M_v is given by

$$M_v = \frac{E[\|\boldsymbol{x}\|^4]}{E[\|\boldsymbol{x}\|^2]}. \tag{4-16}$$

Proof: Since the condition stated in (4-13) holds, we observe that (4-14) represents a consistent set of equations. Taking the trace of both sides of (4-12), we arrive at

$$M_v E[\|\boldsymbol{x}\|^2] = E[\|\boldsymbol{x}\|^4]. \tag{4-17}$$

Thus, the optimal choice for M_v is $M_v = \frac{E[\|\boldsymbol{x}\|^4]}{E[\|\boldsymbol{x}\|^2]}$. ❑

It can be seen from (4-13) that if the input alphabet is CM for any one user ($\kappa_i = 1$ for any i), then $K = 0$ for all n users. This result implies that all n users must draw symbols from either a CM input alphabet or a non-CM input alphabet, but not both; in other words, there is no way to mix CM input alphabets with non-CM input alphabets. Hence, $K = m_2^{(i)}(\kappa_i - 1)$ for all n users only if n users draw symbols from an arbitrary set of CM input alphabets ($\kappa_i = 1, \forall\ i$), or if all n users draw symbols from an arbitrary set of non-CM input alphabets that have a kurtosis $\kappa_i = K / m_2^{(i)} + 1, \forall\ i$, or if all n users are independent and identically distributed. We should also emphasize that if (4-13) does not hold, then there does not exist an M_v that satisfies (4-11). To see this result, we look at the following example:

Example 4-4. Consider a two-user system where the first user selects data from a normalized 16-QAM constellation ($m_2^{(1)} = 1$ and $\kappa_1 = 1.32$) and the second user selects data from a normalized 4-QAM constellation ($m_2^{(2)} = 1$, $\kappa_2 = 1$). In this case, (4-12) simplifies to the following:

$$\begin{bmatrix} M_v & 0 \\ 0 & M_v \end{bmatrix} = \begin{bmatrix} 2.32 & 0 \\ 0 & 2 \end{bmatrix}. \tag{4-18}$$

Clearly, there does not exist an M_v that satisfies this equation.

For discussion throughout the rest of this chapter, we will assume that all users are independent and identically distributed, *i.e.*, $m_2 = m_2^{(i)}$, $m_4 = m_4^{(i)}$, and $\kappa = \kappa_i \; \forall \; i$. Note that M_v is related to the kurtosis κ by:

$$M_v = m_2 \, (n + \kappa - 1). \tag{4-19}$$

This assumption will simplify the analysis of the cost function later in this chapter.

4.3 Local Minima in the Absence of Noise

The usefulness of the vector CMA cost function is directly related to the answers of the following two questions:

- What are the local minima of the vector CMA cost function?
- How many local minima are there?

Initially, we will assume that the noise term in (4-1) is zero. This assumption is made to simplify the analysis of the cost function. Later in this chapter, we will analyze the cost function when the noise term is nonzero. In the previous section, we designed the vector CMA cost function such that $\mathbf{F} = \mathbf{I}$ is a local minimum. This local minimum is desirable because it implies that $\boldsymbol{y}_k = \boldsymbol{x}_k$ and that the transmitted vector can be completely recovered. Since the vector CMA cost function is invariant to an arbitrary unitary transformation, $\mathbf{F} = \mathbf{U}$, where $\mathbf{U}$ is any unitary matrix. Therefore, $\mathbf{U}$ is also a local minimum of the cost function. Thus, the vector CMA cost function may be minimized by *any* unitary matrix, and so there exists an infinite number of local minima! We should emphasize that a unitary matrix is actually a desirable minimum because we have already designed an algo-

rithm that can estimate and resolve the unitary ambiguity; this algorithm is the MPLL, which is described in detail in Chapter 3.

Definition 4-3. A unitary matrix is a *desirable minimum* of the vector CMA cost function.

One questions remains: are there any other local minima of the vector CMA cost function? The answer to this question depends upon the subset in which the input alphabet lies. The following two theorems summarize the local minima of the vector CMA cost function; the first theorem applies only to non-CM input alphabets and the second applies only to CM input alphabets.

Theorem 4-3. If the input alphabet is non-CM ($\kappa > 1$), then J_v is minimized if and only if $\mathbf{F}$ is unitary.

Proof: See Appendix 4.1.

If the input alphabet is non-CM and the linear detector is a local minima of vector CMA, then the detector output vector is related to the channel input vector by an unknown unitary matrix (an unknown arbitrary rotation), *i.e.*, $\boldsymbol{y}_k = \mathbf{U}\boldsymbol{x}_k$, for an arbitrary unitary matrix $\mathbf{U}$.

Property 4-2. For a non-CM input alphabet, vector CMA can resolve a channel up to a unitary ambiguity.

This particular detector is referred to as a *whitener*, because the covariance of the detector output vector $\boldsymbol{y}_k$ is the identity matrix, $\mathrm{E}[\boldsymbol{y}_k \boldsymbol{y}_k^*] = \mathbf{I}$.

> **Definition 4-4.** An $n \times m$ linear detector $\mathbf{C}$ is said to be a whitener if the auto-correlation of $\boldsymbol{y}_k = \mathbf{C}\boldsymbol{r}_k$ is the identity matrix, *i.e.*, $\mathbf{C}E[\boldsymbol{r}_k \boldsymbol{r}_k^*]\mathbf{C}^* = \mathbf{I}$.

In order to recover the transmitted signals, the detector must be followed by a blind unitary estimator, which identifies and eliminates this unitary ambiguity. Possible solutions to this problem include the MPLL (see Chapter 3), JADE, and EASI algorithms.

It is important to emphasize that Theorem 4-3 holds for *all* non-CM input alphabets, including a complex Gaussian input alphabet. This result seems to contradict the widely-held intuition that CMA does not work for a Gaussian input. To better understand this conundrum, we begin by considering the single-user system depicted in Fig. 4-3, where x_k is a normalized complex Gaussian input signal with $m_2 = 1$ and $f(z)$ is the overall transfer function. The noise term is assumed to be zero. The single-user CMA algorithm attempts to restore the modulus of the equalizer output y_k; in other words, the algorithm tries to match the modulus of the equalizer output with the modulus of the channel input. In this

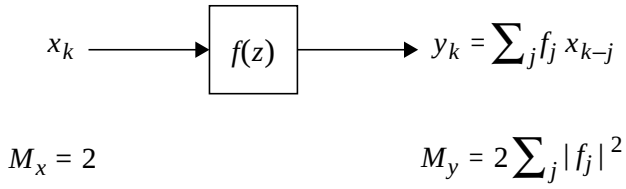


Fig. 4-3. A noiseless single-user system with a complex Gaussian input signal.

example, the modulus of the input is $M_x = 2$, while the modulus of the equalizer output is $M_y = 2\sum_j |f_j|^2$. The moduli of the two signals will match if and only if the power of the overall transfer function is unity: $\sum_j |f_j|^2 = 1$. For a Gaussian input alphabet, the single-user CMA ensures that the channel does not amplify the power of the input signal. Unfortunately, the CMA cannot equalize a channel with memory. However, if we restrict our attention to a memoryless channel, we see that the CMA can resolve it up to an arbitrary rotation, *i.e.*,

$$|f_0|^2 = |c_0 h_0|^2 = 1 \Rightarrow c_0 = e^{j\theta} h_0, \tag{4-20}$$

for some $\theta \in [0, 2\pi)$. Thus, a single-user CMA can be used to equalize a memoryless channel when the input alphabet is Gaussian. An analogous interpretation exists for the vector CMA utilizing a Gaussian input alphabet. The important lesson derived from this argument is that the vector CMA is compatible with capacity achieving systems that employ highly-shaped or near-Gaussian inputs.

Property 4-3. The vector CMA works for highly-shaped or near Gaussian input alphabets.

Theorem 4-4. If the input alphabet is CM ($\kappa = 1$), then J_v is minimized if and only if $\mathbf{F}$ has the following form:

$$\mathbf{F} = \mathbf{U}\mathbf{D}^{1/2}, \tag{4-21}$$

where **U** is a unitary matrix and **D** is a non-negative real diagonal matrix satisfying $tr(\mathbf{D}) = n$.

Proof: See Appendix 4.4.

If the input alphabet is CM, the detector output vector $\boldsymbol{y}_k$ is related to the channel input vector $\boldsymbol{x}_k$ by the following relationship: $\boldsymbol{y}_k = \mathbf{U}\mathbf{D}^{1/2}\boldsymbol{x}_k$, where **U** is an arbitrary unitary matrix and **D** is a non-negative real diagonal matrix satisfying $tr(\mathbf{D}) = n$. We see that the overall transfer function matrix given by (4-21) is *only* unitary for the special case when **D** = **I**. For all other **D**, the overall transfer function matrix is non-unitary. A non-unitary overall transfer function matrix is undesirable because a blind unitary estimator, such as the MPLL, cannot resolve the ambiguity present in the detector output vector and therefore more processing would be needed in order to recover the transmitted data.

Definition 4-5. A non-unitary matrix is an *undesirable minimum* of the vector CMA cost function.

The implications of Theorem 4-4 can be best understood by looking at the following example.

Example 4-5. A non-unitary matrix that satisfies (4-21) is given by

$$\mathbf{F} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}, \tag{4-22}$$

where $\mathbf{U} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$ and $\mathbf{D} = \begin{bmatrix} \sqrt{2} & 0 \\ 0 & 0 \end{bmatrix}$.

The overall transfer function given by (4-22) is an undesirable minimum because it implies that both detector outputs lock on to the first user and therefore the information from the second user is completely lost. In this case, because it is impossible to recover information from the second user, appending a blind unitary estimator serves no purpose. In fact, no amount of processing would be able to recover the lost information from the second user.

Other examples of undesirable minima satisfying (4-21) include:

$$\mathbf{F} = \begin{bmatrix} \sqrt{3/2} & 0 \\ 0 & \sqrt{1/2} \end{bmatrix}, \mathbf{F} = \begin{bmatrix} \sqrt{6/5} & 0 & 0 \\ 0 & \sqrt{2/5} & 0 \\ 0 & 0 & \sqrt{7/5} \end{bmatrix}. \tag{4-23}$$

Even though these minima recover information for all of the users, they are undesirable because of the incorrect gain for each user, which may lead to incorrect decisions at the decision device.

Property 4-4. For a CM input alphabet, the vector CMA cost function is minimized by both unitary and non-unitary matrices.

We should emphasize that the vector CMA cost function relies on *both* second-order and fourth-order statistics in order to invert the channel. Unfortunately, a CM input alphabet is completely described by its second-order statistics, and therefore, the cost function does not have enough information to correctly invert the channel. Thus, it is possible that the cost function can be minimized by non-unitary matrices, such as (4-22) and (4-23) given in Example 4-5.

4.4 Local Minima in the Presence of Noise

In the previous section, we determined the local minima of the vector CMA cost function assuming no noise. While this analysis is valid and informative, it is only of academic interest since in any real-world application, noise is always present. Therefore, in this section, we derive the local minima of the cost function in the presence of noise and compare the performance against the well-known minimum mean-squared error (MMSE) detector.

Recently, many authors [108-111] have demonstrated that in the presence of noise, the single-user CMA exhibits near MMSE-like performance; in other words, the performance of the CMA is similar to that of the MMSE equalizer. Unfortunately, in most cases, the analysis is based on high SNR approximations, and as a result, this analysis is only valid at high SNR. The exact behavior of the cost function at low to medium SNR is also very important, because, we can determine from it some of the intrinsic properties of the cost function, such as the valid SNR operating range. The reason that many authors use high SNR approximations is that it simplifies the analysis, which otherwise becomes intractable when there is memory in either the channel or the equalizer. However, if the attention is restricted to memoryless channels, then the analysis becomes straightforward. In fact, it is then possible to derive a closed-form expression for the behavior of the cost function at all SNR. For a single-user system, this result is uninteresting, but at higher dimensions, it becomes much more important.

Consider the noisy memoryless channel and detector model illustrated in Fig. 4-2:

$$\boldsymbol{y}_k = \mathbf{F}\boldsymbol{x}_k + \mathbf{C}\boldsymbol{n}_k, \tag{4-24}$$

where $\mathbf{F} = \mathbf{CH}$ is an $n \times n$ overall transfer function matrix, $\mathbf{C}$ is an $n \times m$ linear detector, and $\boldsymbol{x}_k$ is the $n \times 1$ channel input vector. We assume that $\boldsymbol{n}_k$ is an $m \times 1$ zero-mean white Gaussian noise vector with power spectral density $E[\boldsymbol{n}_k \boldsymbol{n}_k^*] = \sigma^2\mathbf{I}$. Substituting (4-24) into (4-8), we find that the cost function can be written as follows:

$$J_v = E\left[(\boldsymbol{x}^*\mathbf{F}^*\mathbf{F}\boldsymbol{x})^2 + (\boldsymbol{n}^*\mathbf{C}^*\mathbf{C}\boldsymbol{n})^2 + 2(\boldsymbol{x}^*\mathbf{F}^*\mathbf{F}\boldsymbol{x})(\boldsymbol{n}^*\mathbf{C}^*\mathbf{C}\boldsymbol{n}) + 2(\boldsymbol{x}^*\mathbf{F}^*\mathbf{C}\boldsymbol{n}\boldsymbol{n}^*\mathbf{C}^*\mathbf{F}\boldsymbol{x}) \right]$$

$$- 2M_v E\left[\boldsymbol{x}^*\mathbf{F}^*\mathbf{F}\boldsymbol{x} + \boldsymbol{n}^*\mathbf{C}^*\mathbf{C}\boldsymbol{n} \right] + M_v^2. \tag{4-25}$$

We refer to (4-25) as the "noisy" vector CMA cost function.

The goal of this section is to determine the overall transfer function matrix $\mathbf{F}$ and the corresponding linear detector $\mathbf{C}$ that minimizes the noisy vector CMA cost function. We restrict our attention to non-CM input alphabets because convergence to a desirable minima is guaranteed in the absence of noise. Before we state the results, we give the following definition which will be used to simplify the notation.

Definition 4-6. Let $\mathcal{U}$: $\mathbb{C}^{n\times n} \rightarrow \mathbb{C}^{n^2}$ be a function that unwraps an $n \times n$ matrix onto an $n^2 \times 1$ column vector. The unwrapping of the matrix amounts to stacking of the columns; in other words,

$$\mathcal{U}(\mathbf{G}) = [\boldsymbol{g}_1^T \, \boldsymbol{g}_2^T \, \ldots \, \boldsymbol{g}_n^T]^T, \tag{4-26}$$

where $\mathbf{G} = [\boldsymbol{g}_1 \, \boldsymbol{g}_2 \, \ldots \, \boldsymbol{g}_n]$. We see that the inverse of this function $\mathcal{U}^{-1}$: $\mathbb{C}^{n^2} \rightarrow \mathbb{C}^{n\times n}$ also exists.

The local minima for the noisy vector CMA cost function for a non-CM input alphabet are summarized by the following theorem:

Theorem 4-5. If the channel matrix $\mathbf{H}$ has the following truncated singular-value decomposition: $\mathbf{H}_{m\times n} = \mathbf{U}_{m\times n}\Sigma_{m\times n}[\mathbf{Q}_{m\times n}]^*$, and if the input alphabet is non-CM, then the noisy vector CMA cost function is minimized if and only if

$$\mathbf{F} = \mathbf{U}\mathbf{D}^{1/2}\mathbf{V}^*, \tag{4-27}$$

where $\mathbf{U}$ is an arbitrary $n \times n$ unitary matrix. The $n \times n$ unitary matrix $\mathbf{V}$ and the $n \times n$ diagonal matrix $\mathbf{D}$ are found by taking the eigendecomposition of $\mathbf{Q}\Sigma U^{-1}(\boldsymbol{g})\Sigma\mathbf{Q}^*$, *i.e.*,

$$\mathbf{V}\mathbf{D}\mathbf{V}^* = \mathbf{Q}\Sigma U^{-1}(\boldsymbol{g})\Sigma\mathbf{Q}^*. \tag{4-28}$$

The $n^2 \times 1$ vector $\boldsymbol{g}$ is given by:

$$\boldsymbol{g} = M_v\Big[(m_2)^2 U(\Sigma^2)U(\Sigma^2)^T + (m_2)^2\Sigma_\sigma^2 + (m_2)^2(\kappa-2)\mathbf{W}^*\mathbf{W} + \sigma^4 U(\mathbf{I}_n)U(\mathbf{I}_n)^T$$

$$+ \sigma^4\mathbf{I}_{n^2} + m_2\sigma^2 U(\Sigma^2)U(\mathbf{I}_n)^T + m_2\sigma^2 U(\mathbf{I}_n)U(\Sigma^2)^T + 2m_2\sigma^2\Sigma_B^2\Big]^{-1}$$

$$\Big[m_2 U(\Sigma^2) + \sigma^2 U(\mathbf{I}_n)\Big], \tag{4-29}$$

where,

$$\Sigma_{\sigma}^{2} = \begin{bmatrix} \sigma_1^2\Sigma^2 & & & 0 \\ & \sigma_2^2\Sigma^2 & & \\ & & \ddots & \\ 0 & & & \sigma_n^2\Sigma^2 \end{bmatrix} \text{ and } \Sigma_{B}^{2} = \begin{bmatrix} \Sigma^2 & & & 0 \\ & \Sigma^2 & & \\ & & \ddots & \\ 0 & & & \Sigma^2 \end{bmatrix}. \tag{4-30}$$

The $n \times n^2$ matrix $\mathbf{W}$ is defined as follows[1]:

$$\mathbf{W} = [\sigma_1{}^2 \boldsymbol{q}_1{}^* \otimes \boldsymbol{q}_1, \ldots, \sigma_1\sigma_n \boldsymbol{q}_1{}^* \otimes \boldsymbol{q}_n, \ldots, \sigma_1\sigma_n \boldsymbol{q}_n{}^* \otimes \boldsymbol{q}_1, \ldots, \sigma_n{}^2 \boldsymbol{q}_n{}^* \otimes \boldsymbol{q}_n], \tag{4-31}$$

where σ_i is the i-th diagonal element of Σ, $\boldsymbol{q}_i$ is the i-th column of $\mathbf{Q}$, and '$\otimes$' represents the Hadamard product (component-by-component product) [106]. The corresponding linear detector that minimizes the noisy vector CMA cost function is given by $\mathbf{C} = \mathbf{F}\mathbf{H}^{\dagger}$, where $\mathbf{H}^{\dagger} = (\mathbf{H}^*\mathbf{H})^{-1}\mathbf{H}^*$.

Proof: See Appendix 4.6.

Unfortunately, the results of this theorem are not intuitively obvious, but we can gain some insight as to how the linear detector combats the noise:

$$\mathbf{F} = \mathbf{U}\mathbf{D}^{1/2}\mathbf{V}^* = \mathbf{U}\mathbf{V}^* + \mathbf{U}(\mathbf{D}^{1/2} - \mathbf{I})\mathbf{V}^*. \tag{4-32}$$

By resolving the unitary ambiguity in the overall transfer function matrix, (4-32) reduces to:

1. For this particular equation, '*' represents a complex conjugate.

$$\tilde{\mathbf{F}} = \mathbf{V}\mathbf{U}^{*}\mathbf{F} = \mathbf{I} + \mathbf{V}(\mathbf{D}^{1/2} - \mathbf{I})\mathbf{V}^{*}. \tag{4-33}$$

We see from (4-33) that the local minimum of the noisy vector CMA cost function are related to the local minimum of the noiseless vector CMA cost function by a perturbation matrix. This perturbation matrix is *non-diagonal*, indicating that a certain amount of multiuser interference is required to combat the effects of noise.

By using (4-27) through (4-31), we can compare the performance of the vector CMA to that of the MMSE detector. We choose the MMSE detector because it exhibits a desirable balance between the suppression of multiuser interference and the enhancement of noise. Let $\mathrm{MSE}_i = \mathrm{E}[|y_k^{(i)} - x_k^{(i)}|^2]$ denote the mean-squared error (MSE) for the i-th user. The MMSE detector that minimizes the MSE for each user, can be expressed in two equivalent ways:

$$\mathbf{C} = \mathbf{H}^{*}(\mathbf{H}\mathbf{H}^{*} + \frac{\sigma^2}{m_2}\mathbf{I})^{-1}, \tag{4-34}$$

$$= (\mathbf{H}^{*}\mathbf{H} + \frac{\sigma^2}{m_2}\mathbf{I})^{-1}\mathbf{H}^{*}. \tag{4-35}$$

We should emphasize that both (4-34) and (4-35) produce the same result, except when the noise is zero ($\sigma^2 = 0$) and the channel is tall ($m > n$), in which case (4-34) is not valid.

The MSE for the i-th user of the vector CMA detector can be expressed as:

$$\mathrm{MSE}_i = m_2\|\boldsymbol{J}_i(\mathbf{F} - \mathbf{I})\|^2 + \sigma^2\|\boldsymbol{J}_i\mathbf{C}\|^2, \tag{4-36}$$

where $\boldsymbol{J}_i$ is a $1 \times n$ row vector with a one in the i-th position and zeros elsewhere. Assuming that an ideal rotator resolves the remaining unitary ambiguity, the $n \times n$ overall transfer function and the $n \times n$ linear detector that minimize the noisy vector CMA cost function are $\mathbf{F} = \mathbf{V}\mathbf{D}^{1/2}\mathbf{V}^*$, where $\mathbf{V}$ and $\mathbf{D}$ are specified by (4-28), and $\mathbf{C} = \mathbf{F}(\mathbf{H}^*\mathbf{H})^{-1}\mathbf{H}^*$, respectively. The MSE for the i-th user of the MMSE detector, expressed in terms of a singular-value decomposition of the channel matrix: $\mathbf{H} = \mathbf{U}\mathbf{S}\mathbf{V}^*$, where $\mathbf{U}$ is an $m \times m$ unitary matrix, $\mathbf{V}$ is an $n \times n$ unitary matrix, and Σ is an $n \times n$ non-negative real diagonal matrix, is

$$\mathrm{MSE}_i = \sigma^2 \boldsymbol{v}_i^* (\mathbf{S}^*\mathbf{S} + \frac{\sigma^2}{m_2}\mathbf{I})^{-1}\boldsymbol{v}_i, \tag{4-37}$$

where $\boldsymbol{v}_i$ is the i-th column of $\mathbf{V}$.

In the following experiment, we compare the theoretical performance of the vector CMA using (4-36) to that of the MMSE detector using (4-37).

Experiment 4-1. Consider a receiver with $m = 6$ sensors. In Fig. 4-4, we plot MSE_1 versus $\mathrm{SNR}_1 = \sum_{j=1}^{m} |h_{j,1}|^2/\sigma^2$ for two different cases: $n = 2$ and $n = 6$ users. The curves are generated by averaging the mean-squared error over 500 different random $6 \times n$ channels, where the coefficients are drawn independently from a zero-mean, unit-variance complex Gaussian distribution. The columns are scaled so that all of the odd-numbered users have energy 10 dB below that of the even-numbered users. The input alphabet is assumed to be 16-QAM. For the two-user case, the curves demonstrate that both the vector CMA detector and the MMSE detector achieve similar performance for

$SNR_1 > 0$ dB. Even at a low SNR_1 of –15 dB, the vector CMA detector only suffers a modest 2 dB penalty when compared to the MMSE detector. For the six-user case, the gap in performance between the two detectors widens only slightly.

This experiment suggests that the MSE performance of the vector CMA detector is essentially similar to that of the MMSE detector.

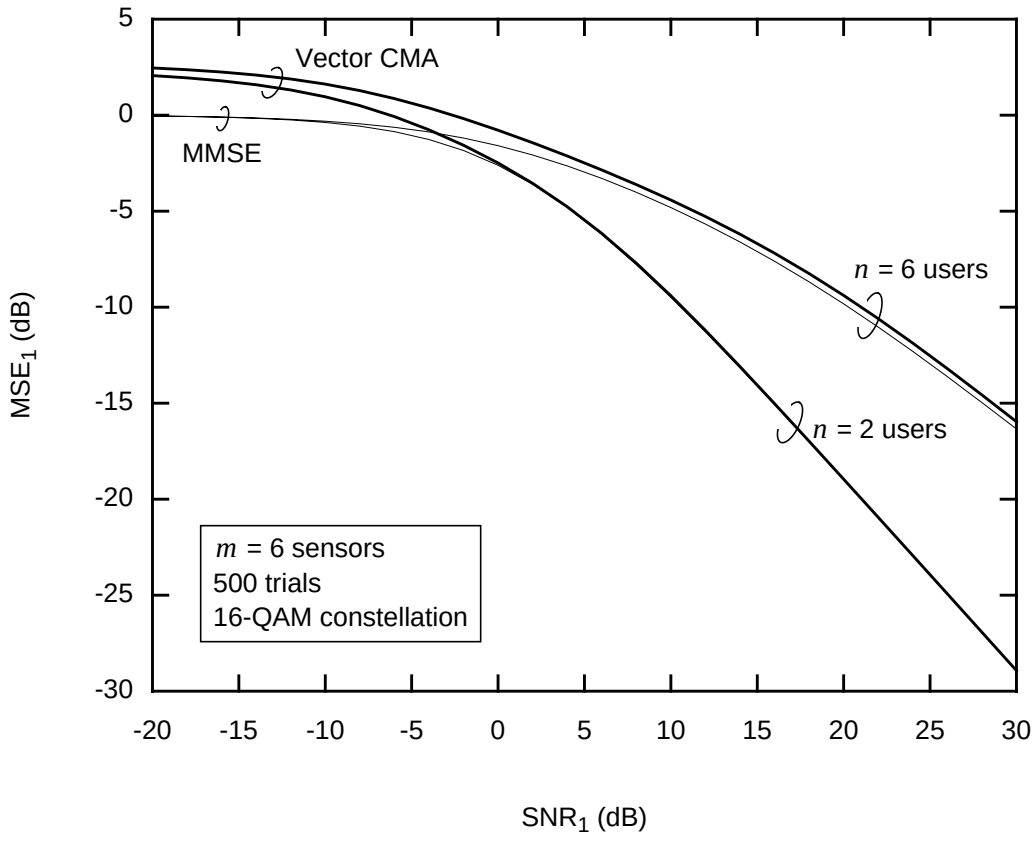


Fig. 4-4. A comparison of the mean-squared error versus SNR of the vector CMA detector and the minimum-MSE detector.

Property 4-5. Vector CMA exhibits near MMSE-like performance.

This result is reassuring because achieving performance similar to that of the MMSE detector is the goal of all blind algorithms.

4.5 VECTOR CMA WITH GRAM-SCHMIDT CONSTRAINT

In Section 4.3, we showed that the local minima for the noiseless vector CMA cost function is always desirable if the input alphabet is non-CM. If the input alphabet is CM, the cost function is minimized by both desirable and undesirable local minima. We can make the vector CMA cost function more compatible with CM input alphabets by modifying the cost function to penalize the non-unitary matrices. The non-unitary matrices described by (4-21) do not satisfy the following property: $E[\boldsymbol{y}_k \boldsymbol{y}_k^*] = m_2 \mathbf{I}$. Hence, the undesirable minima can be eliminated by modifying the vector CMA cost function to penalize those solutions for which $E[\boldsymbol{y}_k \boldsymbol{y}_k^*] \neq m_2 \mathbf{I}$:

$$J_{GS} = J_v + \left\| E[\boldsymbol{y}_k \boldsymbol{y}_k^*] - m_2 \mathbf{I} \right\|_F^2, \tag{4-38}$$

where $\| \cdot \|_F$ is the Frobenius norm for matrices and where J_v is the vector CMA cost function described by (4-8). We refer to (4-38) as the *vector CMA cost function with Gram-Schmidt constraint* (GSC), because the additional term in (4-38) is minimized when the rows of the linear detector are orthonormal with respect to the autocorrelation of the received vector, a condition that is equivalent to a Gram-Schmidt constraint on the rows of the linear detector $\mathbf{C}$.

The vector CMA cost function with GSC can be thought of as a multi-dimensional generalization of the CRIMNO (criterion with memory non-linearity) algorithm [Nikias]. We have extended their concept of decorrelation in time to decorrelation in space. We should also point out that the second term in (4-38) is closely related to the second term of the decorrelation CMA cost function (see (2-32)). The difference between these two additional terms is that while the former penalizes cross-correlations in space among the users as well as restores the modulus for each user, the latter only penalizes for cross-correlations in space among the users. Whereas both terms of (4-38) help to restore the modulus for each user, the decorrelation CMA cost function leaves the restoration of the moduli to the pointwise CMA cost function. From this argument, it is clear that the additional term in (4-38) accomplishes more, and in fact, as we will show later, it helps to speed the convergence of the algorithm.

Now consider the case when the noise term in (4-1) is zero. It is easy to show that the vector CMA cost function is zero at its local minima when the input alphabet is CM. Recall from Theorem 4-4 that the local minima for a CM input alphabet are given by $\mathbf{F} = \mathbf{U}\mathbf{D}^{1/2}$, where $\mathbf{U}$ is an arbitrary unitary matrix and $\mathbf{D}$ is a non-negative diagonal matrix satisfying $tr(\mathbf{D}) = n$. Using the relationship $\boldsymbol{y} = \mathbf{F}\boldsymbol{x}$, the cost function at the local minima simplifies to the following:

$$J_v = E\left[\left(\left\|\mathbf{D}^{1/2}\boldsymbol{x}\right\|^2 - n m_2\right)^2\right], \tag{4-39}$$

where we have used the fact that $M_v = n m_2$ for a CM input alphabet. Expanding (4-39), we obtain:

$$J_v = E\left[\left\|\mathbf{D}^{1/2}\boldsymbol{x}\right\|^4\right] - 2nm_2E\left[\left\|\mathbf{D}^{1/2}\boldsymbol{x}\right\|^2\right] + (nm_2)^2. \tag{4-40}$$

Since the input alphabet is CM and $tr(\mathbf{D}) = n$, $E\left[\left\|\mathbf{D}^{1/2}\boldsymbol{x}\right\|^4\right]$ and $E\left[\left\|\mathbf{D}^{1/2}\boldsymbol{x}\right\|^2\right]$ reduce to:

$$E\left[\left\|\mathbf{D}^{1/2}\boldsymbol{x}\right\|^4\right] = (nm_2)^2, \tag{4-41}$$

$$E\left[\left\|\mathbf{D}^{1/2}\boldsymbol{x}\right\|^2\right] = nm_2. \tag{4-42}$$

Substituting (4-41) and (4-42) into (4-40), we see that $J_v = 0$ at the local minima.

We can show that the vector CMA cost function with GSC eliminates the undesirable local minima by looking at the value of the cost function at both the desirable and undesirable local minima. For a desirable local minimum, the first term in (4-38) is zero, since it is a local minimum, and the second term in (4-38) is zero, since it satisfies the property: $E[\boldsymbol{y}_k\boldsymbol{y}_k^*] = m_2\mathbf{I}$. Thus, $J_{GS} = 0$ for all desirable minima. For an undesirable local minimum, the first term in (4-38) is zero, since it is a local minimum, and the second term in (4-38) is non-zero, because it does not satisfy the property: $E[\boldsymbol{y}_k\boldsymbol{y}_k^*] = m_2\mathbf{I}$. Thus, $J_{GS} > 0$ for all undesirable minima. Clearly, the undesirable local minima do not minimize (4-38). Therefore, the undesirable local minima cannot be local minima of the vector CMA cost function with GSC.

The question remains: does the additional term in the vector CMA cost function with GSC introduce any new local minima? Fortunately, the answer is no. The local minima for this cost function can be summarized by the following theorem:

Theorem 4-6. For all input alphabets ($\kappa \geq 1$), J_{GS} is minimized if and only if $\mathbf{F}$ is unitary.

Proof: See Appendix 4.8.

Thus, the detector output vector is related to the channel input vector by an unknown unitary matrix (an unknown arbitrary rotation), *i.e.*, $\boldsymbol{y}_k = \mathbf{U}\boldsymbol{x}_k$, for an arbitrary unitary matrix $\mathbf{U}$.

Property 4-6. Vector CMA with GSC can resolve a channel up to a unitary ambiguity for all input alphabets.

This particular detector is also referred to as a *whitener*. The transmitted vector can be recovered by using a blind unitary estimator, such as the MPLL described in Chapter 3, to identify and eliminate the remaining unitary ambiguity.

4.6 ADAPTIVE STOCHASTIC ALGORITHM

The main focus of this research is to design an adaptive linear detector. There are several methods for adaptively minimizing a cost function, including the classical steepest-descent algorithm, the Newton-Raphson algorithm, and the recursive-least squares algorithm. We intend to focus here on the classical steepest-descent algorithm, because this technique has low-complexity and it provides reasonable performance. In this algorithm, the detector tap weights are adjusted according to the following algorithm:

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \frac{\mu}{4} \nabla_{\mathbf{C}} J_v, \tag{4-43}$$

where $\nabla_{\mathbf{C}}$ is the complex gradient of the cost function with respect to the detector tap weights and where μ is the step size.

4.6.1 Vector CMA

The vector CMA cost function (see Section 4.2) is defined as follows:

$$J_v = E\left[\left(\|\boldsymbol{y}_k\|^2 - M_v\right)^2\right], \tag{4-44}$$

where $\boldsymbol{y}_k = \mathbf{C}\boldsymbol{r}_k$ and $M_v = E[\|\boldsymbol{x}_k\|^4] / E[\|\boldsymbol{x}_k\|^2]$. This cost function can be written in terms of the linear detector $\mathbf{C}$ and the detector input vector $\boldsymbol{r}_k$ as follows:

$$J_v = E\left[\ tr\left\{(\mathbf{C}\boldsymbol{r}_k\boldsymbol{r}_k{}^*\mathbf{C}^*)^2\right\} - 2M_v tr\left\{\mathbf{C}\boldsymbol{r}_k\boldsymbol{r}_k{}^*\mathbf{C}^*\right\} + M_v{}^2\right]. \tag{4-45}$$

The complex gradient of (4-45) with respect to $\mathbf{C}_k$ is given by:

$$\nabla_{\mathbf{C}} J_v = 4E[\boldsymbol{e}_k\boldsymbol{r}_k{}^*], \tag{4-46}$$

where the "error signal" $\boldsymbol{e}_k$ is defined by

$$\boldsymbol{e}_k = \boldsymbol{y}_k\left(\|\boldsymbol{y}_k\|^2 - M_v\right). \tag{4-47}$$

Substituting $4\boldsymbol{e}_k\boldsymbol{r}_k{}^*$ as a stochastic approximation of the gradient in the steepest-descent algorithm, we arrive at the following update equation for the linear detector:

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \mu \boldsymbol{e}_k \boldsymbol{r}_k^*, \tag{4-48}$$

where μ is the step size. We refer to this algorithm as the vector CMA. We see that this algorithm reduces to the familiar single-user CMA when $n = 1$. It is also important to point out that the update equation described by (4-48) is the same as the one for vector LMS (see Section 2.2), except that the error signal $\boldsymbol{e}_k$ is defined differently.

In summary, the vector constant-modulus algorithm is defined by (4-47), (4-48), and $M_v = E[\|\boldsymbol{x}_k\|^4] / \mathrm{E}[\|\boldsymbol{x}_k\|^2]$.

4.6.2 Vector CMA with Gram-Schmidt Constraint

The vector CMA cost function with Gram-Schmidt constraint (see Section 4.5) is defined as follows:

$$J_{GS} = J_v + \left\| E[\boldsymbol{y}_k \boldsymbol{y}_k^*] - m_2 \mathbf{I} \right\|_F^2, \tag{4-49}$$

where $\boldsymbol{y}_k = \mathbf{C}\boldsymbol{r}_k$. The cost function can be written in terms of the linear detector $\mathbf{C}$ and the detector input vector $\boldsymbol{r}_k$ as follows:

$$J_{mv} = J_v + tr\left[(\mathbf{C}\Phi_0\mathbf{C}^*)^2 - 2m_2(\mathbf{C}\Phi_0\mathbf{C}^*) + (m_2)^2\mathbf{I} \right], \tag{4-50}$$

where $\Phi_0 = E[\boldsymbol{r}_k \boldsymbol{r}_k^*]$. The complex gradient of (4-50) with respect to $\mathbf{C}$ is given by:

$$\nabla_{\mathbf{C}} J_{GS} = 4E[\boldsymbol{e}_k \boldsymbol{r}_k^*] + 4(E[\boldsymbol{y}_k \boldsymbol{y}_k^*] - m_2\mathbf{I})E[\boldsymbol{y}_k \boldsymbol{r}_k^*]. \tag{4-51}$$

Substituting a stochastic approximation of (4-51) in the classical steepest-descent algorithm, we arrive at the following update for the linear detector:

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \mu \tilde{\boldsymbol{e}}_k \boldsymbol{r}_k^*, \tag{4-52}$$

where $\tilde{\boldsymbol{e}}_k = \boldsymbol{e}_k + (E[\boldsymbol{y}_k \boldsymbol{y}_k^*] - m_2\mathbf{I})\boldsymbol{y}_k$ and μ is the step size.

The update for linear detector given by (4-52) requires an estimate of $E[\boldsymbol{y}_k \boldsymbol{y}_k^*]$. Since $\boldsymbol{y}_k$ is a non-stationary vector random process, it is difficult to obtain an accurate estimate of $E[\boldsymbol{y}_k \boldsymbol{y}_k^*]$. By substituting $E[\boldsymbol{y}_k \boldsymbol{y}_k^*] = \mathbf{C}_k \Phi_0 \mathbf{C}_k^*$ in (4-52), we arrive at an alternative update equation for the linear detector:

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \mu \bar{\boldsymbol{e}}_k \boldsymbol{r}_k^*. \tag{4-53}$$

where $\bar{\boldsymbol{e}}_k = \boldsymbol{e}_k + (\mathbf{C}_k \Phi_0 \mathbf{C}_k^* - m_2\mathbf{I})\boldsymbol{y}_k$. An estimate of $\Phi_0 = E[\boldsymbol{r}_k \boldsymbol{r}_k^*]$ is now required instead of an estimate of $E[\boldsymbol{y}_k \boldsymbol{y}_k^*]$. Since we have assumed that both the input vector $\boldsymbol{x}_k$ and the channel $\mathbf{H}$ are stationary, it is easy to obtain an accurate estimate of Φ_0. In fact, this estimate can be generated by using a simple running average:

$$\hat{\Phi}_0 = \frac{1}{k-1} \sum_{l=1}^{k} \boldsymbol{r}_l \boldsymbol{r}_l^*. \tag{4-54}$$

As k gets larger, this sum approaches Φ_0. Since we can estimate Φ_0 fairly accurately, we expect the update equation specified by (4-53) to converge more quickly. In a practical implementation $\hat{\Phi}_0$ should be substituted for Φ_0 in (4-53).

An advantage of using $\mathbf{C}_k\hat{\Phi}_0\mathbf{C}_k^*$ instead of an estimate of $E[\boldsymbol{y}_k\boldsymbol{y}_k^*]$ is that since $\mathbf{C}_k\hat{\Phi}_0\mathbf{C}_k^*$ is determined more accurately, the algorithm should converge more quickly. A disadvantage of using $\mathbf{C}_k\hat{\Phi}_0\mathbf{C}_k^*$ is that, because this term is a product of three matrices, there is an increase in computational complexity. The computational burden for this additional term can be quite large even for moderate values of n and m. However, a redeeming feature is that this additional term that is based solely on second-order statistics should converge quickly and require less symbols than the original vector CMA to invert the channel. A comparison between the computational complexity of vector CMA and vector CMA with GSC will be given in the next section.

Finally, even though the vector CMA with GSC was designed especially for CM input alphabets, it can also be used for non-CM input alphabets. If computational complexity is not an issue and the speed of convergence is important, then the vector CMA with GSC would be preferred.

In summary, the vector constant-modulus algorithm with Gram-Schmidt constraint is defined by (4-47), (4-53), (4-54), and $M_v = E[\|\boldsymbol{x}_k\|^4] / \mathrm{E}[\|\boldsymbol{x}_k\|^2]$.

4.7 Experimental Results

We conclude this chapter with several computer experiments. In the first three experiments, we consider a uniform linear array with half-wavelength spacing and a non-CM (16-QAM) input alphabet. For this channel, we compare the performance, in terms of mean-squared error, speed of convergence, and complexity, of the vector CMA detector with that of the decorrelation CMA detector [100,101], the combination CMA detector [102], and the project-whiten-rotate (PWR) detector [54]. In the next three experiments,

we consider a synchronous CDMA application with a CM input alphabet. Since the input alphabet is CM, we compare the performance of the vector CMA with GSC detector with that of the decorrelation CMA detector and the PWR detector. The seventh experiment demonstrates that the vector CMA detector is compatible with a shaped input alphabet. Finally, in the last experiment, we compare the computational complexity of all the detectors used in the previous experiments.

4.7.1 Rotational Ambiguity and Performance Measure

As mentioned previously in this chapter, the vector CMA detector and the vector CMA with GSC detector are both invariant to an arbitrary unitary ambiguity. Causey has shown in [54] that the PWR detector is also invariant to a unitary ambiguity. The remaining unitary ambiguity can be resolved by filtering the detector output through an appropriate unitary matrix **Q**, as shown in Fig. 4-5. If the overall transfer function matrix **F** = **CH** is known, then the unitary matrix **Q** that minimizes the MSE is given by:

$$\mathbf{Q} = \mathbf{U}\mathbf{V}^{*}, \tag{4-55}$$

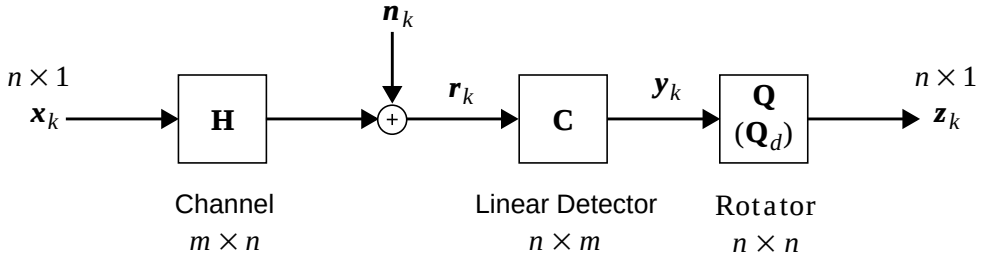


Fig. 4-5. A block diagram of a memoryless channel followed by a memoryless linear detector and a memoryless unitary rotator.

where the $n \times n$ unitary matrices $\mathbf{U}$ and $\mathbf{V}$ are specified by the singular-value decomposition of $\mathbf{F} = \mathbf{U}\mathbf{D}\mathbf{V}^*$. The derivation of (4-55) can be found in Appendix 4.9. In a practical implementation, the appropriate unitary matrix $\mathbf{Q}$ can be determined by using any one of the following blind unitary estimators: JADE [47-49], EASI [50], or the MPLL (see Chapter 3).

The pointwise CMA-based detectors, described in Chapter 2, do not realize an exact inverse of the channel, because of their invariance to an arbitrary diagonal unitary ambiguity. This type of ambiguity can be resolved by filtering the detector output through an appropriate diagonal unitary matrix $\mathbf{Q}_d$, as shown in Fig. 4-5. If the overall transfer function matrix $\mathbf{F} = \mathbf{C}\mathbf{H}$ is known, then the diagonal unitary matrix $\mathbf{Q}_d$ that minimizes the MSE is given by:

$$[\mathbf{Q}_d]_{ii} = exp(j\angle[\mathbf{F}\mathbf{P}_{\mathrm{F}}{}^*]_{ii}), \tag{4-56}$$

where $\mathbf{P}_{\mathrm{F}}$ removes both the ambiguity in assigning labels to each user and the ambiguity inherent in all QAM constellation from the overall transfer function matrix $\mathbf{F}$. The derivation of (4-56) can be found in Appendix 4.10. In a practical implementation, the proper the proper $\mathbf{Q}_d$ can be determined by using a bank of independent single-user phase-locked loops (PLLs).

Even after filtering the detector output through either $\mathbf{Q}$ or $\mathbf{Q}_d$, there still remains two sources of ambiguities: an ambiguity in assigning labels to each user and a 90° ambiguity inherent to all QAM constellations. The first ambiguity can be resolved at a higher layer, while the second ambiguity can be resolved by using differential encoding. These two

ambiguities typically lead to the relation $\mathbf{Q}^*\mathbf{F} = \mathbf{P}$ (or $\mathbf{Q}_d{}^*\mathbf{F} = \mathbf{P}$), where $\mathbf{P}$ is a complex permutation matrix, *i.e.*, a matrix consisting of only one nonzero value from the set $\{1, j, -1, -j\}$ in each row and each column. The matrix $\mathbf{P}$ is an inherent problem of blind multiuser detection and must be taken into account when calculating the MSE.

For each of the detectors in the following experiments, we use the MSE as a measure of its performance. The MSE for the i-th user is defined as follows:

$$\mathrm{MSE}_i = E[\,|\,\bar{x}_k^{(i)} - z_k^{(i)}\,|^2], \tag{4-57}$$

where $\bar{\boldsymbol{x}}_k = \mathbf{P}\boldsymbol{x}_k$ is a reordered and rotated version of the channel input vector and $\boldsymbol{z}_k = \mathbf{Q}^*\boldsymbol{y}_k$ (or $\boldsymbol{z}_k = \mathbf{Q}_d{}^*\boldsymbol{y}_k$) is the filtered detector output vector.

4.7.2 Uniform Linear Array

In the following three experiments, we consider an 16-user, 3-sensor uniform linear array with $\lambda/2$-spacing. We assume that each users draws symbols independently and uniformly from a 16-QAM input alphabet and that the angle of arrival for each user is given by: $\theta_1 = 65°$, $\theta_2 = -35°$, and $\theta_3 = 0°$, respectively. All angles were measured from broadside. Given these parameters, the channel model is given by (4-1) with

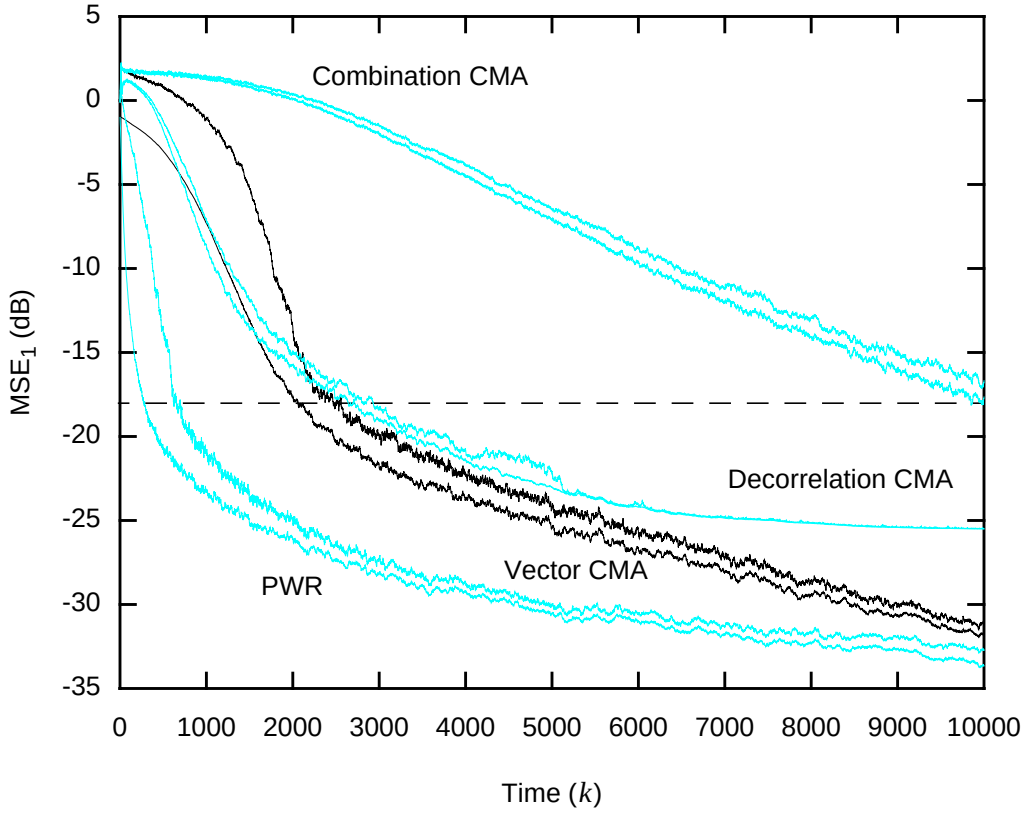
$$\mathbf{H}_{16\times 3} = \frac{1}{\sqrt{16}}\mathbf{V}\mathbf{A}, \tag{4-58}$$

where $[\mathbf{V}]_{ij} = exp[j\frac{\pi}{\lambda}(i-1)sin(\theta_j)]$, $\mathbf{A}_{3\times 3} = diag(A_1, A_2, A_3)$, and $A_i{}^2$ is the received power of the i-th user. The received powers were chosen such that the second and the third users

are 6 dB and 12 dB stronger than the first user, respectively. We also assume that the noise vector $\boldsymbol{n}_k$ in (4-1) is a complex Gaussian random variable with covariance matrix $\sigma^2\mathbf{I}$.

In the first experiment, we demonstrate the convergence of the vector CMA detector, decorrelation detector, combination CMA detector [102], and PWR detector [54] in the absence of noise.

Experiment 4-2. In Fig. 4-6, we plot the MSE_1 versus time for these four detectors, assuming no noise. Each curve is an ensemble average of 500 different random input sequences. For each detector, two curves are displayed in the figure: the lower curve corresponds to a fictitious system employing the minimum-MSE (MMSE) PLL or MPLL, while the upper curve corresponds to an actual PLL or MPLL implementation. The parameters for each detector were optimized to provide the fastest rate of convergence to achieve an open-eye diagram, or equivalently an $\text{MSE}_1 = -18$ dB. The step size for the vector CMA detector, decorrelation CMA detector ($A = 1$, $B = 1$), and combination CMA detector ($A = 1$, $B = 1.5$), were $\mu_{k,\text{vec}} = 0.075/(1 + k/2200)$, $\mu_{k,\text{dec}} = 0.15/(1 + k/1000)$, and $\mu_{k,\text{com}} = 0.045/(1 + k/6000)$, respectively. We used a 15-point causal rectangular window to estimate the cross-correlation terms in decorrelation CMA. The step size was $\mu_{k,\text{pwr}} = 0.06/(1 + k/500)$ for the PWR detector, it was $\beta_{k,\text{agc}} = 0.06/(1 + k/500)$ for the AGC, and it was $\lambda_k = 0.80/2^{(k/500)}$ for the subspace projector. The step size for the MPLL was decreased with time according to $\lambda_k = 0.6/2^{(k/2000)}$. The bank of scalar PLLs parameters, used by the two extensions of pointwise CMA, were $\alpha_1 = 0.08$ and $\alpha_2 = 10^{-5}$. From the curves, we see that the PWR detector is fastest to converge,



Vector CMA:	$\mu_k = 0.075/(1+k/2200)$
Decorrelation CMA:	$\mu_k = 0.15/(1+k/1000)$
	$A = 1,\ B = 1$
Combination CMA:	$\mu_k = 0.045/(1+k/6000)$
	$A = 1,\ B = 1.5$
PWR:	$\mu_k = 0.06/(1+k/500)$
	$\beta_k = 0.06/(1+k/500)$
	$\lambda_k = 0.80/2^{(k/500)}$
MPLL:	$\lambda_k = 0.60/2^{(k/2000)}$
PLLs:	$\alpha_1 = 0.08,\ \alpha_2 = 10^{-5}$

Fig. 4-6. Comparison of the vector CMA detector, decorrelation CMA detector, combination CMA detector, and the PWR detector, in terms of MSE_1 versus time, for a noiseless 16×3 uniform linear-array with half-wavelength spacing application with 16-QAM input alphabet, assuming both actual rotators and MMSE rotators.

followed closely by the vector CMA and the decorrelation CMA detector. The combination CMA fails to converge to an open-eye within 10,000 symbols. Even though the initial convergence of the decorrelation CMA detector is faster than that of the vector CMA detector, the vector CMA detector actually opens the eye faster than the decorrelation CMA detector. Both the vector CMA detector and the PWR detector curves show that the actual MPLL initially suffers a 4 dB penalty relative to the MMSE MPLL, which is due to the fact that the initial decisions made by the MPLL are incorrect. Once these detectors converge, the discrepancy between them disappears. A similar degradation also appears when the actual bank of PLLs is compared to the MMSE bank of PLLs, but as the detectors converge, the discrepancy in this case also disappears.

In the next experiment given below, we examine the effect of noise on the convergence time for each detector.

Experiment 4-3. We define the *convergence time* to be the average number of symbols it takes for each detector to reach an $\mathrm{MSE}_1 = -18$ dB. The convergence time is a measure of how many symbols it takes to open the eye diagram. In Fig. 4-7, the convergence time for the detectors, with the exception of the combination CMA detector, versus $\mathrm{SNR}_1 = \sum_{j=1}^{m} |h_{j,1}|^2/\sigma^2$ is plotted. The combination CMA detector does not appear on this plot, because its convergence time is greater than 10,000 symbols for all values of SNR. Each curve was generated by averaging over 500 different random input and noise

realizations at each SNR. The step size for the vector CMA detector and decorrelation CMA detector ($A = 1$, $B = 1$) were $\mu_{k,\text{vec}} = \mu_{0,\text{vec}}/(1 + k/k_\mu)$ and $\mu_{k,\text{dec}} = \mu_{0,\text{dec}}/(1 + k/k_\mu)$, respectively. We used a 15-point causal rectangular window to estimate the cross-correlation terms in decorrelation CMA. The step size was $\mu_{0,\text{pwr}}/(1 + k/k_\mu)$ for the PWR detector, it was $\beta_{k,\text{agc}} = \beta_{0,\text{agc}}/(1 + k/k_\beta)$ for the AGC, and it was given by $\lambda_k = 0.80/2^{(k/500)}$ for the subspace projector. The step size for the MPLL was decreased with time according to $\lambda_k = 0.6/2^{(k/2000)}$. The bank of scalar PLLs parameters, used by the two extensions of pointwise CMA, were $\alpha_1 = 0.08$ and $\alpha_2 = 10^{-5}$. The parameters

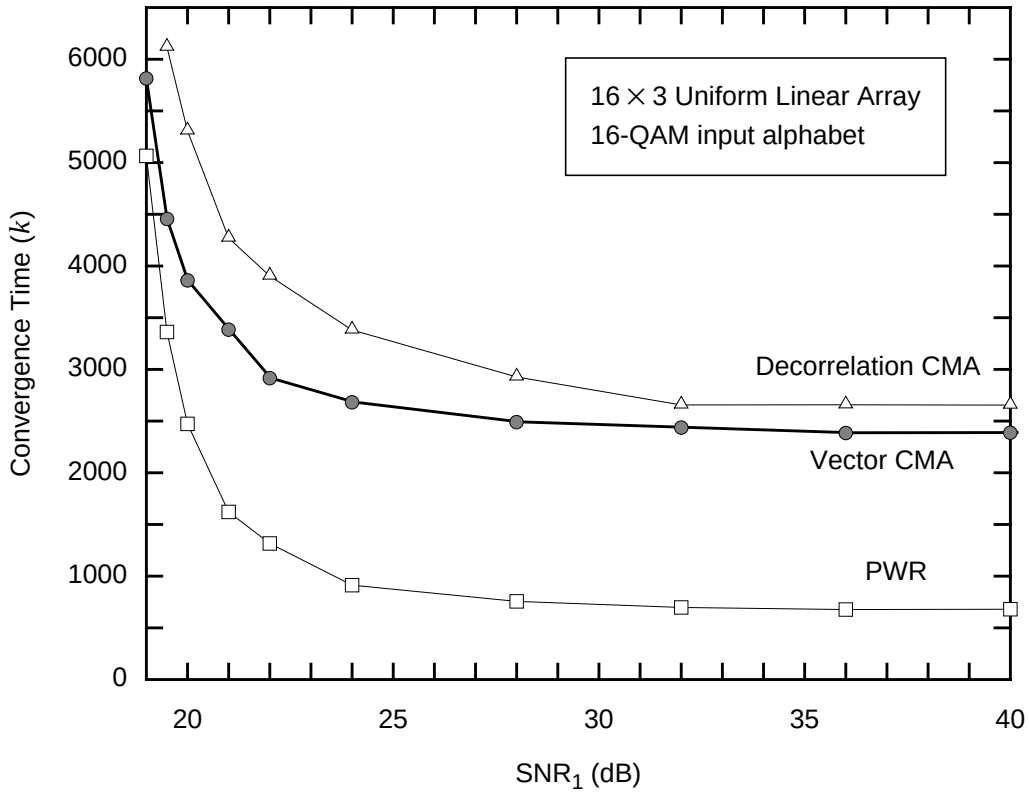


Fig. 4-7. Comparison of the convergence time of the three detector versus SNR_1 for a 16×3 uniform linear-array with half-wavelength spacing application with 16-QAM input alphabet. The optimal parameters for each detector can be found in Table 4-1.

for each detector were optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram at each SNR; the values for the optimal parameters are listed in Table 4-1. These curves show that the PWR detector provides the fastest convergence time, followed by the vector CMA detector and the decorrelation CMA detector. We also observe that as the SNR increases, the convergence time for each detector becomes a constant, suggesting that there is a fundamental limit to the speed of convergence.

In this next experiment, we examine both the convergence time and the computational complexity of the detectors.

Experiment 4-4. We define the *total complexity* to be the product of the average convergence time and the number of floating point operations (FLOPS) required per symbol. It is a measure of how many FLOPS it takes to open the eye diagram. In Fig. 4-8, we plot the total complexity for each

TABLE 4-1: Optimized Parameters for a Noisy Uniform Linear-Array Application.

SNR_1 (dB)	Vector CMA	PWR	Decorrelation CMA
19.0	$\mu_0 = 0.070$, $k_\mu = 1800$	$\mu_0 = 0.06$, $k_\mu = 300$ $\beta_0 = 0.06$, $k_\beta = 300$	—
19.5	0.070, 1800	—	$\mu_0 = 0.13$, $k_\mu = 1000$
20.0	0.072, 1800	0.06, 300; 0.06, 300	0.13, 1000
21.0 – 24.0	0.077, 1800	0.06, 400; 0.06, 400	0.13, 1000
28.0	0.077, 1800	0.06, 500; 0.06, 500	0.15, 1000
32.0 – 40.0	0.075, 2200	0.06, 500; 0.06, 500	0.15, 1000

detector versus SNR_1. We see that the total complexity of the vector CMA detector is actually the smallest, followed by the decorrelation CMA detector, and the PWR detector, which has the largest total complexity. Even though the PWR detector converges more quickly than vector CMA detector, it needs *three* times as many FLOPS to open the eye diagram.

From Experiment 4-3 and Experiment 4-4, we can infer that, for a 16×3 ULA with $\lambda/2$-spacing, the vector CMA detector provides the best trade-off between convergence time and computational complexity. The PWR detector provides the fastest convergence, but at

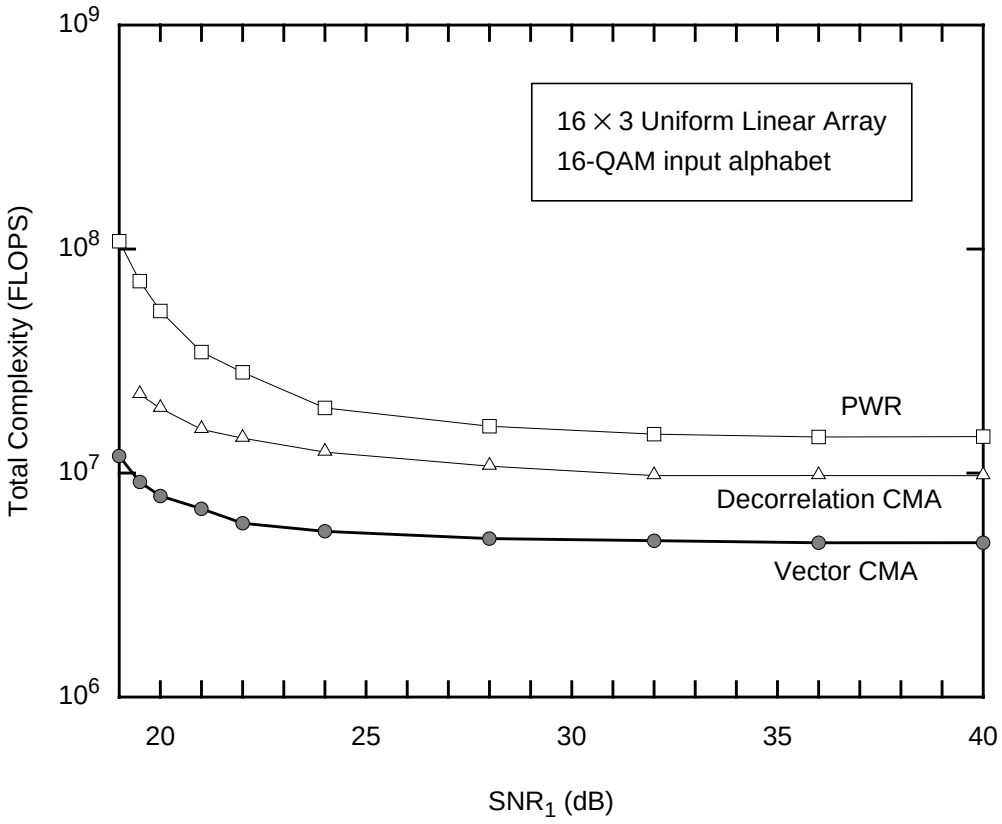


Fig. 4-8. Comparison of the total complexity of the three detector versus SNR_1 for a 16×3 uniform linear-array with half-wavelength spacing application with 16-QAM input alphabet. The optimal parameters for each detector can be found in Table 4-1.

the expense of high computational complexity. Finally, the decorrelation CMA detector has the worst performance and the highest computational complexity.

4.7.3 Synchronous CDMA

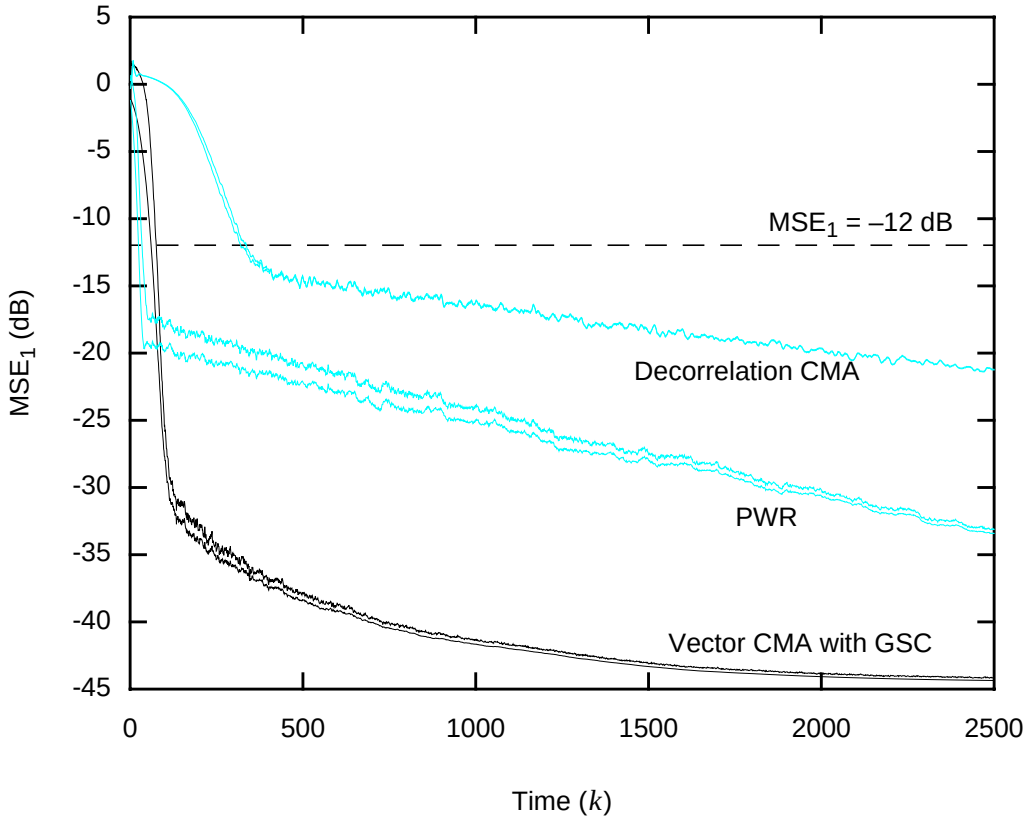
In the following three experiments, we consider a two-user synchronous direct-sequence CDMA application. We assume that each user draws symbols independently and uniformly from a 4-QAM input alphabet. We further assume that the transmit pulse-shape filters are Nyquist and that the received uses a chip-rate sampled-match filter followed by a serial-to-parallel (S/P) converter. Let $c_i \in \{\pm 1\}^{16}$ denote the 16-chip binary signature sequence for the *i*-th user. Given this definition, the channel model is given by (4-1) with

$$\mathbf{H}_{16\times 2} = \frac{1}{\sqrt{16}}[c_1\ c_2]\mathbf{A}, \tag{4-59}$$

where $\mathbf{A}_{2\times 2} = diag(A_1, A_2)$ and ${A_i}^2$ is the received power of the *i*-th user. The correlation between the two binary signature sequences is given by $\rho = \frac{1}{16}{c_1}^T c_2 = -\frac{3}{8}$. The received powers were chosen such that $SIR_1 = -10$ dB. Finally, we assume that the noise vector $\boldsymbol{n}_k$ in (4-1) is a complex Gaussian random variable with covariance matrix $\sigma^2\mathbf{I}$.

In the first experiment, we demonstrate the convergence of the vector CMA with GSC detector, decorrelation detector, and PWR detector when the noise is zero.

> **Experiment 4-5.** In Fig. 4-9, we plot the MSE_1 versus time for each detector, assuming no noise. Each curve is an ensemble average of 500 different random input sequences. There are two curves for each detector: the lower curve corresponds to a fictitious system employing the minimum-MSE



Vector CMA with GSC:	$\mu_k = 0.038/(1+k/200)$
Decorrelation CMA:	$\mu_k = 0.24/(1+k/800)$
	$A = 1,\ B = 1$
PWR:	$\mu_k = 0.20/(1+k/500)$
	$\beta_k = 0.10/(1+k/200)$
	$\lambda_k = 0.40/2^{(k/200)}$
MPLL:	$\lambda_k = 0.40/2^{(k/2000)}$
PLLs:	$\alpha_1 = 0.05,\ \alpha_2 = 10^{-5}$

Fig. 4-9. Comparison of the vector CMA detector, decorrelation CMA detector, and the PWR detector, in terms of MSE_1 versus time, for a noiseless synchronous CDMA application with 4-QAM input alphabet, assuming both actual rotators and MMSE rotators.

(MMSE) PLL or MPLL, while the upper curve corresponds to an actual PLL or MPLL implementation. The parameters for each detector were optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram, or equivalently an $\mathrm{MSE}_1 = -12$ dB. The step size for the vector CMA with GSC detector and decorrelation CMA detector ($A = 1$, $B = 1$) were $\mu_{k,\mathrm{vec}} = 0.38 \,/\, (1 + k/200)$ and $\mu_{k,\mathrm{dec}} = 0.24 \,/\, (1 + k/800)$, respectively. We used a 10-point causal rectangular window to estimate the cross-correlation terms in decorrelation CMA. The step size was $\mu_{k,\mathrm{pwr}} = 0.20 \,/\, (1 + k/500)$ for the PWR detector, it was $\beta_{k,\mathrm{agc}} = 0.1 \,/\, (1 + k/200)$ for the AGC, and it was $\lambda_k = 0.4 \,/\, 2^{(k/200)}$ for the subspace projector. The step size for the MPLL was decreased with time according to $\lambda_k = 0.4 \,/\, 2^{(k/2000)}$. The bank of scalar PLLs parameters used by decorrelation CMA were $\alpha_1 = 0.05$ and $\alpha_2 = 10^{-5}$. From the curves, we see that the speed of convergence for both the vector CMA with GSC detector and the PWR detector are nearly identical. On the other hand, the speed of convergence of the decorrelation CMA detector is quite large when compared to the other detectors. We also observe that there exists a small initial discrepancy between the actual MPLL and the MMSE MPLL; it is less than 2 dB and it quickly disappears.

In the second experiment, we examine the effect of noise on the convergence time for each detector.

Experiment 4-6. We define the *convergence time* to be the average number of symbols it takes for each detector to reach an $\mathrm{MSE}_1 = -12$ dB. In Fig. 4-10,

we plot the convergence time for each detector versus $\mathrm{SNR}_1 = \sum_{j=1}^{m} |h_{j,1}|^2/\sigma^2$. Each curve was generated by averaging over 500 different random input and noise realizations at each SNR. The step size for the vector CMA detector and decorrelation CMA detector ($A = 1$, $B = 1$) were $\mu_{k,\mathrm{vec}} = \mu_{0,\mathrm{vec}}/(1 + k/k_\mu)$ and $\mu_{k,\mathrm{dec}} = \mu_{0,\mathrm{dec}}/(1 + k/k_\mu)$, respectively. We used a 10-point causal rectangular window to estimate the cross-correlation terms in decorrelation CMA. The step size was $\mu_{0,\mathrm{pwr}}/(1 + k/k_\mu)$ for the PWR detector, it was $\beta_{k,\mathrm{agc}} = \beta_{0,\mathrm{agc}}/(1 + k/k_\beta)$ for the AGC, and it was $\lambda_k = \lambda_{0,\mathrm{sp}}/2^{(k/k_\lambda)}$ for the sub-

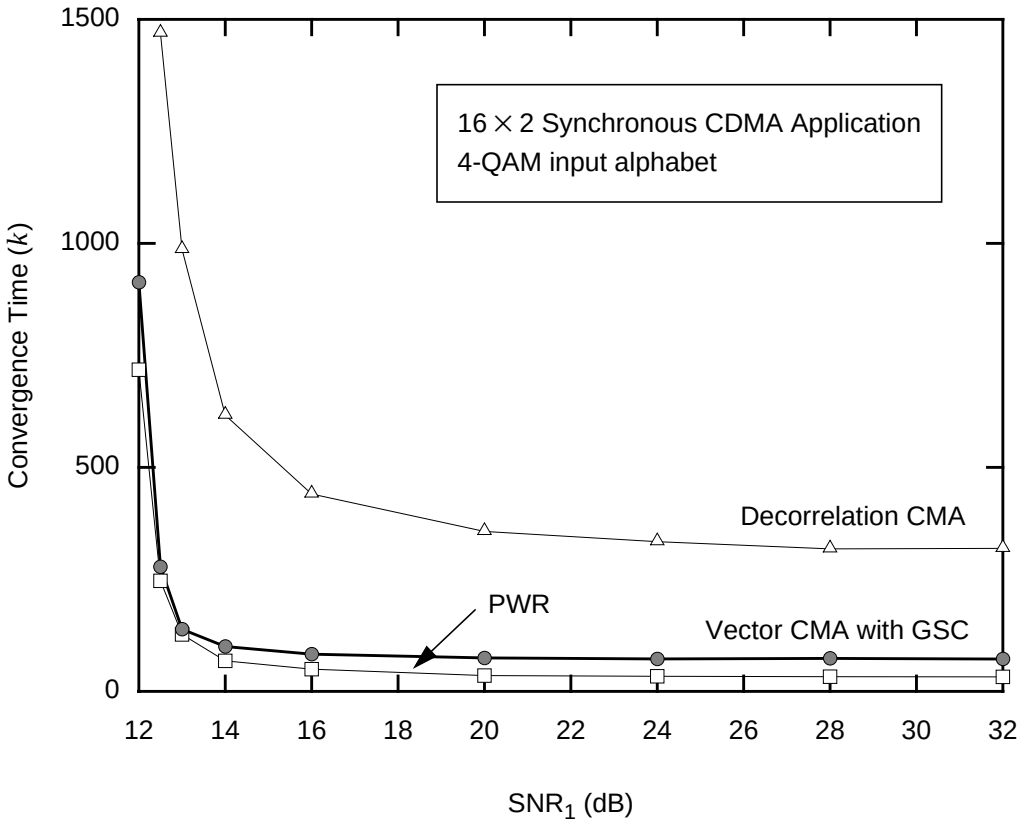


Fig. 4-10. Comparison of the convergence time of the three detector versus SNR_1 for a synchronous CDMA application. The optimal parameters for each detector can be found in Table 4-2.

space projector. The step size for the MPLL was decreased with time according to $\lambda_k = 0.4/2^{(k/2000)}$. The bank of scalar PLLs parameters, used by the two extensions of pointwise CMA, were $\alpha_1 = 0.05$ and $\alpha_2 = 10^{-5}$. The parameters for each detector were optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram at each SNR; the values of the optimal parameters are listed in Table 4-2. The figure shows that the convergence time of the PWR detector is slightly less than that of vector CMA with GSC detector. The convergence of the decorrelation CMA detector is by far the worst of the three detectors. We also observe that as the SNR increases, the convergence time for each detector becomes a constant, suggesting that there is a fundamental limit to the speed of convergence.

TABLE 4-2: Optimized Parameters for a Noisy Synchronous CDMA Application.

SNR_1 (dB)	Vector CMA	PWR	Decorrelation CMA
12.5	$\mu_0 = 0.30$, $k_\mu = 100$	$\mu_0 = 0.08$, $k_\mu = 100$ $\beta_0 = 0.05$, $k_\beta = 100$ $\lambda_0 = 0.40$, $k_\lambda = 75$	$\mu_0 = 0.10$, $k_\mu = 800$
13.0	0.32, 100	0.1, 100; 0.08, 100; 0.40, 75	0.12, 800
14.0	0.38, 100	0.1, 200; 0.1, 200; 0.40, 100	0.16, 800
16.0	0.38, 200	0.1, 200; 0.1, 200; 0.40, 200	0.20, 800
20.0 – 32.0	0.38, 200	0.2, 500; 0.1, 200; 0.40, 200	0.24, 800

This experiment suggests that either the PWR detector or the vector CMA with GSC detector can be used, since both provide reasonably fast convergence times. In the next experiment, we examine the computational complexity of each detector.

Experiment 4-7. In Fig. 4-11, we plot the total complexity for each detector versus SNR_1. It is seen that even though the total complexity of all three detectors is nearly identical, the PWR detector has the least total complexity,

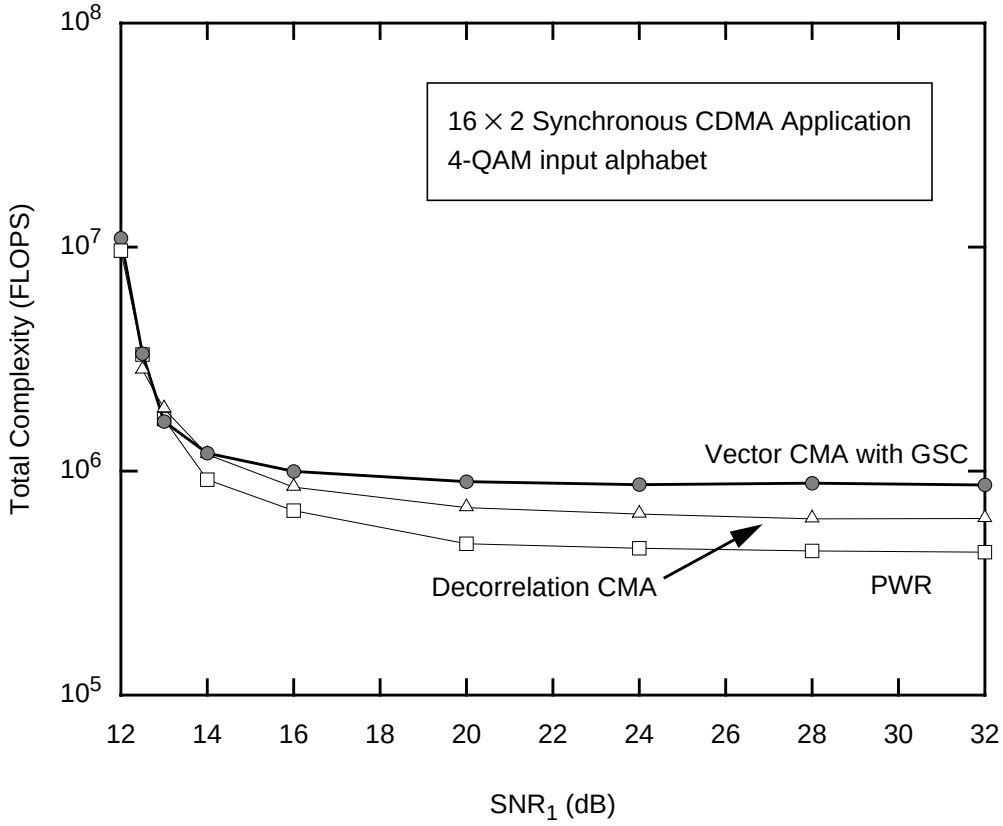


Fig. 4-11. Comparison of the total complexity of the three detector versus SNR_1 for a synchronous CDMA application.The optimal parameters for each detector can be found in Table 4-2.

followed by the decorrelation CMA detector and then by the vector CMA with GSC detector.

From Experiments 4-6 and 4-7, we can infer that, for a 16×2 synchronous CDMA application, both the PWR detector and the vector CMA with GSC detector provide a reasonable trade-off between convergence time and computational complexity. We should point out that the computational complexity of the vector CMA with GSC detector is slightly higher than that of the PWR detector because the former algorithm requires an estimate of the autocorrelation of the received vector.

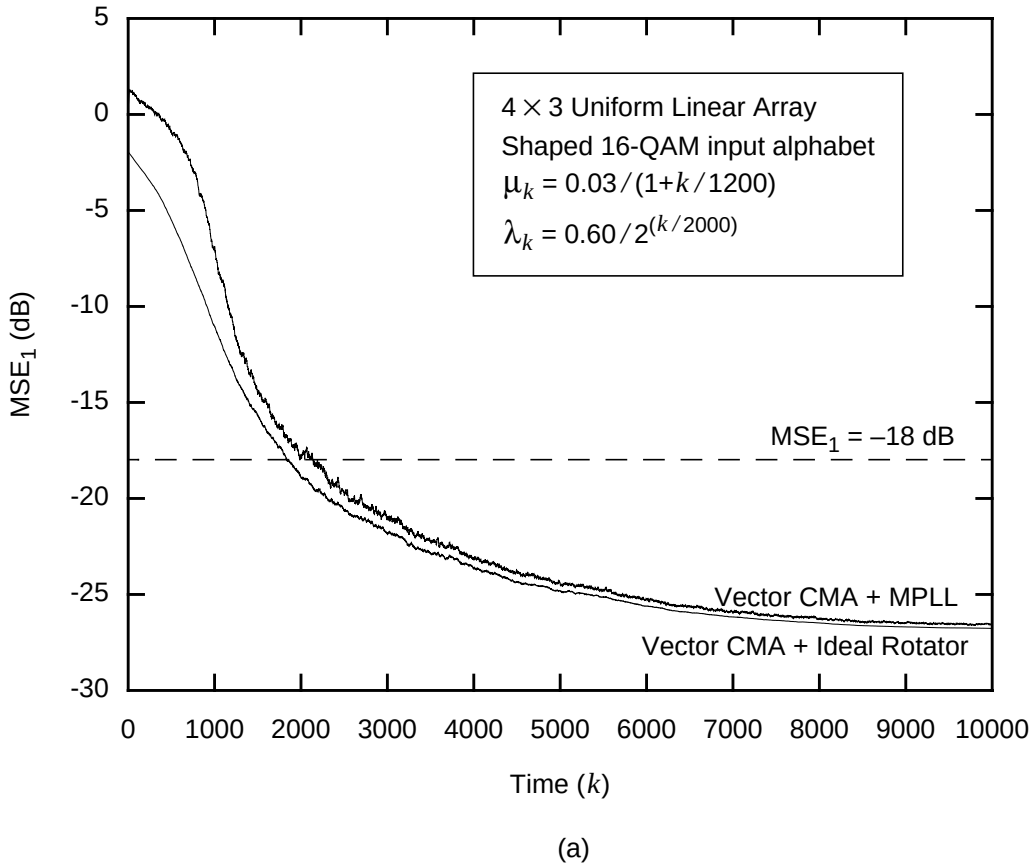
4.7.4 Shaped Constellation

In the previous experiments, we assumed that each user selects symbols uniformly from a QAM input alphabet. Thus, the probability distribution for the symbols is uniform. It is a well-known fact that it is possible to approach the Shannon capacity on an additive white-Gaussian noise (AWGN) channel by using a near-Gaussian input alphabet. One way to generate a near-Gaussian input alphabet is to shape the signal constellation such that the symbols closer to the origin have a greater probability of being transmitted than symbols farther away [112]. In the following example, we demonstrate that the vector CMA detector converges when the input constellation is shaped.

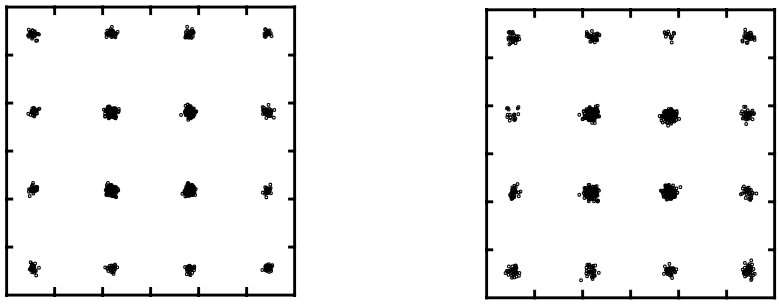
Experiment 4-8. Consider the channel described by (4-58) except with $m = 4$ sensors. We assume that each user draws symbols independently from a shaped 16-QAM constellation, which is defined as follows:

$$x_k^{(i)} = \begin{cases} \{\pm 1 \pm j\} & \text{each with probability } p(x_k) = 5/32 \\ \{\pm 3 \pm j, \pm 1 \pm 3j\} & \text{each with probability } p(x_k) = 1/32 \\ \{\pm 3 \pm 3j\} & \text{each with probability } p(x_k) = 1/32 \end{cases} . \quad (4\text{-}60)$$

It is easily verified that the kurtosis of this constellation is $\kappa = 1.89$, which is very close to the kurtosis of a complex Gaussian distribution. In Fig. 4-12(a), we plot the MSE_1 versus time for the vector CMA detector. The curves are an ensemble average of 500 different random input and noise sequences, with $SNR_1 = 28$ dB. Again, there are two curves for this detector: the lower curve corresponds to a fictitious system employing the MMSE MPLL, while the upper curve corresponds to an actual MPLL implementation. The parameter for the vector CMA detector was optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram, or equivalently an $MSE_1 = -18$ dB. The step size for the vector CMA detector was given by $\mu_{k,\text{vec}} = 0.030 / (1 + k/1200)$. The step size for the MPLL was decreased with time according to $\lambda_k = 0.6 / 2^{(k/2000)}$. From the curves, we see that the vector CMA detector converges quickly and is able to successfully open the eye diagram after only 2000 symbols. Fig. 4-12(b) shows the constellations of the last 1000 symbols from the last trial. We see that the eye diagram is indeed open and that the transmitted symbols can be recovered using a simple decision device.



(a)



(b)

Fig. 4-12. The vector CMA detector applied to a uniform linear-array with half-wavelength spacing and a shaped 16-QAM input alphabet: (a) learning curves, assuming both an actual MPLL and an MMSE MPLL; (b) constellations from the last trial, baud 9000 to 10,000.

This example clearly demonstrated that the vector CMA detector is compatible with capacity-achieving systems.

4.7.5 Computational Complexity

In the following example, we compare the computational complexity for all of the detectors.

Experiment 4-9. Consider a receiver with $m = 20$ sensors. In Fig. 4-13, we plot the number of FLOPS required per symbol for each detector with the appropriate blind unitary estimator (either the MPLL or a bank of single-user

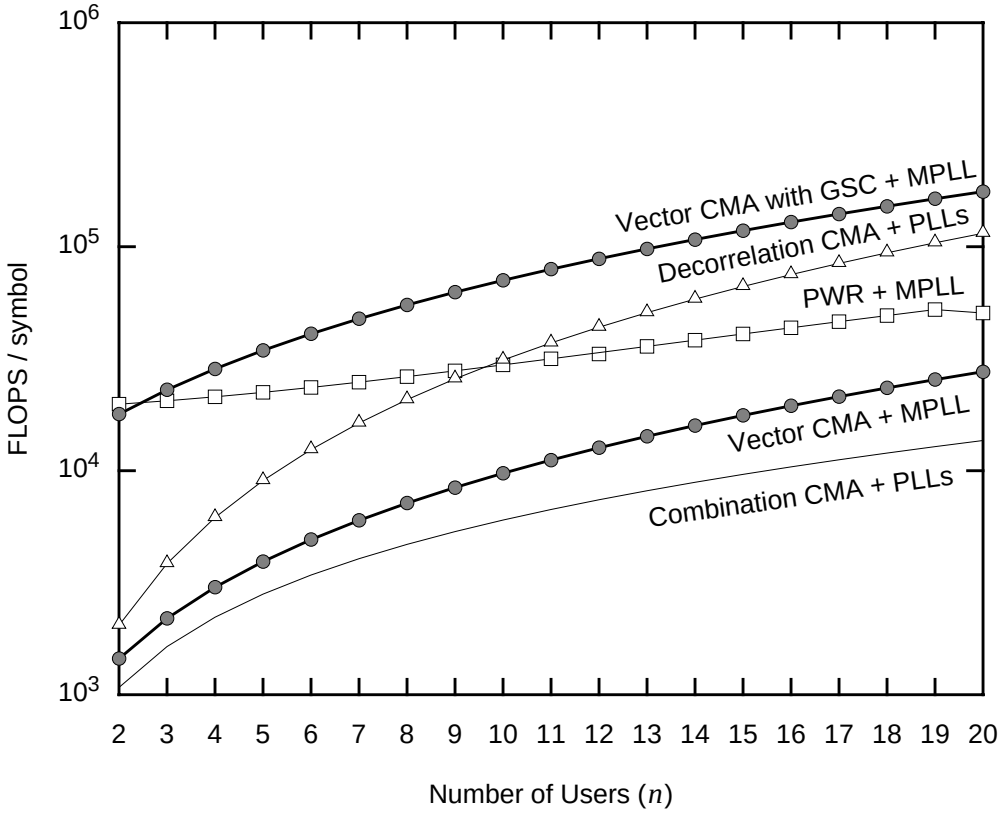


Fig. 4-13. Comparison of the computational complexity of the various detectors versus the number of users n for a fixed number of sensors $m = 20$.

PLLs) versus the number of users (n). This figure shows that the combination CMA detector with a scalar PLL bank has the lowest complexity of any detector. The vector CMA detector by itself has a lower complexity than combination CMA detector, but the MPLL, which is somewhat more complex algorithm, raises the overall complexity of this detector. The PWR detector has a nearly constant computational complexity for all values of n, because the dimension of the subspace projector also depends *only* on the number of sensors m, which is fixed in this experiment. For $n < 9$, the decorrelation CMA detector has the third lowest complexity, while for $n > 12$, it has the second highest complexity. The vector CMA with GSC detector has the largest computational complexity when $n \geq 5$. The reason that these two detectors have the highest computational complexity is because each detector requires an estimation of a system parameter.

Both computational complexity and performance of the detector are important considerations when selecting a detector. As mentioned earlier in Experiments 4-4 and 4-7, there is an inherent trade-off between performance and complexity. This trade-off must be carefully balanced when selecting a detector.

4.8 SUMMARY

We began this chapter by defining the vector CMA detector which is based on a unique multidimensional generalization of the constant-modulus algorithm. This detector inherits the most important property, the invariance to an arbitrary unitary rotation, from CMA. It

uses a linear detector with the smallest possible dimensions to recover the transmitted data. We have also included a detailed implementation of this blind detector.

One of the main results of this chapter is determination of the local minima of the vector CMA cost function. We have shown that, in the absence of noise, this cost function is minimized by only unitary matrices when the input alphabet is non-CM (see Theorem 4-3), and by both unitary and non-unitary matrices when the input alphabet is CM (see Theorem 4-4). The unitary matrices are indeed desirable because we have already developed an algorithm that can estimate and resolve the unitary ambiguity. A consequence of Theorem 4-3 is that the vector CMA detector is compatible with both nearly-Gaussian and highly shaped input alphabets and therefore it can be used on capacity-achieving systems. We have also determined the local minima of the cost function in the presence of noise. Using the noisy local minima, we were able to demonstrate that the vector CMA has near-MMSE like performance in the presence of noise.

By exploiting the properties of the desirable minima, we were able to add an additional term to the vector CMA cost function that penalized the undesirable local minima. The modified cost function was referred to as the vector CMA cost function with Gram-Schmidt constraint (GSC). We have shown that this cost function is minimized by only unitary matrices for all input alphabets, including CM input alphabets. The elimination of the undesirable local minima, however, is accompanied by an increase in complexity. Fortunately, the additional term in this cost function, which is based solely on second-order statistics, assists in reducing the convergence time. We have also included a detailed implementation for the vector CMA with GSC detector. This detector is certainly impor-

tant because, as will be shown in the next chapter, it extends quite naturally to channels with memory.

Finally, we have shown, through computer simulations, that both the vector CMA detector and the vector CMA with GSC detector compare favorably in terms of performance and complexity to other blind multiuser detectors.

APPENDIX 4.1

PROOF OF THEOREM 4-3

The noiseless vector CMA cost function can be written in terms of the channel input vector $\boldsymbol{x}$ and the overall transfer function matrix $\mathbf{F}$ as follows[2]:

$$J_V(\mathbf{F}) = E\left[(\boldsymbol{x}^*\mathbf{F}^*\mathbf{F}\boldsymbol{x})^2 - 2M_V(\boldsymbol{x}^*\mathbf{F}^*\mathbf{F}\boldsymbol{x}) + M_V{}^2 \right]. \tag{4-61}$$

Since $\boldsymbol{x}$ is a random vector sequence, (4-61) is completely parameterized by the overall transfer function matrix $\mathbf{F}$. The matrix $\mathbf{F}^*\mathbf{F}$ is Hermitian[3] and positive-semidefinite[4] matrix [106]; therefore, it has a unique eigendecomposition: $\mathbf{F}^*\mathbf{F} = \mathbf{V}\mathbf{D}\mathbf{V}^*$, where $\mathbf{V}$ is a unitary matrix and $\mathbf{D}$ is a diagonal matrix with non-negative real entries.

Let $\boldsymbol{u} = \mathbf{V}^*\boldsymbol{x}$ and let $\boldsymbol{w}$ denote an $n \times 1$ vector whose i-th component is given by

$$w_i = |u_i|^2 = |\boldsymbol{v}_i{}^*\boldsymbol{x}|^2, \tag{4-62}$$

---

2. The dependence on time has been suppressed to simplify the notation.

3. The matrix $\mathbf{G}$ is Hermitian matrix if and only if $\mathbf{G} = \mathbf{G}^*$.

4. An Hermitian matrix $\mathbf{G}$ is positive semi-definite if and only if $\boldsymbol{r}^*\mathbf{G}\boldsymbol{r} \geq 0$ for all $\boldsymbol{r} \in \mathbb{C}^n$.

where $\boldsymbol{v}_i$ is the i-th column of $\mathbf{V}$. Using these definitions, it is easy to show that $E[\boldsymbol{x}^*\mathbf{F}^*\mathbf{F}\boldsymbol{x}] = \boldsymbol{d}^T E[\boldsymbol{w}]$, where $\boldsymbol{d}$ is an $n \times 1$ vector composed of the diagonal elements of $\mathbf{D}$. The expectation of the i-th term of $\boldsymbol{w}$ is given by:

$$E[w_i] = E[\,|\,\boldsymbol{v}_i^*\boldsymbol{x}\,|^2] = \boldsymbol{v}_i^* E[\boldsymbol{x}\boldsymbol{x}^*]\boldsymbol{v}_i = m_2\,|\,\boldsymbol{v}_i\,|^2 = m_2, \tag{4-63}$$

where the third equality is due to the assumption that all users are independent and identically distributed. Hence, $E[\boldsymbol{w}] = m_2\mathbf{1}_n$, where the $n \times 1$ vector $\mathbf{1}_n = [1 \ldots 1]^T$. We can simplify (4-61) to:

$$J_v(\mathbf{V}, \boldsymbol{d}) = \boldsymbol{d}^T\mathbf{R}_{ww}\boldsymbol{d} - 2m_2M_v\boldsymbol{d}^T\mathbf{1}_n + M_v^2, \tag{4-64}$$

where $\mathbf{R}_{ww} = E[\boldsymbol{w}\boldsymbol{w}^T]$ is a function of $\mathbf{V}$.

We observe that (4-64) is completely parameterized by $\mathbf{V}$ and $\boldsymbol{d}$. Since the variables are independent, the local minima can be determined by first minimizing the cost function with respect to $\boldsymbol{d}$ and then with respect to $\mathbf{V}$. The gradient of J_v with respect to $\boldsymbol{d}$:

$$\nabla_d J_v = 2\mathbf{R}_{ww}\boldsymbol{d} - 2m_2M_v\mathbf{1}_n. \tag{4-65}$$

The optimal $\boldsymbol{d}$ occurs when (4-65) is equal to the zero vector:

$$\mathbf{R}_{ww}\boldsymbol{d} = m_2M_v\mathbf{1}_n. \tag{4-66}$$

Adding the n equations of (4-66) yields:

$$E[\|\boldsymbol{u}\|^2 \sum_{i=1}^{n} |u_i|^2 d_i] = n m_2 M_v. \tag{4-67}$$

Since $E[\|\boldsymbol{u}\|^2] = E[\|\boldsymbol{x}\|^2] = n m_2$ and $M_v = E[\|\boldsymbol{x}\|^4] / E[\|\boldsymbol{x}\|^2]$, we can rewrite (4-67) as:

$$E[\|\boldsymbol{u}\|^2 \sum_{i=1}^{n} |u_i|^2 d_i] = E[\|\boldsymbol{u}\|^4]. \tag{4-68}$$

Clearly, $\boldsymbol{d} = \mathbf{1}_n$ satisfies (4-68). Since $J_v(\mathbf{F}) \geq 0$ and (4-64) is quadratic in $\boldsymbol{d}$, $\boldsymbol{d} = \mathbf{1}_n$ is a global minimum for the vector CMA cost function. If the matrix $\mathbf{R}_{ww}$ is full rank, then the global minimum $\boldsymbol{d} = \mathbf{1}_n$ is also unique (independent of the choice of $\mathbf{V}$).

Lemma 4-1: The matrix $\mathbf{R}_{ww}$ can be written as a linear combination of three matrices:

$$\mathbf{R}_{ww} = (m_2)^2 \left[\mathbf{1}_n \mathbf{1}_n{}^T + (\mathbf{I} - \mathbf{B}^T \mathbf{B}) + (\kappa - 1)\mathbf{B}^T \mathbf{B} \right], \tag{4-69}$$

where $[\mathbf{B}]_{ij} = |v_{ij}|^2$.

Proof: See Appendix 4.2.

Lemma 4-2: If $\kappa > 1$, then the matrix $\mathbf{R}_{ww}$ is a positive definite.

Proof: See Appendix 4.3.

Since $\mathbf{R}_{ww}$ is positive definite, it must also be full rank.Therefore, $\boldsymbol{d} = \mathbf{1}_n$ is the only solution to (4-66) when the input alphabet is non-CM. In fact, this solution is also the unique global minimum. It is seen that when $\boldsymbol{d} = \mathbf{1}_n$, $\mathbf{F}^*\mathbf{F} = \mathbf{I}$. Thus, the vector CMA cost function is minimized if and only if $\mathbf{F}$ is unitary. ❑

APPENDIX 4.2

PROOF OF LEMMA 4-1

Recall that the i-th component of $\boldsymbol{w}$ is given by:

$$w_i = |\boldsymbol{v}_i^* \boldsymbol{x}|^2 = \left| \sum_{l=1}^{n} v_{li}^* x_l \right|^2. \tag{4-70}$$

Thus, the (i,j)-th component of $\mathbf{R}_{ww}$ can be written as:

$$[\mathbf{R}_{ww}]_{ij} = E[w_i w_j^*], \tag{4-71}$$

$$= E\left[\left| \sum_{l=1}^{n} v_{li}^* x_l \right|^2 \right] \left[\left| \sum_{p=1}^{n} v_{pj}^* x_p \right|^2 \right], \tag{4-72}$$

$$= \sum_{l=1}^{n} \sum_{m=1}^{n} \sum_{p=1}^{n} \sum_{q=1}^{n} v_{li}^* v_{mi} v_{pj}^* v_{qj} E[x_l x_m^* x_p x_q^*]. \tag{4-73}$$

We can use the following identity: $E[x_k x_m^* x_p x_q^*] = (m_2)^2[\delta_{km}\delta_{pq}+\delta_{kq}\delta_{mp}+(\kappa-2)\delta_{kmpq}]$, to simplify (4-73):

$$[\mathbf{R}_{ww}]_{ij} = (m_2)^2\Bigg[\sum_{l=1}^{n}|v_{li}|^2\sum_{p=1}^{n}|v_{pj}|^2 + \sum_{l=1}^{n}v_{lj}v_{li}^{*}\sum_{p=1}^{n}v_{pi}v_{pj}^{*} + (\kappa-2)\sum_{l=1}^{n}|v_{li}|^2|v_{lj}|^2\Bigg], \tag{4-74}$$

$$= (m_2)^2\Bigg[1 + \delta_{ij} + (\kappa-2)\Bigg(\sum_{l=1}^{n}|v_{li}|^2|v_{lj}|^2\Bigg)\Bigg], \tag{4-75}$$

$$= (m_2)^2\Big[[\mathbf{1}_n\mathbf{1}_n{}^T]_{ij} + [\mathbf{I}]_{ij} + (\kappa-2)[\mathbf{B}^T\mathbf{B}]_{ij}\Big], \tag{4-76}$$

$$= (m_2)^2\Big[[\mathbf{1}_n\mathbf{1}_n{}^T]_{ij} + [\mathbf{I}-\mathbf{B}^T\mathbf{B}]_{ij} + (\kappa-1)[\mathbf{B}^T\mathbf{B}]_{ij}\Big], \tag{4-77}$$

where we have defined $[\mathbf{B}]_{ij} = |v_{ij}|^2$.

Hence, $\mathbf{R}_{ww}$ can be expressed in terms of its elemental component as follows:

$$\mathbf{R}_{ww} = (m_2)^2\Big[\mathbf{1}_n\mathbf{1}_n{}^T + (\mathbf{I}-\mathbf{B}^T\mathbf{B}) + (\kappa-1)\mathbf{B}^T\mathbf{B}\Big], \tag{4-78}$$

where $[\mathbf{B}]_{ij} = |v_{ij}|^2$. ❑

APPENDIX 4.3

PROOF OF LEMMA 4-2

First, we show that $\mathbf{R}_{ww}$ is a positive-semidefinite matrix. Recall from (4-69) that $\mathbf{R}_{ww}$ can be written in terms of its elemental components as follows:

$$\mathbf{R}_{ww} = (m_2)^2\Big[\mathbf{1}_n\mathbf{1}_n{}^T + (\mathbf{I} - \mathbf{B}^T\mathbf{B}) + (\kappa-1)\mathbf{B}^T\mathbf{B}\Big], \tag{4-79}$$

where $[\mathbf{B}]_{ij} = |v_{ij}|^2$. Clearly, the matrix $\mathbf{1}_n\mathbf{1}_n{}^T$ is positive semidefinite because its eigenvalues are zero, with multiplicity n–1, and n. The matrix $(\kappa-1)\mathbf{B}^T\mathbf{B}$ is also positive semidefinite because $\boldsymbol{r}^*\mathbf{B}^T\mathbf{B}\boldsymbol{r} = \|\mathbf{B}\boldsymbol{r}\|^2 \geq 0$ for all $\boldsymbol{r} \in \mathbb{C}^n$ and $(\kappa-1) > 0$ for a non-CM input alphabets.

The matrix $(\mathbf{I} - \mathbf{B}^T\mathbf{B})$ is positive semidefinite if $\boldsymbol{r}^*(\mathbf{I} - \mathbf{B}^T\mathbf{B})\boldsymbol{r} = \|\boldsymbol{r}\|^2 - \|\mathbf{B}\boldsymbol{r}\|^2 \geq 0$ for all $\boldsymbol{r} \in \mathbb{C}^n$. Observe that:

$$\|\mathbf{B}\boldsymbol{r}\|^2 = \sum_{i=1}^{n}\left|\sum_{j=1}^{n}|v_{ij}|^2 r_j\right|^2. \tag{4-80}$$

Define $p_j = |v_{ij}|^2$ and observe that $\sum_{j=1}^{n} p_j = 1$. We can view the sum $\sum_{j=1}^{n} p_j r_j$ as an expectation $E[R]$, where R is a random variable over the set $\{r_j\}$ with probability mass

function $\{p_j\}$. From Jensen's inequality [Cover & Thomas], the square of the mean cannot exceed the second moment:

$$\left| \sum_{j=1}^{n} |v_{ij}|^2 r_j \right|^2 = |\mathrm{E}[R]|^2 \le \mathrm{E}[|R|^2] = \sum_{j=1}^{n} |v_{ij}|^2 |r_j|^2. \tag{4-81}$$

Hence, (4-80) is upper-bounded by:

$$\|\mathbf{B}\boldsymbol{r}\|^2 \le \sum_{i=1}^{n} \sum_{j=1}^{n} |v_{ij}|^2 |r_j|^2 = \sum_{j=1}^{n} |r_j|^2 \sum_{i=1}^{n} |v_{ij}|^2 = \sum_{j=1}^{n} |r_j|^2 = \|\boldsymbol{r}\|^2, \tag{4-82}$$

where $\sum_{i=1}^{n} |v_{ij}|^2 = 1$ because the columns of $\mathbf{V}$ have unit length. This equation implies that $\|\boldsymbol{r}\|^2 - \|\mathbf{B}\boldsymbol{r}\|^2 \ge 0$ for all $\boldsymbol{r} \in \mathbb{C}^n$ and so $(\mathbf{I} - \mathbf{B}^T\mathbf{B})$ is a positive-semidefinite matrix. Since the sum of positive-semidefinite matrices is positive semidefinite [106], $\mathbf{R}_{ww}$ is a positive-semidefinite matrix.

The matrix $\mathbf{R}_{ww}$ can be either singular or positive definite. If we assume this matrix to be singular, then there exists a nonzero vector $\boldsymbol{r}$ such that:

$$\boldsymbol{r}^*\mathbf{R}_{ww}\boldsymbol{r} = 0 \Leftrightarrow (m_2)^2 \boldsymbol{r}^* \left[\mathbf{1}_n \mathbf{1}_n{}^T + (\mathbf{I} - \mathbf{B}^T\mathbf{B}) + (\kappa - 1)\mathbf{B}^T\mathbf{B} \right] \boldsymbol{r} = 0, \tag{4-83}$$

$$\Leftrightarrow (m_2)^2 \left[|\mathbf{1}_n{}^T\boldsymbol{r}|^2 + (\|\boldsymbol{r}\|^2 - \|\mathbf{B}\boldsymbol{r}\|^2) + (\kappa - 1)\|\mathbf{B}\boldsymbol{r}\|^2 \right] = 0, \tag{4-84}$$

$$\Leftrightarrow |\mathbf{1}_n{}^T\boldsymbol{r}|^2 = 0 \text{ and } \|\boldsymbol{r}\|^2 = \|\mathbf{B}\boldsymbol{r}\|^2 \text{ and } \|\mathbf{B}\boldsymbol{r}\|^2 = 0, \tag{4-85}$$

where the third implication is due to the fact that each term in (4-84) is non-negative. From (4-85), we see that the last two conditions imply that $\|\boldsymbol{r}\|^2 = 0$, which is impossible since we assumed that $\boldsymbol{r}$ is nonzero. Therefore, $\mathbf{R}_{ww}$ is never singular and thus, must be positive definite. ❑

APPENDIX 4.4

PROOF OF THEOREM 4-4

For a CM input alphabet ($\kappa = 1$), (4-69) reduces to:

$$\mathbf{R}_{ww} = (m_2)^2 \left[\mathbf{1}_n \mathbf{1}_n{}^T + (\mathbf{I} - \mathbf{B}^T\mathbf{B}) \right]. \tag{4-86}$$

In Appendix 4.3, we saw that $\mathbf{1}_n\mathbf{1}_n{}^T$ and $(\mathbf{I} - \mathbf{B}^T\mathbf{B})$ are both positive-semidefinite matrices; hence, $\mathbf{R}_{ww}$ is also a positive-semidefinite matrix. The matrix $\mathbf{R}_{ww}$ can either be singular or full rank, depending on the exact value of $\mathbf{V}$. If $\mathbf{R}_{ww}$ is full rank, then $\boldsymbol{d} = \mathbf{1}_n$ is the only solution that satisfies (4-66). (It is independent of the choice of $\mathbf{V}$.) However, if $\mathbf{R}_{ww}$ is singular, then the solution to (4-66) depends upon the choice of $\mathbf{V}$. The rank of $\mathbf{R}_{ww}$ is summarized in the following lemma.

Lemma 4-3: If $\kappa = 1$, then $\mathbf{R}_{ww}$ is singular if

$$\mathbf{V} = \mathbf{P}_L \begin{bmatrix} \mathbf{Q}_1 & & \mathbf{0} \\ & \mathbf{Q}_2 & \\ & & \ddots \\ \mathbf{0} & & \mathbf{Q}_P \end{bmatrix} \mathbf{P}_R, \tag{4-87}$$

where $\mathbf{Q}_i$ is a $k_i \times k_i$ unitary matrix with $k_i \in \{1, \ldots, n-1\}$ satisfying $\sum_{i=1}^{P} k_i = n$, and where $\mathbf{P}_L$ and $\mathbf{P}_R$ are real permutation matrices; otherwise, $\mathbf{R}_{ww}$ is nonsingular.

Proof: See Appendix 4.5.

In Appendix 4.1, we showed that $\boldsymbol{d} = \mathbf{1}_n$ is a solution to (4-66) regardless of the rank of $\mathbf{R}_{ww}$. If $\mathbf{R}_{ww}$ is singular, then any vector in the nullspace of this matrix added to $\boldsymbol{d} = \mathbf{1}_n$ is also a solution. In Appendix 4.5, we show that the nullspace of $\mathbf{R}_{ww}$ must have the form:

$$\boldsymbol{r} = \mathbf{P}_R{}^T \begin{bmatrix} \alpha_1 \mathbf{1}_{k_1} \\ \alpha_2 \mathbf{1}_{k_2} \\ \vdots \\ \alpha_P \mathbf{1}_{k_P} \end{bmatrix}, \tag{4-88}$$

where α_i is a nonzero constant and $k_i \in \{1, \ldots, n-1\}$ satisfying the relations $\sum_{i=1}^{P} k_i = n$, and $\sum_{i=1}^{P} \alpha_i k_i = 0$. Thus, when $\mathbf{R}_{ww}$ is singular, the solution to (4-66) is given by:

$$\boldsymbol{d} = \mathbf{1}_n + \mathbf{P}_R{}^T \begin{bmatrix} \alpha_1 \mathbf{1}_{k_1} \\ \alpha_2 \mathbf{1}_{k_2} \\ \vdots \\ \alpha_P \mathbf{1}_{k_P} \end{bmatrix}. \tag{4-89}$$

Observe that this vector satisfies the following property:

$$\sum_{i=1}^{n} d_i = n, \tag{4-90}$$

or equivalently, $tr(\mathbf{D}) = n$.

In summary, when $\mathbf{R}_{ww}$ is singular, the vector CMA cost function is minimized if and only if $\mathbf{F}^*\mathbf{F} = \mathbf{V}\mathbf{D}\mathbf{V}^*$, where $\mathbf{V}$ is given by (4-87) and where $\mathbf{D}$ is a non-negative diagonal matrix satisfying $tr(\mathbf{D}) = n$. By expanding the product $\mathbf{F}^*\mathbf{F}$, it easy to show that $\mathbf{F}^*\mathbf{F} = \tilde{\mathbf{D}}$, where $\tilde{\mathbf{D}}$ is a diagonal matrix, whose diagonal entries are a possibly reordered version of the diagonal entries of $\mathbf{D}$. Hence, the vector CMA cost function is minimized if and only if $\mathbf{F}^*\mathbf{F} = \tilde{\mathbf{D}}$, where $\tilde{\mathbf{D}}$ is a positive diagonal matrix satisfying $tr(\tilde{\mathbf{D}}) = n$.

In both cases, whether $\mathbf{R}_{ww}$ is either singular or nonsingular, the vector CMA cost function is minimized when $\mathbf{F}^*\mathbf{F} = \mathbf{D}$, where $\mathbf{D}$ is a positive diagonal matrix satisfying $tr(\mathbf{D}) = n$. Therefore, the optimal overall transfer function matrix is given by $\mathbf{F} = \mathbf{U}\mathbf{D}^{1/2}$, where $\mathbf{U}$ is a unitary matrix and $\mathbf{D}$ is a positive diagonal matrix satisfying $tr(\mathbf{D}) = n$. ❑

APPENDIX 4.5

PROOF OF LEMMA 4-3

Since $\mathbf{R}_{ww}$ is a positive-semidefinite matrix for a CM input alphabet, it can either be singular or nonsingular. The matrix $\mathbf{R}_{ww}$ is singular if and only if there exists a nonzero vector $\boldsymbol{r} \in \mathbb{C}^n$ such that:

$$\boldsymbol{r}^* \mathbf{R}_{ww} \boldsymbol{r} = 0 \Leftrightarrow (m_2)^2 \boldsymbol{r}^* \left[\mathbf{1}_n \mathbf{1}_n{}^T + (\mathbf{I} - \mathbf{B}^T \mathbf{B}) \right] \boldsymbol{r} = 0, \tag{4-91}$$

$$\Leftrightarrow (m_2)^2 \left[| \mathbf{1}_n{}^T \boldsymbol{r} |^2 + (\|\boldsymbol{r}\|^2 - \|\mathbf{B}\boldsymbol{r}\|^2) \right] = 0, \tag{4-92}$$

$$\Leftrightarrow \mathbf{1}_n{}^T \boldsymbol{r} = 0 \text{ and } \|\mathbf{B}\boldsymbol{r}\|^2 = \|\boldsymbol{r}\|^2. \tag{4-93}$$

In Appendix 4.3, we showed that

$$\|\mathbf{B}\boldsymbol{r}\|^2 = \sum_{i=1}^{n} \left| \sum_{j=1}^{n} | v_{ij} |^2 r_j \right|^2 \leq \sum_{i=1}^{n} \sum_{j=1}^{n} | v_{ij} |^2 | r_j |^2 = \|\boldsymbol{r}\|^2. \tag{4-94}$$

Equality in (4-94) is achieved when the random variable R is no longer random; but is deterministic. This random variable becomes deterministic when the components of $\boldsymbol{r}$ are

equal to a constant for all nonzero entries on the i-th row of $\mathbf{V}$. For example, R is deterministic when $\boldsymbol{r} = \alpha\mathbf{1}_n$ for some nonzero constant α. Unfortunately, $\boldsymbol{r} = \alpha\mathbf{1}_n$ violates the first condition of (4-93), and therefore this vector does not lie in the nullspace of $\mathbf{R}_{ww}$.

If $\mathbf{V}$ is a block-diagonal matrix, then the vector $\boldsymbol{r}$ can be subdivided into disjoint subvectors corresponding to the nonzero blocks of $\mathbf{V}$. In this case, the random variable R is deterministic if the components of each subvector are equal to some nonzero constant. If there are at least two subvectors, then we can choose the constant for each subvector in such a way that $\mathbf{1}_n{}^T\boldsymbol{r} = 0$. Since R is deterministic and the vector $\boldsymbol{r}$ sums to zero, (4-93) is satisfied and therefore $\mathbf{R}_{ww}$ is singular.

With this information, we see that $\mathbf{R}_{ww}$ is singular if and only if the matrix $\mathbf{V}$ has the form:

$$\mathbf{V} = \mathbf{P}_L \begin{bmatrix} \mathbf{Q}_1 & & & \mathbf{0} \\ & \mathbf{Q}_2 & & \\ & & \ddots & \\ \mathbf{0} & & & \mathbf{Q}_P \end{bmatrix} \mathbf{P}_R, \tag{4-95}$$

where $\mathbf{Q}_i$ is a $k_i \times k_i$ unitary matrix with $k_i \in \{1, \ldots, n-1\}$ satisfying $\sum_{i=1}^{P} k_i = n$ and where $\mathbf{P}_L$ and $\mathbf{P}_R$ are $(M+N+1)n \times (M+N+1)n$ and $n \times n$ real permutation matrices, respectively. The vector $\boldsymbol{r}$ that lies in the nullspace of $\mathbf{R}_{ww}$ has the form:

$$\boldsymbol{r} = \mathbf{P}_R{}^T \begin{bmatrix} \alpha_1\mathbf{1}_{k_1} \\ \alpha_2\mathbf{1}_{k_2} \\ \vdots \\ \alpha_P\mathbf{1}_{k_P} \end{bmatrix}, \tag{4-96}$$

where α_i is a nonzero constant satisfying the constraint that $\sum_{i=1}^{P} \alpha_i k_i = 0$.

In summary, $\mathbf{R}_{ww}$ is singular if and only if $\mathbf{V}$ is given by (4-95). ❑

APPENDIX 4.6

PROOF OF THEOREM 4-5

The noisy vector CMA cost function can be expressed in terms of the $n \times 1$ channel input vector $\boldsymbol{x}$, the $n \times 1$ noise vector $\boldsymbol{n}$, the $n \times n$ overall transfer function matrix $\mathbf{F}$, and the $n \times m$ linear detector $\mathbf{C}$ as follows:

$$J_V = E\left[(\boldsymbol{x}^*\mathbf{F}^*\mathbf{F}\boldsymbol{x})^2 + (\boldsymbol{n}^*\mathbf{C}^*\mathbf{C}\boldsymbol{n})^2 + 2(\boldsymbol{x}^*\mathbf{F}^*\mathbf{F}\boldsymbol{x})(\boldsymbol{n}^*\mathbf{C}^*\mathbf{C}\boldsymbol{n}) + 2(\boldsymbol{x}^*\mathbf{F}^*\mathbf{C}\boldsymbol{n}\boldsymbol{n}^*\mathbf{C}^*\mathbf{F}\boldsymbol{x}) \right]$$

$$- 2M_V E\left[\boldsymbol{x}^*\mathbf{F}^*\mathbf{F}\boldsymbol{x} + \boldsymbol{n}^*\mathbf{C}^*\mathbf{C}\boldsymbol{n} \right] + M_V{}^2. \qquad (4\text{-}97)$$

Recall from (4-24) that the overall transfer function matrix $\mathbf{F}$ is related to the linear detector $\mathbf{C}$ by the following relationship: $\mathbf{F} = \mathbf{CH}$, where $\mathbf{H}$ is the $m \times n$ channel matrix. Both the channel matrix and the linear detector can be expressed in terms of a unique truncated singular-value decomposition: $\mathbf{H} = \mathbf{U}\Sigma\mathbf{Q}^*$ and $\mathbf{C} = \mathbf{Q}\Lambda\mathbf{W}^*$, where $\mathbf{U}$ and $\mathbf{W}$ are $m \times n$ truncated unitary matrices, $\mathbf{Q}$ is an $n \times n$ unitary matrix, and Σ and Λ are $n \times n$ real non-negative diagonal matrices. Using these definitions, we find that $\mathbf{F}^*\mathbf{F} = \mathbf{Q}\Sigma\mathbf{G}\Sigma\mathbf{Q}^*$ and that $\mathbf{C}^*\mathbf{C} = \mathbf{U}\mathbf{G}\mathbf{U}^*$, where the $n \times n$ matrix $\mathbf{G} = \mathbf{U}^*\mathbf{W}\Lambda\mathbf{W}^*\mathbf{U}$.

Let $\boldsymbol{s} = \Sigma\mathbf{Q}^*\boldsymbol{x}$ and $\boldsymbol{w} = \mathbf{U}^*\boldsymbol{n}$. Using these definitions, we can rewrite (4-97) as:

$$J_v = E\Big[(\boldsymbol{s}^*\mathbf{G}\boldsymbol{s})^2 + (\boldsymbol{w}^*\mathbf{G}\boldsymbol{w})^2 + 2(\boldsymbol{s}^*\mathbf{G}\boldsymbol{s})(\boldsymbol{w}^*\mathbf{G}\boldsymbol{w}) + 2(\boldsymbol{s}^*\mathbf{G}\boldsymbol{w}\boldsymbol{w}^*\mathbf{G}\boldsymbol{s}) \Big]$$

$$- 2M_v E\Big[\boldsymbol{s}^*\mathbf{G}\boldsymbol{s} + \boldsymbol{w}^*\mathbf{G}\boldsymbol{w} \Big] + M_v{}^2. \qquad (4\text{-}98)$$

This equation is seen to be completely parameterized by the matrix **G**. By unwrapping the matrix **G** into a column vector $\boldsymbol{g}$ = U(**G**), expanding the products, and taking the expectation, we find that (4-98) reduces to:

$$J_v = \boldsymbol{g}^*\Big[(m_2)^2 U(\Sigma^2)U(\Sigma^2)^T + (m_2)^2\Sigma_\sigma^2 + (m_2)^2(\kappa-2)\mathbf{W}^*\mathbf{W} + \sigma^4 U(\mathbf{I}_n)U(\mathbf{I}_n)^T + \sigma^4\mathbf{I}_{n^2}$$

$$+ 2m_2\sigma^2 U(\Sigma^2)U(\mathbf{I}_n)^T + 2m_2\sigma^2\Sigma_B^2 \Big]\boldsymbol{g} - 2M_v\boldsymbol{g}^*\Big[m_2 U(\Sigma^2) + \sigma^2 U(\mathbf{I}_n) \Big] + M_v{}^2, \qquad (4\text{-}99)$$

where

$$\Sigma_\sigma^2 = \begin{bmatrix} \sigma_1^2\Sigma^2 & & & 0 \\ & \sigma_2^2\Sigma^2 & & \\ & & \ddots & \\ 0 & & & \sigma_n^2\Sigma^2 \end{bmatrix} \text{ and } \Sigma_B^2 = \begin{bmatrix} \Sigma^2 & & 0 \\ & \Sigma^2 & \\ & & \ddots \\ 0 & & \Sigma^2 \end{bmatrix}. \qquad (4\text{-}100)$$

The $n \times n^2$ matrix **W** is defined as follows[5]:

$$\mathbf{W} = [\sigma_1{}^2\boldsymbol{q}_1{}^*\otimes\boldsymbol{q}_1, \ldots, \sigma_1\sigma_n\boldsymbol{q}_1{}^*\otimes\boldsymbol{q}_n, \ldots, \sigma_1\sigma_n\boldsymbol{q}_n{}^*\otimes\boldsymbol{q}_1, \ldots, \sigma_n{}^2\boldsymbol{q}_n{}^*\otimes\boldsymbol{q}_n], \qquad (4\text{-}101)$$

5. For this particular equation, '*' represents a complex conjugate.

where σ_i is the *i*-th diagonal element of Σ, $\boldsymbol{q}_i$ is the *i*-th column of $\mathbf{Q}$, and '$\otimes$' represents the Hadamard product (component-by-component product) [HJ]. The derivation of (4-99) can be found in Appendix 4.7. As before, this cost function is completely parameterized by the vector $\boldsymbol{g}$.

The local minima of the noisy vector CMA cost function can be found by taking the gradient of (4-99) with respect to $\boldsymbol{g}$:

$$\nabla_g J_v = \Big[2(m_2)^2 U(\Sigma^2)U(\Sigma^2)^T + 2(m_2)^2\Sigma_\sigma^2 + 2(m_2)^2(\kappa-2)\mathbf{W}^*\mathbf{W} + 2\sigma^4 U(\mathbf{I}_n)U(\mathbf{I}_n)^T$$
$$+ 2\sigma^4\mathbf{I}_{n^2} + 2m_2\sigma^2 U(\Sigma^2)U(\mathbf{I}_n)^T + 2m_2\sigma^2 U(\mathbf{I}_n)U(\Sigma^2)^T + 4m_2\sigma^2\Sigma_B^2 \Big]\boldsymbol{g}$$
$$- 2M_v\Big[m_2 U(\Sigma^2) + \sigma^2 U(\mathbf{I}_n) \Big]. \qquad (4\text{-}102)$$

The inflection points of the cost function occur when the gradient is equal to the zero vector; or equivalently when,

$$\boldsymbol{g} = M_v\Big[(m_2)^2 U(\Sigma^2)U(\Sigma^2)^T + (m_2)^2\Sigma_\sigma^2 + (m_2)^2(\kappa-2)\mathbf{W}^*\mathbf{W} + \sigma^4 U(\mathbf{I}_n)U(\mathbf{I}_n)^T + \sigma^4\mathbf{I}_{n^2}$$
$$+ m_2\sigma^2 U(\Sigma^2)U(\mathbf{I}_n)^T + m_2\sigma^2 U(\mathbf{I}_n)U(\Sigma^2)^T + 2m_2\sigma^2\Sigma_B^2 \Big]^{-1}\Big[m_2 U(\Sigma^2) + \sigma^2 U(\mathbf{I}_n) \Big]. \qquad (4\text{-}103)$$

If we map the column vector $\boldsymbol{g}$ back onto the matrix $\mathbf{G}$, we find that the noisy vector CMA cost function is minimized if and only if $\mathbf{F}^*\mathbf{F} = \mathbf{Q}\Sigma U^{-1}(\boldsymbol{g})\Sigma\mathbf{Q}^*$, where $\boldsymbol{g}$ is defined by (4-100), (4-101), and (4-103).

Recall from Appendix 4.1 that $\mathbf{F}^*\mathbf{F}$ has a unique eigendecomposition: $\mathbf{F}^*\mathbf{F} = \mathbf{V}\mathbf{D}\mathbf{V}^*$, where $\mathbf{V}$ is an $n \times n$ unitary matrix and $\mathbf{D}$ is an $n \times n$ non-negative real diagonal matrix. Relating the two forms of $\mathbf{F}^*\mathbf{F}$, we find that

$$\mathbf{V}\mathbf{D}\mathbf{V}^* = \mathbf{Q}\Sigma\mathcal{U}^{-1}(\boldsymbol{g})\Sigma\mathbf{Q}^*. \tag{4-104}$$

The overall transfer function matrix that minimizes the noisy vector CMA cost function can be found by taking the square-root of the eigendecomposition: $\mathbf{F} = \mathbf{U}\mathbf{D}^{1/2}\mathbf{V}^*$, where $\mathbf{U}$ is an arbitrary $n \times n$ unitary matrix. The corresponding linear detector that minimizes the noisy vector CMA cost function is given by $\mathbf{C} = \mathbf{F}\mathbf{H}^\dagger$, where $\mathbf{H}^\dagger = (\mathbf{H}^*\mathbf{H})^{-1}\mathbf{H}^*$. ❑

APPENDIX 4.7

DERIVATION OF (4-99)

The noisy vector CMA cost function is written as follows (see (4-98)):

$$J_v = E\left[(\boldsymbol{s}^*\mathbf{G}\boldsymbol{s})^2 + (\boldsymbol{w}^*\mathbf{G}\boldsymbol{w})^2 + 2(\boldsymbol{s}^*\mathbf{G}\boldsymbol{s})(\boldsymbol{w}^*\mathbf{G}\boldsymbol{w}) + 2(\boldsymbol{s}^*\mathbf{G}\boldsymbol{w}\boldsymbol{w}^*\mathbf{G}\boldsymbol{s}) \right]$$

$$- 2M_v E\left[\boldsymbol{s}^*\mathbf{G}\boldsymbol{s} + \boldsymbol{w}^*\mathbf{G}\boldsymbol{w} \right] + M_v^2, \tag{4-105}$$

where $\boldsymbol{s} = \Sigma\mathbf{Q}^*\boldsymbol{x}$, $\boldsymbol{w} = \mathbf{U}^*\boldsymbol{n}$, $\mathbf{U}$ is an $m \times n$ truncated unitary matrix, $\mathbf{Q}$ is an $n \times n$ unitary matrix, and Σ is an $n \times n$ non-negative real diagonal matrix. The first term of (4-105) can be expanded as follows:

$$E\left[(\boldsymbol{s}^*\mathbf{G}\boldsymbol{s})^2 \right] = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{l=1}^{n}\sum_{p=1}^{n} g_{ij}g_{lp}E[s_i^* s_j s_l^* s_p], \tag{4-106}$$

where

$$E[s_i^* s_j s_l^* s_p] = \sigma_i\sigma_j\sigma_l\sigma_p \sum_{a=1}^{n}\sum_{b=1}^{n}\sum_{c=1}^{n}\sum_{d=1}^{n} q_{ai}q_{bj}^* q_{cl}q_{dp}^* E[x_a^* x_b x_c^* x_d]. \tag{4-107}$$

Using the fact that $E[x_a^* x_b x_c^* x_d] = (m_2)^2[\delta_{ab}\delta_{cd} + \delta_{ad}\delta_{bc} + (\kappa-2)\delta_{abcd}]$, we simplify (4-107) as follows:

$$E[s_i^* s_j s_l^* s_p] = (m_2)^2\sigma_i\sigma_j\sigma_l\sigma_p\Big[\sum_{a=1}^{n} q_{ai}q_{aj}^* \sum_{c=1}^{n} q_{cl}q_{cp}^* + \sum_{a=1}^{n} q_{ai}q_{ap}^* \sum_{c=1}^{n} q_{cl}q_{cj}^* + (\kappa-2)\sum_{a=1}^{n} q_{ai}q_{aj}^* q_{al}q_{ap}^*\Big]. \quad (4\text{-}108)$$

$$= (m_2)^2\Big[\sigma_i^2\sigma_l^2\delta_{ij}\delta_{lp} + \sigma_i^2\sigma_l^2\delta_{ip}\delta_{jl} + \sigma_i\sigma_j\sigma_l\sigma_p(\kappa-2)\sum_{a=1}^{n} q_{ai}q_{aj}^* q_{al}q_{ap}^*\Big]. \quad (4\text{-}109)$$

Substituting (4-109) into (4-106), we obtain:

$$E\Big[(\boldsymbol{s}^*\mathbf{G}\boldsymbol{s})^2\Big] = (m_2)^2\Big[\sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_i^2\sigma_l^2 g_{ii}g_{jj} + \sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_i^2\sigma_l^2 |g_{ij}|^2 + (\kappa-2)\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{l=1}^{n}\sum_{p=1}^{n}\sum_{a=1}^{n} g_{ij}g_{lp}\sigma_i\sigma_j\sigma_l\sigma_p q_{ai}q_{aj}^* q_{al}q_{ap}^*\Big]. \quad (4\text{-}110)$$

This equation can be expressed in terms of the column vector $\boldsymbol{g} = U(\mathbf{G})$ as follows:

$$E\Big[(\boldsymbol{s}^*\mathbf{G}\boldsymbol{s})^2\Big] = (m_2)^2\boldsymbol{g}^*\Big[U(\Sigma^2)U(\Sigma^2)^T + \Sigma_\sigma^2 + (\kappa-2)\mathbf{W}^*\mathbf{W}\Big]\boldsymbol{g}, \quad (4\text{-}111)$$

where

$$\Sigma_{\sigma}^{2} = \begin{bmatrix} \sigma_1^2\Sigma^2 & & & 0 \\ & \sigma_2^2\Sigma^2 & & \\ & & \ddots & \\ 0 & & & \sigma_n^2\Sigma^2 \end{bmatrix}, \tag{4-112}$$

and the $n \times n^2$ matrix $\mathbf{W}$ is defined as follows[6]:

$$\mathbf{W} = [\sigma_1{}^2 \boldsymbol{q}_1{}^* \otimes \boldsymbol{q}_1, \ldots, \sigma_1\sigma_n \boldsymbol{q}_1{}^* \otimes \boldsymbol{q}_n, \ldots, \sigma_1\sigma_n \boldsymbol{q}_n{}^* \otimes \boldsymbol{q}_1, \ldots, \sigma_n{}^2 \boldsymbol{q}_n{}^* \otimes \boldsymbol{q}_n], \tag{4-113}$$

where σ_i is the i-th diagonal element of Σ, $\boldsymbol{q}_i$ is the i-th column of $\mathbf{Q}$, and '$\otimes$' represents the Hadamard product (component-by-component product) [106].

Expanding the second term of (4-105), we find that:

$$E\left[(\boldsymbol{w}^* \mathbf{G} \boldsymbol{w})^2 \right] = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{l=1}^{n} \sum_{p=1}^{n} g_{ij} g_{lp} E[w_i{}^* w_j w_l{}^* w_p]. \tag{4-114}$$

We have assumed that $\boldsymbol{n}$ is a zero-mean white Gaussian noise vector with power spectral density $\sigma^2\mathbf{I}$. Therefore, $\boldsymbol{w}$ is also a zero-mean white Gaussian noise vector with power spectral density $\sigma^2\mathbf{I}$. Using this information, we obtain $E[w_a{}^* w_b w_c{}^* w_d] = \sigma^4\delta_{ab}\delta_{cd} + \sigma^4\delta_{ad}\delta_{bc}$. Hence, (4-114) can be simplified to:

$$E\left[(\boldsymbol{w}^* \mathbf{G} \boldsymbol{w})^2 \right] = \sigma^4 \sum_{i=1}^{n} \sum_{j=1}^{n} g_{ii} g_{jj} + \sigma^4 \sum_{i=1}^{n} \sum_{j=1}^{n} |g_{ij}|^2. \tag{4-115}$$

---

6. Again, '*' represents a complex conjugate.

We express this equation in terms of the column vector $\boldsymbol{g} = U(\mathbf{G})$:

$$E\left[(\boldsymbol{w}^{*}\mathbf{G}\boldsymbol{w})^{2} \right] = \sigma^{4}\boldsymbol{g}^{*}\left[U(\mathbf{I}_{n})U(\mathbf{I}_{n})^{T} + \mathbf{I}_{n^{2}} \right]\boldsymbol{g}. \tag{4-116}$$

The third term of (4-105) can be expanded as follows:

$$E\left[(\boldsymbol{s}^{*}\mathbf{G}\boldsymbol{s})(\boldsymbol{w}^{*}\mathbf{G}\boldsymbol{w}) \right] = \left[\sum_{i=1}^{n}\sum_{j=1}^{n} g_{ij}E[s_{i}^{*}s_{j}] \right]\left[\sum_{l=1}^{n}\sum_{p=1}^{n} g_{lp}E[w_{l}^{*}w_{p}] \right], \tag{4-117}$$

where

$$E[s_{i}^{*}s_{j}] = \sigma_{i}\sigma_{j}\sum_{a=1}^{n}\sum_{b=1}^{n} q_{ai}q_{bj}^{*}E[x_{a}^{*}x_{b}] = m_{2}\sigma_{i}\sigma_{j}\sum_{a=1}^{n} q_{ai}q_{aj}^{*} = m_{2}\sigma_{i}^{2}\delta_{ij}. \tag{4-118}$$

Since $\boldsymbol{w}$ is a zero-mean Gaussian noise vector with power spectral density $\sigma^{2}\mathbf{I}$, $E[w_{i}^{*}w_{j}] = \sigma^{2}$. Hence, (4-117) reduces to:

$$E\left[(\boldsymbol{s}^{*}\mathbf{G}\boldsymbol{s})(\boldsymbol{w}^{*}\mathbf{G}\boldsymbol{w}) \right] = \left[m_{2}\sum_{i=1}^{n}\sigma_{i}g_{ii} \right]\left[\sum_{l=1}^{n}\sigma^{2}g_{ll} \right]. \tag{4-119}$$

We express this equation also in terms of the column vector $\boldsymbol{g} = U(\mathbf{G})$:

$$E\left[(\boldsymbol{s}^{*}\mathbf{G}\boldsymbol{s})(\boldsymbol{w}^{*}\mathbf{G}\boldsymbol{w}) \right] = \boldsymbol{g}^{*}\left[m_{2}\sigma^{2}U(\Sigma^{2})U(\mathbf{I}_{n})^{T} \right]\boldsymbol{g}. \tag{4-120}$$

Expanding the fourth term of (4-105), we obtain:

$$E\left[\boldsymbol{s}^{*}\mathbf{G}\boldsymbol{w}\boldsymbol{w}^{*}\mathbf{G}\boldsymbol{s}\right] = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{l=1}^{n}\sum_{p=1}^{n} g_{ij}g_{lp}{}^{*}E[s_i{}^{*}s_l]E[w_j w_p{}^{*}]. \tag{4-121}$$

From (4-118), we see that $E[s_i{}^{*}s_l] = m_2\sigma_i{}^2\delta_{il}$. Using the fact that $\boldsymbol{w}$ is a zero-mean Gaussian noise vector with power spectral density $\sigma^2\mathbf{I}$, we find that $E[w_j w_p{}^{*}] = \sigma^2$. Therefore, (4-121) reduces to:

$$E\left[\boldsymbol{s}^{*}\mathbf{G}\boldsymbol{w}\boldsymbol{w}^{*}\mathbf{G}\boldsymbol{s}\right] = m_2\sigma^2\sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_i{}^2\,|\,g_{ij}\,|\,^2. \tag{4-122}$$

We express this equation also in terms of the column vector $\boldsymbol{g} = U(\mathbf{G})$:

$$E\left[\boldsymbol{s}^{*}\mathbf{G}\boldsymbol{w}\boldsymbol{w}^{*}\mathbf{G}\boldsymbol{s}\right] = m_2\sigma^2\boldsymbol{g}^{*}\Sigma_B^2\boldsymbol{g}, \tag{4-123}$$

where

$$\Sigma_B^2 = \begin{bmatrix} \Sigma^2 & & 0 \\ & \Sigma^2 & \\ & \ddots & \\ 0 & & \Sigma^2 \end{bmatrix}. \tag{4-124}$$

Finally, the last two terms of (4-105) can be expanded as follows:

$$E\left[\boldsymbol{s}^{*}\mathbf{G}\boldsymbol{s}\right] = \sum_{i=1}^{n}\sum_{j=1}^{n} g_{ij}E[s_i^{*}s_j] = m_2\sum_{i=1}^{n}\sigma_i^2 g_{ij} = m_2\boldsymbol{g}^{*}\boldsymbol{U}(\Sigma^2), \tag{4-125}$$

$$E\left[\boldsymbol{w}^{*}\mathbf{G}\boldsymbol{w}\right] = \sum_{i=1}^{n}\sum_{j=1}^{n} g_{ij}E[w_i^{*}w_j] = \sigma^2\sum_{i=1}^{n} g_{ij} = \sigma^2\boldsymbol{g}^{*}\boldsymbol{U}(\mathbf{I}_n). \tag{4-126}$$

Substituting each of the expanded terms back into the cost function, we arrive at (4-99). ❑

APPENDIX 4.8

PROOF OF THEOREM 4-6

The vector CMA cost function with Gram-Schmidt constraint can be written in terms of the channel input vector $\boldsymbol{x}$ and the overall transfer function matrix $\mathbf{F}$ as follows[7]:

$$J_{GS}(\mathbf{F}) = E\left[\left(\boldsymbol{x}^*\mathbf{F}^*\mathbf{F}\boldsymbol{x}\right)^2 - 2M_v\left(\boldsymbol{x}^*\mathbf{F}^*\mathbf{F}\boldsymbol{x}\right) + M_v{}^2\right] + (m_2)^2\left\|\mathbf{F}\mathbf{F}^* - \mathbf{I}\right\|_F^2. \qquad (4\text{-}127)$$

Since $\boldsymbol{x}$ is random vector sequence, (4-127) is completely parameterized by the overall transfer function matrix $\mathbf{F}$. Both $\mathbf{F}^*\mathbf{F}$ and $\mathbf{F}\mathbf{F}^*$ are positive semi-definite (PSD)[8] and Hermitian[9] matrices; therefore, both matrices have a unique eigendecomposition: $\mathbf{F}^*\mathbf{F} = \mathbf{V}\mathbf{D}\mathbf{V}^*$ and $\mathbf{F}\mathbf{F}^* = \mathbf{U}\mathbf{D}\mathbf{U}^*$, where $\mathbf{V}$ and $\mathbf{U}$ are unitary matrices and $\mathbf{D}$ is a diagonal matrix with non-negative real entries.

In Appendix 4.1, we showed that the first term in (4-127) can be expressed as follows:

$$J_v(\mathbf{V}, \boldsymbol{d}) = \boldsymbol{d}^T\mathbf{R}_{ww}\boldsymbol{d} - 2m_2M_v\boldsymbol{d}^T\mathbf{1}_n + M_v{}^2, \qquad (4\text{-}128)$$

---

7. The dependence on time has been suppressed to simplify the notation.

8. A matrix $\mathbf{G}$ is positive semi-definite if and only if $\boldsymbol{r}^*\mathbf{G}\boldsymbol{r} \geq 0$ for all complex vectors $\boldsymbol{r}$.

9. The matrix $\mathbf{G}$ is Hermitian matrix if and only if $\mathbf{G} = \mathbf{G}^*$.

where $\boldsymbol{d}$ is an $n \times 1$ vector composed of the diagonal elements of $\mathbf{D}$ and where $\mathbf{R}_{ww} = E[\boldsymbol{w}\boldsymbol{w}^T]$. The i-th component of the $n \times 1$ vector $\boldsymbol{w}$ is given by:

$$w_i = |\boldsymbol{v}_i^* \boldsymbol{x}|^2, \tag{4-129}$$

where $\boldsymbol{v}_i$ is the i-th column of $\mathbf{V}$.

We can express the second term in (4-127) in a similar form. The second term can be expanded as follows:

$$(m_2)^2 \left\| \mathbf{F}\mathbf{F}^* - \mathbf{I} \right\|_F^2 = (m_2)^2 \left\| \mathbf{U}\mathbf{D}\mathbf{U}^* - \mathbf{I} \right\|_F^2 = (m_2)^2 tr \left\| \mathbf{D} - \mathbf{I} \right\|_F^2, \tag{4-130}$$

where the second equality is a result of the fact that the Frobenius norm is invariant to a unitary transformation. We can express (4-130) in terms of the vector $\boldsymbol{d}$ as follows:

$$(m_2)^2 \left\| \mathbf{F}\mathbf{F}^* - \mathbf{I} \right\|_F^2 = (m_2)^2 (\boldsymbol{d}^T \boldsymbol{d} - 2\boldsymbol{d}^T \mathbf{1}_n + n). \tag{4-131}$$

Therefore, the vector CMA cost function with GSC can be simplified to:

$$J_{GS}(\mathbf{V}, \boldsymbol{d}) = \boldsymbol{d}^T \left(\mathbf{R}_{ww} + (m_2)^2 \mathbf{I} \right) \boldsymbol{d} - 2\left(m_2 M_v + (m_2)^2 \right) \boldsymbol{d}^T \mathbf{1}_n + \left(M_v^2 + (m_2)^2 n \right). \tag{4-132}$$

We see that (4-132) is completely parameterized by $\mathbf{V}$ and $\boldsymbol{d}$. Since the variables $\mathbf{V}$ and $\boldsymbol{d}$ are independent, the local minima can be determined by first minimizing the cost function with respect to $\boldsymbol{d}$ and then with respect to $\mathbf{V}$. The gradient of J_{GS} with respect to $\boldsymbol{d}$ is given by:

$$\nabla_d J_{GS} = 2\left(\mathbf{R}_{ww} + (m_2)^2\mathbf{I}\right)\boldsymbol{d} - 2\left(m_2 M_v + (m_2)^2\right)\mathbf{1}_n. \tag{4-133}$$

The optimal $\boldsymbol{d}$ occurs when (4-133) is equal to the zero vector, or equivalently when $\boldsymbol{d}$ satisfies the following equation:

$$\left(\mathbf{R}_{ww} + (m_2)^2\mathbf{I}\right)\boldsymbol{d} = \left(m_2 M_v + (m_2)^2\right)\mathbf{1}_n. \tag{4-134}$$

In Appendix 4.1, we showed that $\mathbf{R}_{ww}\mathbf{1}_n = m_2 M_v \mathbf{1}_n$. Using this relationship, it easy to show that $\boldsymbol{d} = \mathbf{1}_n$ satisfies (4-134). Since $J_{GS}(\mathbf{F}) \geq 0$ and (4-132) is quadratic in $\boldsymbol{d}$, $\boldsymbol{d} = \mathbf{1}_n$ is a global minimum for the vector CMA cost function with GSC. If $(\mathbf{R}_{ww} + (m_2)^2\mathbf{I})$ is a full rank matrix, then $\boldsymbol{d} = \mathbf{1}_n$ is also a unique global minimum (independent of $\mathbf{V}$).

We know that if a matrix is positive definite, then it also full rank. In Appendix 4.1, we showed that $\mathbf{R}_{ww}$ is positive definite for non-CM input alphabets ($\kappa > 1$), and in Appendix 4.4, we showed that $\mathbf{R}_{ww}$ is positive semi-definite for CM input alphabets ($\kappa = 1$). In either case, the matrix $(\mathbf{R}_{ww} + (m_2)^2\mathbf{I})$ is positive definite because $(m_2)^2\mathbf{I}$ is positive definite and the sum of a positive definite matrix with either a positive definite or positive semi-definite matrix is always positive definite matrix [106]. Hence, the matrix $(\mathbf{R}_{ww} + (m_2)^2\mathbf{I})$ is full rank, and therefore, $\boldsymbol{d} = \mathbf{1}_n$ is the only solution to (4-134). In fact, this solution is also the unique global minimum for any input alphabet. We observe that when $\boldsymbol{d} = \mathbf{1}_n$, $\mathbf{F}^*\mathbf{F} = \mathbf{I}$ and $\mathbf{F}\mathbf{F}^* = \mathbf{I}$ and therefore, the vector CMA cost function with GSC is minimized for any input alphabet if and only if $\mathbf{F}$ is unitary. ❑

APPENDIX 4.9

DERIVATION OF (4-55)

Recall that the output of the linear detector is given by $\boldsymbol{y}_k = \mathbf{F}\boldsymbol{x}_k + \mathbf{C}\boldsymbol{n}_k$. The $n \times n$ unitary matrix $\mathbf{Q}$ that minimizes the total (and individual) mean-squared error can be found by minimizing:

$$\text{MSE} = \text{E}[\|\mathbf{Q}^*(\mathbf{F}\boldsymbol{x}_k + \mathbf{C}\boldsymbol{n}_k) - \boldsymbol{x}_k\|^2], \tag{4-135}$$

with respect to $\mathbf{Q}$. Expanding (4-135) and taking the expectation, we find it reduces to:

$$\text{MSE} = \text{E}[\|(\mathbf{Q}^*\mathbf{F} - \mathbf{I})\boldsymbol{x}_k + \mathbf{Q}^*\mathbf{C}\boldsymbol{n}_k\|^2], \tag{4-136}$$

$$= m_2\|\mathbf{Q}^*\mathbf{F} - \mathbf{I}\|_F^2 + \|\mathbf{Q}^*\mathbf{C}\|_F^2, \tag{4-137}$$

$$= m_2\|\mathbf{Q}^*\mathbf{F} - \mathbf{I}\|_F^2 + \|\mathbf{C}\|_F^2, \tag{4-138}$$

where the third equality is due to the fact that the Frobenius norm is invariant to a unitary matrix. We observe that last term in (4-138) is independent of the unitary matrix $\mathbf{Q}$. Hence, minimizing the MSE is equivalent to minimizing $\|\mathbf{Q}^*\mathbf{F} - \mathbf{I}\|_F^2$ with respect to $\mathbf{Q}$.

This minimization problem is a classical problem in factor analysis [106]. The unitary matrix that minimizes the MSE is given by $\mathbf{Q} = \mathbf{U}\mathbf{V}^*$, where the $n \times n$ unitary matrices $\mathbf{U}$ and $\mathbf{V}$ are specified by the singular-value decomposition $\mathbf{F} = \mathbf{U}\mathbf{D}\mathbf{V}^*$. ❑

APPENDIX 4.10

DERIVATION OF (4-56)

Recall that the output of the linear detector is given by $\boldsymbol{y}_k = \mathbf{F}\boldsymbol{x}_k + \mathbf{C}\boldsymbol{n}_k$. The $n \times n$ diagonal unitary matrix $\mathbf{Q}_d$ that minimizes the total (and individual) mean-squared error can be found by minimizing:

$$\text{MSE} = \text{E}[\|\mathbf{Q}_d^*(\mathbf{F}\boldsymbol{x}_k + \mathbf{C}\boldsymbol{n}_k) - \mathbf{P}_\text{F}\boldsymbol{x}_k\|^2], \tag{4-139}$$

where $\mathbf{P}_\text{F}$ is a complex permutation matrix that accounts for the inherent ambiguities associated with a blind detection problem, with respect to $\mathbf{Q}_d$. Expanding (4-139) and taking the expectation, we find it reduces to:

$$\text{MSE} = \text{E}[\|(\mathbf{Q}_d^*\mathbf{F} - \mathbf{P}_\text{F})\boldsymbol{x}_k + \mathbf{Q}_d^*\mathbf{C}\boldsymbol{n}_k\|^2], \tag{4-140}$$

$$= m_2\|\mathbf{Q}_d^*\mathbf{F} - \mathbf{P}_\text{F}\|_F^2 + \|\mathbf{Q}_d^*\mathbf{C}\|_F^2, \tag{4-141}$$

$$= m_2\|\mathbf{Q}_d^*\mathbf{F} - \mathbf{P}_\text{F}\|_F^2 + \|\mathbf{C}\|_F^2, \tag{4-142}$$

where the third equality is due to the fact that the Frobenius norm is invariant to a unitary matrix. We observe that last term in (4-142) is independent of the unitary matrix $\mathbf{Q}_d$. Hence, minimizing the MSE is equivalent to minimizing $\|\mathbf{Q}_d{}^*\mathbf{F} - \mathbf{P}_\mathrm{F}\|_F^2$ with respect to $\mathbf{Q}_d$.

Expanding the first term in (4-142), we find that it reduces to:

$$\mathrm{MSE} = \|\mathbf{Q}_d{}^*\mathbf{F} - \mathbf{P}_\mathrm{F}\|_F^2, \tag{4-143}$$

$$= \|\tilde{\mathbf{F}} - \mathbf{Q}_d\|_F^2, \tag{4-144}$$

$$= \sum_{i=1}^{n} |f_{ii}|^2 + 1 - 2|f_{ii}|\cos(\phi_{ii} - \theta_i) + \sum_{i \neq j} |f_{ij}|^2, \tag{4-145}$$

where we have defined $\tilde{\mathbf{F}} = \mathbf{F}\mathbf{P}_\mathrm{F}{}^*$, $[\tilde{\mathbf{F}}]_{ij} = |f_{ij}|exp(j\phi_{ij})$, and $[\mathbf{Q}_d]_{ij} = exp(j\theta_i)$. Clearly, (4-145) is minimized if and only if $\phi_{ii} = \theta_i$. Hence, the diagonal unitary matrix $\mathbf{Q}_d$ that minimizes the total MSE is given by:

$$[\mathbf{Q}_d]_{ii} = exp(j\angle[\mathbf{F}\mathbf{P}_\mathrm{F}{}^*]_{ii}), \tag{4-146}$$

where $\mathbf{P}_\mathrm{F}$ removes both the ambiguity in assigning labels to each user and the ambiguity inherent in all QAM constellation from the overall transfer function matrix $\mathbf{F}$. ❑

CHAPTER 5

SPATIO-TEMPORAL VECTOR CMA

The original constant-modulus algorithm (CMA) cost function was designed to blindly mitigate intersymbol interference (ISI) for channels with memory [85]. Godard showed that, in the absence of noise, the global minimum of the cost function corresponds to the case of zero ISI [85]. Later, Foschini demonstrated that this minimum is the only minima of the cost function when the equalizer has infinite length [86]. These results are valid for both minimum-phase and non-minimum phase channels. The CMA cost function can thus mitigate ISI for all types of channels. This result is a consequence of the fact that the cost function uses both second-order statistics and higher-order statistics when inverting a channel.

In Chapter 2, we showed that almost all square channels with memory are non-minimum phase and that almost all tall channels with memory are minimum-phase. One of the reasons that we proposed a generalization of the CMA to vector-valued signals is because of its ability to mitigate interference on both minimum-phase and non-minimum phase channels. In this chapter, the vector CMA-based algorithms and results from the previous chapter are extended to channels with memory. We show that the resulting cost function

can mitigate intersymbol interference and multiuser interference on both square and tall channels with memory. We also show that the resulting algorithms fit in with the general whiten-rotate structure described in Chapter 1.

In Section 5.1, we introduce the channel model and assumptions that will be used throughout the remainder of this chapter. In Section 5.2, we show that tall channels with memory have some astonishing properties: they are almost always minimum-phase and more importantly, they can always be inverted by an FIR linear detector. In Section 5.3, we introduce the vector constant-modulus algorithm cost function and we derive its corresponding stochastic gradient-descent algorithm. In Section 5.4, we determine the local minima of the cost function in the absence of noise. We show that for certain input alphabets, the cost function is minimized only by desirable local minima, while for other input alphabets, it is minimized by both desirable and undesirable local minima. In Section 5.5, we propose a modification to the vector CMA cost function, which can eliminate the undesirable local minima for all input alphabets. We also determine the local minima of the modified vector CMA cost function in the absence of noise, and derive the corresponding stochastic gradient-descent algorithm. In Section 5.6, we quantify the performance of this algorithm in the presence of noise. Finally, in Section 5.7, we present several simulation results which demonstrate the effectiveness, in terms of speed of convergence and complexity, of the proposed algorithm.

5.1 Channel Model and Assumptions

Consider the multiuser channel model depicted in Fig. 5-1:

$$\boldsymbol{r}_k = \sum_{i=0}^{M} \mathbf{H}_i \boldsymbol{x}_{k-i} + \boldsymbol{n}_k, \tag{5-1}$$

where $\mathbf{H}(z) = \mathbf{H}_0 + \mathbf{H}_1 z^{-1} + \ldots + \mathbf{H}_M z^{-M}$ is an $m \times n$ causal and finite-impulse response (FIR) channel transfer function matrix with memory M. This type of channel arises in a wide variety of real-world applications, such as an m-sensor, n-user uniform linear array with multipath and an n-user asynchronous CDMA system. The transmitted vector $\boldsymbol{x}_k$ is an $n \times 1$ vector consisting of the symbols sent by the n independent users. The received vector $\boldsymbol{r}_k$ is an $m \times 1$ vector composed of the receiver observations, while $\boldsymbol{n}_k$ represents an $m \times 1$ noise vector with power spectral density $E[\boldsymbol{n}_k \boldsymbol{n}_k^*] = \sigma^2 \mathbf{I}$, with $\sigma^2 > 0$.

We make the following assumptions: first, the channel $\mathbf{H}(z)$ has full-column rank on the unit circle ($|z| = 1$), which implies that the channel input vector $\boldsymbol{x}_k$ can be recovered using a linear detector (see Section 2.1.2). Second, the signal and noise components are independent and have zero mean. Third, the components of the channel input vector $\boldsymbol{x}_k$ are stationary, independent, and uniformly selected from a finite input-alphabet.

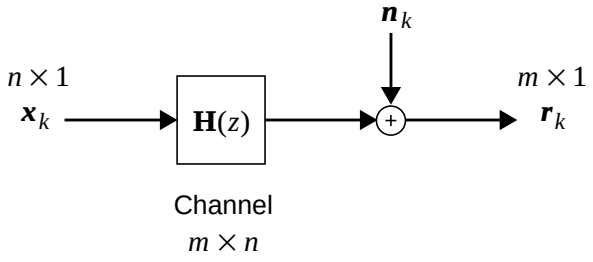


Fig. 5-1. A block diagram of an $m \times n$ noisy channel with memory.

As in Section 4.1, we can use the kurtosis to subdivide the set of all input alphabets into three distinct and disjoint subsets. The following three definitions summarize the different subsets.

Definition 5-1. If $\kappa < 2$, then the input alphabet is said to be *sub-Gaussian*.

Almost all well-known input alphabets, such phase-shift keying (PSK) and quadrature amplitude modulation (QAM) constellations, are sub-Gaussian (see Section 4.1). The reason these input alphabets are called sub-Gaussian is because their kurtosis is less than the kurtosis for a complex Gaussian input alphabet.

Definition 5-2. An input alphabet is said to be *meso-Gaussian* if $\kappa = 2$.

Clearly, a complex Gaussian input alphabet is meso-Gaussian. However, not all meso-Gaussian input alphabets have a complex Gaussian distribution. For example, the following input alphabet does not have a complex Gaussian distribution:

$$x_k^{(i)} = \begin{cases} \{\pm 1 \pm j\} & \text{each with probability } p(x_k) = \dfrac{13 - \sqrt{7}}{6} \\ \{\pm 3 \pm 3j\} & \text{each with probability } p(x_k) = \dfrac{3 + \sqrt{7}}{64} \end{cases}, \tag{5-2}$$

but it is a meso-Gaussian input alphabet. Finally, the last subset is defined as follows:

Definition 5-3. If $\kappa > 2$, then the input alphabet is said to be *super-Gaussian*.

A super-Gaussian input alphabet can be created by heavily shaping the inner symbols of an input alphabet. For example, the following input alphabet:

$$x_k^{(i)} = \begin{cases} \{\pm 1 \pm j\} & \text{each with probability } p(x_k) = 0.20 \\ \\ \{\pm 3 \pm 3j\} & \text{each with probability } p(x_k) = 0.05 \end{cases}, \tag{5-3}$$

has a kurtosis of $\kappa = 2.5148$.

We focus our attention primarily on the sub-Gaussian input alphabets, since they are used in most real-world applications.

5.2 TALL CHANNELS

In this chapter, we consider both square ($m = n$) and tall ($m > n$) channels with memory in (5-1). Tall channels have some surprising properties: they are almost always minimum-phase and they can be inverted by an finite-impulse response (FIR) linear detector. However, these results do not apply to square channels.

In the following discussion, we show that a tall channel with memory M is inverted by an FIR linear detector. A zero-forcing linear detector can be defined as follows:

> **Definition 5-4.** A zero-forcing linear detector is any $n \times m$ stable LTI filter $\mathbf{C}(z)$ satisfying $\mathbf{C}(z)\mathbf{H}(z) = \mathbf{D}(z)$, where $\mathbf{D}(z)$ is an $n \times n$ diagonal matrix whose i-th diagonal element has the form z^{-D_i} and D_i is an arbitrary integer.

Consider an $m \times n$ tall channel with memory M: $\mathbf{H}(z) = \mathbf{H}_0 + \mathbf{H}_1 z^{-1} + \ldots + \mathbf{H}_M z^{-M}$, and an $n \times m$ linear detector with memory N: $\mathbf{C}(z) = \mathbf{C}_0 + \mathbf{C}_1 z^{-1} + \ldots + \mathbf{C}_N z^{-N}$. We can

express the relationship $\mathbf{C}(z)\mathbf{H}(z) = \mathbf{D}(z) = \mathbf{D}_0 + \mathbf{D}_1 z^{-1} + \ldots + \mathbf{D}_{M+N} z^{-(M+N)}$ in block-matrix notation as:

$$[\mathbf{C}_0\ \mathbf{C}_1\ \ldots\ \mathbf{C}_N]\mathit{H} = [\mathbf{D}_0\ \mathbf{D}_1\ \ldots\ \mathbf{D}_{M+N}], \tag{5-4}$$

where H is the $(N+1)m \times (M+N+1)n$ block-Toeplitz matrix:

$$\mathit{H} = \begin{bmatrix} \mathbf{H}_0 & \mathbf{H}_1 & \cdots & \mathbf{H}_M & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_0 & \mathbf{H}_1 & \cdots & \mathbf{H}_M & \mathbf{0} & \cdots \\ & & & \ddots & & \ddots & \\ \mathbf{0} & & \cdots & \mathbf{0} & \mathbf{H}_0 & \mathbf{H}_1 & \cdots \mathbf{H}_M \end{bmatrix}. \tag{5-5}$$

A solution to (5-4) exists if there are at least as many unknowns as the number of equations. We observe that the linear detector coefficients $\{\mathbf{C}_0, \mathbf{C}_1, \ldots, \mathbf{C}_N\}$ represent $(N+1)mn$ unknowns, while each block column of (5-4) defines n^2 equations; yielding a total of $(M+N+1)n^2$ equations. Hence, a solution (5-4) exists when $(N+1)mn \geq (M+N+1)n^2$, or equivalently when [39,53,54,113],

$$N \geq \left\lceil \frac{Mn}{m-n} \right\rceil - 1. \tag{5-6}$$

In other words, a solution to (5-4) exists if the total number of taps in the linear detector equals $\left\lceil \frac{Mn}{m-n} \right\rceil$.

Example 5-1. Consider an 8×2 channel with memory $M = 2$. There exists a zero-forcing linear detector $\mathbf{C}(z)$ with memory $N = \left\lceil \frac{Mn}{m-n} \right\rceil = \left\lceil \frac{4}{8-2} \right\rceil = 1$ that inverts the channel.

If $\left\lceil \frac{Mn}{m-n} \right\rceil$ is an integer, then H is a square matrix and so the zero-forcing linear detector satisfying (5-4) is unique and is given by

$$[\mathbf{C}_0\ \mathbf{C}_1\ \ldots\ \mathbf{C}_N] = [\mathbf{D}_0\ \mathbf{D}_1\ \ldots\ \mathbf{D}_{M+N}]\mathsf{H}^{-1}. \tag{5-7}$$

On the other hand, if $\left\lceil \frac{Mn}{m-n} \right\rceil$ is not an integer, then H is a tall matrix (it has more rows than columns). For this case, the zero-forcing linear detector coefficients are not unique because any vector in the nullspace of H^* can be added to any row of (5-4) to yield another linear detector.

A surprising by-product of the above discussion is that both the intersymbol interference and the multiuser interference can often be mitigated with a *memoryless* detector, which is a linear detector consisting of a single tap. A memoryless detector is sufficient when $\left\lceil \frac{Mn}{m-n} \right\rceil = 1$, or equivalently when

$$m \geq (N+1)n. \tag{5-8}$$

This result is due to Falconer *et al*. [113].

If we solve for n instead of m in (5-8), we find that the number of users that can be linearly separated by a memoryless detector is given by [39,53,54,113]:

$$n \leq \frac{m}{M+1}. \tag{5-9}$$

The number of users that can be separated is therefore a fraction of the number of sensors. If we allow memory in the linear detector and fix the value at N, then the number of users that can be linearly separated is given by [53,113]:

$$n \leq \frac{(N+1)m}{M+N+1}. \tag{5-10}$$

As the memory N of the linear detector increases and approaches infinity, the number of users n that can be linearly separated approaches the number of sensors m, which is the dimension of the channel output.

5.3 VECTOR CMA

Since we have assumed that the channel $\mathbf{H}(z)$ has full-column rank on the unit circle, the transmitted vector $\boldsymbol{x}_k$ can be recovered by passing the $m \times 1$ received vector $\boldsymbol{r}_k$ through an $n \times m$ adaptive linear detector $\mathbf{C}(z)$, as illustrated in Fig. 5-2. We assume that the linear detector is causal and FIR with memory N: $\mathbf{C}(z) = \mathbf{C}_0 + \mathbf{C}_1 z^{-1} + \ldots + \mathbf{C}_N z^{-N}$. This detector, often referred to as a short linear detector, has fewer outputs than inputs. We can express the detector output vector as follows:

$$\boldsymbol{y}_k = \sum_{i=0}^{N} \mathbf{C}_i \boldsymbol{r}_{k-i}. \tag{5-11}$$

It should be emphasized that the dimension of the detector output is the same as the dimension for the channel input.

Using block-matrix notation, (5-11) reduces to:

$$\boldsymbol{y}_k = \mathbf{C}\mathbf{R}_k, \tag{5-12}$$

where $\mathbf{C} = [\mathbf{C}_0\ \mathbf{C}_1\ \ldots\ \mathbf{C}_N]$ is $n \times (N+1)m$ matrix composed of the multiuser detector coefficients and $\mathbf{R}_k{}^T = [\boldsymbol{r}_k{}^T\ \boldsymbol{r}_{k-1}{}^T\ \ldots\ \boldsymbol{r}_{k-N}{}^T]$ is an $(N+1)m \times 1$ stacked-observation vector. Finally, we can relate the detector output vector $\boldsymbol{y}_k$ to the channel input vector $\boldsymbol{x}_k$ again using the block-matrix notation:

$$\boldsymbol{y}_k = \mathbf{C}\mathit{H}\mathbf{X}_k + \mathbf{C}\mathbf{N}_k, \tag{5-13}$$

where $\mathbf{X}_k{}^T = [\boldsymbol{x}_k{}^T\ \boldsymbol{x}_{k-1}{}^T\ \ldots\ \boldsymbol{x}_{k-(M+N)}{}^T]$ is an $(M+N+1)n \times 1$ stacked-input vector, $\mathbf{N}_k{}^T = [\boldsymbol{n}_k{}^T\ \boldsymbol{n}_{k-1}{}^T\ \ldots\ \boldsymbol{n}_{k-N}{}^T]$ is an $(N+1)m \times 1$ stacked-noise vector, and H is the $(N+1)m \times (M+N+1)n$ block-Toeplitz matrix of (5-5). The $n \times n$ overall transfer function matrix $\mathbf{F}(z)$

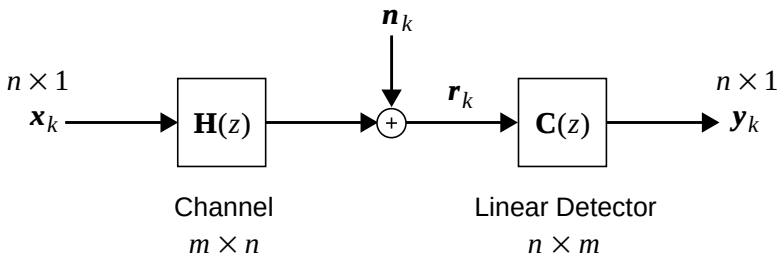


Fig. 5-2. A block diagram of an $m \times n$ channel with followed by an $n \times m$ linear detector with memory.

= $\mathbf{C}(z)\mathbf{H}(z)$ (or equivalently $\mathbf{F} = \mathbf{C}H$) represents the cascade of the channel matrix and the linear detector.

As in Chapter 4, we propose to adapt the linear detector $\mathbf{C}(z)$ using a multidimensional generalization of the constant modulus-algorithm. The reason for choosing CMA is that it is adaptive and has low complexity. Another important advantage of CMA is that it can mitigate intersymbol interference on both minimum and non-minimum phase channels by implicitly using higher-order statistics.

5.3.1 Cost Function

The cost function for vector CMA is defined in (4-8). For convenience, the equation is reproduced below

$$J_v = E\left[\left(\|\boldsymbol{y}_k\|^2 - M_v\right)^2\right], \tag{5-14}$$

where $\boldsymbol{y}_k = \mathbf{C}\mathbf{R}_k$. Following the approach presented in Section 4.2, we can determine the value for the constant M_v. The noiseless vector CMA cost function can be expressed in terms of the stacked-received vector $\mathbf{R}_k$ and the linear detector $\mathbf{C}$ as:

$$J_v = E\left[\, tr[(\mathbf{C}\mathbf{R}\mathbf{R}^*\mathbf{C}^*)^2] - 2M_v tr(\mathbf{C}\mathbf{R}\mathbf{R}^*\mathbf{C}^*) + {M_v}^2\right]. \tag{5-15}$$

We observe that (5-15) is a generalization of (4-9) to channels with memory. The dependence on time has been suppressed in order to simplify the notation. The complex gradient of (5-15) with respect to $\mathbf{C}$ is given by:

$$\nabla_{\mathbf{C}}\, J_v = 4E(\|\boldsymbol{y}\|^2 \boldsymbol{y}\mathbf{X}^*)\boldsymbol{H}^* - 4M_v E(\boldsymbol{y}\mathbf{X}^*)\boldsymbol{H}^*. \tag{5-16}$$

Substituting $\boldsymbol{y} = \boldsymbol{x}$ into (5-16) and setting the gradient equal to the zero matrix, we find that M_v must satisfy:

$$\left[E(\|\boldsymbol{x}\|^2 \boldsymbol{x}\mathbf{X}^*) - M_v E(\boldsymbol{x}\mathbf{X}^*) \right]\boldsymbol{H}^* = \mathbf{0}. \tag{5-17}$$

Since the block-Toeplitz channel matrix $\boldsymbol{H}$ has full rank, the only solution to (5-17) occurs when:

$$E(\|\boldsymbol{x}\|^2 \boldsymbol{x}\mathbf{X}^*) = M_v E(\boldsymbol{x}\mathbf{X}^*). \tag{5-18}$$

Assuming that all users are independent in both space and time, (5-18) reduces to:

$$E(\|\boldsymbol{x}\|^2 \boldsymbol{x}\boldsymbol{x}^*) = M_v E(\boldsymbol{x}\boldsymbol{x}^*), \tag{5-19}$$

which is identical to (4-12).

We now restate the theorem and corollary given in Chapter 4 that describes the conditions for which an M_v can exist and its corresponding value.

Theorem 5-1. There exists an M_v which satisfies (5-19) if and only if

$$m_2^{(i)}(\kappa_i - 1) = K, \quad \forall\ i \tag{5-20}$$

where K is some positive constant.

Corollary 5-1. If (5-20) holds, then M_v is given by

$$M_v = \frac{E[\|\boldsymbol{x}\|^4]}{E[\|\boldsymbol{x}\|^2]}. \tag{5-21}$$

As in Section 4.2, we will assume that all users are independent and identically distributed, *i.e.*, $m_2 = m_2^{(i)}$, $m_4 = m_4^{(i)}$, and $\kappa = \kappa_i \ \forall \ i$, for the remainder of this chapter. Note that M_v is related to the kurtosis κ by the following relationship:

$$M_v = m_2 \, (n + \kappa - 1). \tag{5-22}$$

This assumption will simplify the analysis of the cost function later in this chapter.

5.3.2 Stochastic Algorithm

The main focus of this research is to design an adaptive linear detector. As in Section 4.6.1, we adjust the detector tap weights according to the classical steepest-descent algorithm:

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \frac{\mu}{4} \nabla_{\mathbf{C}} J_v, \tag{5-23}$$

where $\nabla_{\mathbf{C}}$ is the complex gradient of the cost function with respect to the detector tap weights and where μ is the step size. The complex gradient of (5-14) with respect to $\mathbf{C}$ is given by:

$$\nabla_{\mathbf{C}} J_v = 4E[\boldsymbol{e}_k \mathbf{R}_k^*], \tag{5-24}$$

where the "error signal" $\boldsymbol{e}_k$ is the same as that of Section 4.6.1, namely,

$$\boldsymbol{e}_k = \boldsymbol{y}_k(\|\boldsymbol{y}_k\|^2 - M_v). \tag{5-25}$$

Substituting $4\boldsymbol{e}_k \mathbf{R}_k^*$ as a stochastic approximation of the gradient in the steepest-descent algorithm, we arrive at the following update equation for the linear detector:

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \mu \boldsymbol{e}_k \mathbf{R}_k^*. \tag{5-26}$$

We refer to this algorithm as the vector CMA. Observe that this algorithm reduces to that of Section 4.6.1 for memoryless channels and it reduces to the familiar single-user CMA when $n = 1$.

In summary, the vector CMA, for channels with memory, is defined by (5-25), (5-26), and $M_v = E[\|\boldsymbol{x}_k\|^4] / \mathrm{E}[\|\boldsymbol{x}_k\|^2]$.

5.4 Local Minima in the Absence of Noise

In this section, we determine the local minima for the vector CMA cost function in the absence of noise for a channel with memory. As shown in Section 4.3, this cost function is minimized by any unitary matrix, $\mathbf{F}(z) = \mathbf{U}$. Hence, there exists an infinite number of local minima. One question remains: are there any other local minima of the vector CMA cost function? The answer to this question depends on the subset in which the input alphabet

lies. The following four theorems summarize the local minima of the vector CMA cost function for all input alphabets.

Theorem 5-2. If the input alphabet is sub-Gaussian and non-CM ($1 < \kappa < 2$) and the linear detector has infinite length, then the vector CMA cost function is minimized if and only if

$$\mathbf{F}(z) = \mathbf{U}\mathbf{P}(z), \tag{5-27}$$

where $\mathbf{U}$ is an arbitrary unitary matrix and $\mathbf{P}(z)$ is an $n \times n$ matrix which has only one nonzero entry of the form z^{-D} where D is an arbitrary delay value, per row. If there is more than one nonzero entry in any column, then the delay values in that column must all be different.

Proof: See Appendix 5.1.

Theorem 5-3. If the input alphabet is CM ($\kappa = 1$) and the linear detector has infinite length, then the vector CMA cost function is minimized if and only if

$$\mathbf{F}(z) = \mathbf{U}\mathbf{D}^{1/2}\mathbf{P}(z), \tag{5-28}$$

where $\mathbf{U}$ is an arbitrary unitary matrix, $\mathbf{D}$ is a non-negative real diagonal matrix satisfying $tr(\mathbf{D}) = n$, and $\mathbf{P}(z)$ is an $n \times n$ matrix which has only one nonzero entry of the form z^{-D} where D is an arbitrary delay value, per row. If there is more than one nonzero entry in any column, then the delay values in that column must all be different.

Proof: See Appendix 5.3.

Some examples of valid $\mathbf{P}(z)$ are given below:

$$\mathbf{P}(z) = \begin{bmatrix} z^{-D_1} & 0 \\ 0 & z^{-D_2} \end{bmatrix} \mathbf{P}_1, \begin{bmatrix} z^{-D_1} & 0 \\ z^{-D_2} & 0 \end{bmatrix} \mathbf{P}_2, \begin{bmatrix} z^{-D_1} & 0 & 0 \\ 0 & z^{-D_2} & 0 \\ 0 & z^{-D_3} & 0 \end{bmatrix} \mathbf{P}_3, \tag{5-29}$$

where D_i is any integer such that $D_1 \neq D_2 \neq D_3$, and $\mathbf{P}_i$ any real permutation matrix. An example of a matrix that does not satisfy the conditions for $\mathbf{P}(z)$ stated in either Theorem 5-2 or Theorem 5-3 is given by $\begin{bmatrix} z^{-1} & 0 \\ z^{-1} & 0 \end{bmatrix}$.

From Theorem 5-2 and Theorem 5-3, we observe that the vector CMA cost function for a sub-Gaussian input alphabet has both desirable and undesirable local minima.

Property 5-1. For a sub-Gaussian input alphabet, the vector CMA cost function is minimized by both desirable and undesirable local minima.

The desirable minima occur when the detector output vector is related to the channel input vector by an unknown unitary ambiguity and by an unknown delay for each users.

Definition 5-5. A *desirable local* minima of the vector CMA cost function has the form: $\mathbf{F}(z) = \mathbf{U}\mathbf{D}(z)$, where $\mathbf{U}$ is an arbitrary unitary matrix and $\mathbf{D}(z)$ is a diagonal matrix whose diagonal entries are of the form z^{-D}, where D is an arbitrary delay value.

If $\mathbf{F}(z) = \mathbf{U}\mathbf{D}(z)$, then detector output vector takes on the form:

$$\boldsymbol{y}_k = \mathbf{U}[\, x^{(1)}_{k-D_1}, x^{(2)}_{k-D_2}, \ldots, x^{(n)}_{k-D_n}]^T, \tag{5-30}$$

where D_i is an arbitrary integer value. In the general, the delay values $\{D_i\}$ are unknown. Unfortunately, delay in the detector output vector is an inherent problem of blind detection, but fortunately, the delay does not interfere with the reconstruction of the original input sequence $\{\boldsymbol{x}_k\}$.

When the vector CMA cost function is minimized by a desirable local minimum, the autocorrelation of the detector output vector $\boldsymbol{y}_k$ is given by:

$$E[\boldsymbol{y}_k \boldsymbol{y}_{k-l}^*] = \delta_l \mathbf{I}, \quad \forall\, l, \tag{5-31}$$

where δ_l is the Kronecker delta function. The corresponding linear detector that results in this desirable local minimum is referred to as a *whitener.*

> **Definition 5-6.** An $n \times m$ linear detector $\mathbf{C}(z)$ is said to be a whitener if the autocorrelation of $\boldsymbol{y}_k$ is given by: $E[\boldsymbol{y}_k \boldsymbol{y}_{k-l}^*] = \delta_l \mathbf{I}$, $\forall\, l$.

For a desirable local minimum, the transmitted data can be recovered by appending a blind unitary estimator to the output of the linear detector. Possible solutions to this problem include the MPLL (see Chapter 3), JADE, and EASI algorithms.

From Theorem 5-2 and Theorem 5-3, we also see that the vector CMA cost function for a sub-Gaussian input alphabet has many undesirable local minima. For example, the following matrix

$$\mathbf{F}(z) = \begin{bmatrix} z^{-2} & 0 & 0 \\ z^{-5} & 0 & 0 \\ 0 & 0 & z^{2} \end{bmatrix}, \tag{5-32}$$

is an undesirable local minimum for both non-CM and CM input alphabets, because the information from the second user is completely lost; the first two detector outputs lock on to the first user, while the third detector output locks on to the third user. Another example of an undesirable local minimum is given by the matrix

$$\mathbf{F}(z) = \begin{bmatrix} \sqrt{\frac{3}{2}} z^{-1} & 0 \\ 0 & \sqrt{\frac{1}{2}} z^{3} \end{bmatrix}. \tag{5-33}$$

We see that this $\mathbf{F}(z)$ only minimizes the vector CMA cost function when the input alphabet is CM. Even though it recovers information from both users, this local minimum is undesirable because the gain for each user is incorrect, which in turn may lead to an incorrect decision at the decision device.

> **Definition 5-7.** A matrix, where information from one or more of the users is lost or where the gain for each user is incorrect, is an *undesirable minimum* of the vector CMA cost function.

In Section 4.3, we showed that the local minima for a memoryless channel and a non-CM input alphabet are always desirable. However, when we extend this result to channels with memory, we observe that the cost function is minimized by both desirable and undesirable local minima. The reason that the vector CMA cost function is not just minimized by desirable minima is because the cost function is memoryless; (5-14) is only a function of the detector output at time k. The cost function can prevent the detector outputs from being correlated if all the detector outputs have the same delay, but if the delays are different, then it is possible for the detector to recover the same user more than once as long as the delays are different, because to the detector, the same user with different delays appears to be uncorrelated users. For example, this latter case is described by (5-32), where the detector recovers the first user twice, but with different delays. This result suggests that if we add memory to the cost function, then it might be possible to eliminate the undesirable local minima. We will explore this possibility in the next section.

There occur even more undesirable local minima for a CM input alphabet because of the implicit reliance of the vector CMA cost function on both second-order and fourth-order statistics to invert a channel. Unfortunately, a CM input alphabet is completely described by its second-order statistics, and therefore, the cost function does not have enough information to correctly invert the channel.

Before we can state the local minima results for a meso-Gaussian input alphabet, we need the following definition:

Definition 5-8. A multiuser filter $\mathbf{F}(z) = \mathbf{F}_0 + \mathbf{F}_1 z^{-1} + \ldots + \mathbf{F}_K z^{-K}$ is an energy-preserving filter if and only if

$$\sum_{k=0}^{K} \sum_{j=1}^{n} |[\mathbf{F}_k]_{ij}|^2 = 1, \tag{5-34}$$

where $[\mathbf{F}_k]_{ij}$ is the (i, j)-th component of the k-th tap of $\mathbf{F}(z)$.

Theorem 5-4. If the input alphabet is meso-Gaussian ($\kappa = 2$) and the linear detector has infinite length, then the vector CMA cost function is minimized if and only if $\mathbf{F}(z)$ is an energy-preserving filter.

Proof: See Appendix 5.5.

If the transfer function matrix $\mathbf{F}(z)$ is viewed in terms of a block-matrix notation $\mathbf{F}$, then the constraint specified by (5-34) is equivalent to the condition that each row of $\mathbf{F}$ has unit length; in other words, the rows of $\mathbf{F}$ must lie on an $(M+N+1)n$-dimensional unit hypersphere. It is observed that *almost all* of the points on the unit hypersphere are undesirable local minima. Assuming that all points on the unit hypershpere are equally likely, the probability that the linear detector will converge to a desirable local minima is infinitesimal. Hence, the vector CMA cost function is useless for a meso-Gaussian input alphabet.

Property 5-2. The vector CMA cost function is useless for a meso-Gaussian input alphabet.

As with a sub-Gaussian input alphabet, the local minima of the cost function are desirable for a memoryless channel, but are both desirable and undesirable for a channel with memory.

We have seen that the vector CMA cost function relies on both second-order and fourth-order statistics to invert a channel. Since second-order statistics completely describe a meso-Gaussian input alphabet, the vector CMA cost function can only restore the amplitude of the input signal that is completely described by second-order statistics. This results can also be seen by examining (5-34) more closely. If we assume that all of the users are independent and identically distributed and there is no source correlation, then (5-34) ensures that the amplitude of the detector output is the same as the amplitude of the channel input. The phase information of the input signal can never be restored unless more information is known.

Theorem 5-5. If the input alphabet is super-Gaussian ($\kappa > 2$) and the linear detector has infinite length, then the vector CMA cost function is minimized if and only if $\mathbf{F}(z)$ has maximal multiuser and intersymbol interference.

Proof: See Appendix 5.6.

From this theorem, we observe that the local minima of the vector CMA cost function for a super-Gaussian input alphabet results in maximal multiuser and intersymbol interference. Therefore, the vector CMA cost function is useless for this input alphabet.

Property 5-3. The vector CMA cost function is useless for a super-Gaussian input alphabet.

Unlike with a sub-Gaussian and a meso-Gaussian input alphabet, the vector CMA cost function for a super-Gaussian input is minimized only by undesirable local minima for a

channel with memory. In contrast, we have shown in Section 4.3, that for a memoryless channel, the vector CMA cost function for a super-Gaussian input alphabet is minimized only by desirable local minima.

We should emphasize that the results described by Theorem 5-2 through Theorem 5-5 reduce to the familiar results for a single-user CAM cost function when $n = 1$. Also, the results obtained in this chapter are consistent with those of Chapter 4.

5.5 VECTOR CMA WITH GRAM-SCHMIDT CONSTRAINT

In the previous section, we showed that the vector CMA cost function only has desirable local minima for a sub-Gaussian input alphabet. Unfortunately, this cost function is also minimized by undesirable local minima for this input alphabet. We also showed that this cost function is useless for meso-Gaussian and super-Gaussian input alphabets. Therefore, in the remainder of this chapter, we will focus exclusively on sub-Gaussian input alphabets.

5.5.1 Cost Function

The desirable local minima of the vector CMA cost function have the form: $\mathbf{F}(z) = \mathbf{U}\mathbf{D}(z)$, where $\mathbf{U}$ is an arbitrary unitary matrix and $\mathbf{D}(z) = diag(z^{-D_1}, \ldots, z^{-D_n})$. The corresponding detector output vector $\boldsymbol{y}_k$ from (5-30) is given by:

$$\boldsymbol{y}_k = \mathbf{U}[x_{k-D_1}^{(1)}, x_{k-D_2}^{(2)}, \ldots, x_{k-D_n}^{(n)}]^T, \tag{5-35}$$

where D_i is an arbitrary integer value. We see that the autocorrelation of (5-35) is given by:

$$E[\mathbf{y}_k \mathbf{y}_{k-l}^*] = m_2 \delta_l \mathbf{I}, \quad \forall\ l. \tag{5-36}$$

We should point out that the undesirable local minima do not satisfy (5-36). We can therefore eliminate them by modifying the vector CMA cost function to penalize the solutions for which $E[\mathbf{y}_k \mathbf{y}_{k-l}^*] \neq m_2 \delta_l \mathbf{I}$, $\forall\ l$. Since both the channel and the linear detector have finite memory, we only need to consider the autocorrelation for the range $-(M+N)$ to $(M+N)$, because $E[\mathbf{y}_k \mathbf{y}_{k-l}^*] = 0$ for $|l| > (M+N)$. The modified cost function is given by:

$$J_{GS} = J_v + \sum_{l=-(M+N)}^{M+N} \left\| E[\mathbf{y}_k \mathbf{y}_{k-l}^*] - m_2 \mathbf{I} \delta_l \right\|_F^2, \tag{5-37}$$

where $\| \cdot \|_F$ is the Frobenius norm for matrices, and J_v is the vector CMA cost function described by (5-14). We refer to (5-37) as the *vector CMA cost function with Gram-Schmidt constraint* (GSC). Observe that (5-37) is a generalization of (4-38) to channels with memory. We should point out that this cost function penalizes cross-correlation in both time and space.

5.5.2 Local Minima in the Absence of Noise

As in Section 4.5, the second term in the vector CMA cost function is minimized only by the desirable local minima. The value of the cost function is larger when the transfer function matrix is equal to an undesirable local minima. Thus, the additional term in the cost function eliminates the undesirable local minima. The question remains: does the additional term introduce any new local minima? Fortunately, the answer is no. The local

minima for the cost function described by (5-37) are summarized by Theorem Proof: given below:

Theorem 5-6. For all sub-Gaussian input alphabets ($1 \leq \kappa < 2$), J_{GS} is minimized if and only if $\mathbf{F}(z) = \mathbf{UD}(z)$, where $\mathbf{U}$ is an arbitrary unitary matrix and $\mathbf{D}(z)$ is a diagonal matrix whose diagonal entries are of the form z^{-D}, where D is an arbitrary delay value.

Proof: See Appendix 5.7.

When $\mathbf{F}(z) = \mathbf{UD}(z)$, the detector output vector is related to the channel input vector by an unknown unitary ambiguity and an unknown delay for each user; in other words,

$$\boldsymbol{y}_k = \mathbf{U}[x_{k-D_1}^{(1)}, x_{k-D_2}^{(2)}, \ldots, x_{k-D_n}^{(n)}]^T, \tag{5-38}$$

where D_i is an arbitrary integer value. The corresponding linear detector is given by $\mathbf{C}(z) = \mathbf{UD}(z)\mathbf{H}(z)^{-1}$. We see that the cost function inverts the entire channel. As with a zero-forcing detector, this result can lead to noise enhancement if any of the zeros of the channel are near the unit circle.

Property 5-4. For all sub-Gaussian input alphabets, the vector CMA with GSC can resolve a channel up to an unknown unitary ambiguity and an unknown set of delays.

The transmitted vector can be recovered by appending a blind unitary estimator, that identifies and eliminates the unitary ambiguity, to the linear detector. Possible solutions for the blind unitary estimator include the MPLL (see Chapter 3), JADE, and EASI algorithms. In general, the delay values D_i are unknown, but they do not interfere with the reconstruction of the original input vector sequence $\{\boldsymbol{x}_k\}$.

5.5.3 Stochastic Algorithm

Recall that the vector CMA cost function with Gram-Schmidt constraint is given by (5-37):

$$J_{GS} = J_v + \sum_{l=-(M+N)}^{M+N} \left\| E[\boldsymbol{y}_k \boldsymbol{y}_{k-l}^*] - m_2 \mathbf{I} \delta_l \right\|_F^2, \tag{5-39}$$

where $\boldsymbol{y}_k = \mathbf{C}\mathbf{R}_k$. The cost function can be expressed in terms of the stacked-received vector $\mathbf{R}_k$ and the linear detector $\mathbf{C}$ as follows:

$$J_{GS} = J_v + \sum_{l=-(M+N)}^{M+N} tr\left[(\mathbf{C}\Phi_{\mathrm{R},l}\mathbf{C}^*\mathbf{C}\Phi_{\mathrm{R},-l}\mathbf{C}^*) - m_2\delta_l\mathbf{C}(\Phi_{\mathrm{R},l}+\Phi_{\mathrm{R},-l})\mathbf{C}^* + (m_2)^2\delta_l\mathbf{I} \right], \tag{5-40}$$

where $\Phi_{\mathrm{R},l} = E[\mathbf{R}_k \mathbf{R}_{k-l}^*]$. The complex gradient of (5-40) with respect to $\mathbf{C}$ is given by:

$$\nabla_{\mathbf{C}} J_{GS} = 4E[\boldsymbol{e}_k \mathbf{R}_k^*] + 4\left[\sum_{l=-(M+N)}^{M+N} E[\boldsymbol{y}_k \boldsymbol{y}_{k-l}^*] E[\boldsymbol{y}_k \mathbf{R}_{k+l}^*] \right] - m_2 E[\boldsymbol{y}_k \mathbf{R}_k^*]. \tag{5-41}$$

Substituting a stochastic approximation of (5-41) in the classical steepest-descent algorithm, we arrive at the following update equation for the linear detector:

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \mu\left[\boldsymbol{e}_k \mathbf{R}_k^* + \left(\sum_{l=-(M+N)}^{M+N} E[\boldsymbol{y}_k \boldsymbol{y}_{k-l}^*] \boldsymbol{y}_k \mathbf{R}_{k+l}^* \right) - m_2 \boldsymbol{y}_k \mathbf{R}_k^* \right], \tag{5-42}$$

where μ is the step size. The first term in the square brackets of (5-42) is the original update term for the vector constant-modulus algorithm, while the remaining terms in the square brackets penalize any undesirable local minima.

We observe that (5-42) requires an estimate of $E[\boldsymbol{y}_k \boldsymbol{y}_{k-l}^*]$ for $|l| \leq M+N$. Since $\boldsymbol{y}_k$ is a non-stationary vector random process, it is difficult to obtain an accurate estimate of $E[\boldsymbol{y}_k \boldsymbol{y}_{k-l}^*]$ for $|l| \leq M+N$. By substituting $E[\boldsymbol{y}_k \boldsymbol{y}_{k-l}^*] = \mathbf{C}_k \Phi_{\mathrm{R},l} \mathbf{C}_k^*$ into (5-42), we arrive at an alternative update equation for the linear detector:

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \mu\left[\boldsymbol{e}_k \mathbf{R}_k^* + \left(\sum_{l=-(M+N)}^{M+N} (\mathbf{C}_k \Phi_{\mathrm{R},l} \mathbf{C}_k^*) \boldsymbol{y}_k \mathbf{R}_{k+l}^* \right) - m_2 \boldsymbol{y}_k \mathbf{R}_k^* \right]. \tag{5-43}$$

An estimate of $\Phi_{\mathrm{R},l}$ for $|l| \leq M+N$ is now required instead of an estimate of $E[\boldsymbol{y}_k \boldsymbol{y}_{k-l}^*]$ for $|l| \leq M+N$. Since we have assumed that the input vector $\boldsymbol{x}_k$ and the channel $\mathbf{H}(z)$ are both stationary, it is easy to obtain an accurate estimate of $\Phi_{\mathrm{R},l}$ for $|l| \leq M+N$. In fact, the estimates can obtained by using a simple running average:

$$\hat{\Phi}_{\mathrm{R},l} = \frac{1}{k-l} \sum_{i=l+1}^{k} \mathbf{R}_i \mathbf{R}_{i-l}^*. \tag{5-44}$$

As k gets large, this sum approaches $\Phi_{\mathrm{R},l}$. Since we can estimate $\Phi_{\mathrm{R},l}$ fairly accurately we expect the update equation (5-43) to converge more quickly. In a practical implementation, $\hat{\Phi}_{\mathrm{R},l}$ should be substituted for $\Phi_{\mathrm{R},l}$ in (5-43).

As stated earlier in Section 4.6, $\mathbf{C}_k\Phi_{\mathrm{R},l}\mathbf{C}_k^*$ can be estimated more accurately than $E[\boldsymbol{y}_k\boldsymbol{y}_{k-l}^*]$. Although determining $\mathbf{C}_k\hat{\Phi}_{\mathrm{R},l}\mathbf{C}_k^*$ results in an increase in computational complexity, this additional term is based solely on second-order statistics, and therefore, should converge quickly and require less symbols to invert the channel. Furthermore, this complexity in (5-43) can be reduced by exploiting the redundancy in $\mathbf{C}\Phi_{\mathrm{R},l}$. The best way to see this redundancy is to look at the following example:

Example 5-2. If the channel $\mathbf{H}(z)$ has memory $M = 1$ and the linear detector $\mathbf{C}(z)$ has memory $N = 2$, then

$$\mathbf{C}_k\Phi_{\mathrm{R},l} = \begin{cases} [\mathbf{0} & \mathbf{0} & C_0\Phi_{-1}] & l=-3 \\ [\mathbf{0} & C_0\Phi_{-1} & C_0\Phi_0 + C_1\Phi_{-1}] & l=-2 \\ [C_0\Phi_{-1} & C_0\Phi_0 + C_1\Phi_{-1} & C_0\Phi_1 + C_1\Phi_0 + C_2\Phi_{-1}] & l=-1 \\ [C_0\Phi_0 + C_1\Phi_{-1} & C_0\Phi_1 + C_1\Phi_0 + C_2\Phi_{-1} & C_1\Phi_1 + C_2\Phi_0] & l=0 \\ [C_0\Phi_1 + C_1\Phi_0 + C_2\Phi_{-1} & C_1\Phi_1 + C_2\Phi_0 & C_2\Phi_1] & l=1 \\ [C_1\Phi_1 + C_2\Phi_0 & C_2\Phi_1 & \mathbf{0}] & l=2 \\ [C_2\Phi_1 & \mathbf{0} & \mathbf{0}] & l=3 \end{cases} \quad (5\text{-}45)$$

where $\Phi_j = E[\boldsymbol{r}_k\boldsymbol{r}_{k-j}^*]$ and $\mathbf{C}_k = [\mathbf{C}_0\ \mathbf{C}_1\ \mathbf{C}_2]$. We see that $\mathbf{C}_k\Phi_{\mathrm{R},l}$ is a left-shifted version of $\mathbf{C}_k\Phi_{\mathrm{R},l-1}$ where a new term is inserted on the right. Because of this structure, there is appreciable redundancy in (5-45). In fact, there are

only five non-redundant terms, excluding the zero matrices. Define **A** to be the matrix that contains the non-redundant terms:

$$\mathbf{A} = [\mathbf{C}_0\Phi_{-1},\ \mathbf{C}_0\Phi_0 + \mathbf{C}_1\Phi_{-1},\ \mathbf{C}_0\Phi_1 + \mathbf{C}_1\Phi_0 + \mathbf{C}_2\Phi_{-1},\ \mathbf{C}_1\Phi_1 + \mathbf{C}_2\Phi_0,\ \mathbf{C}_2\Phi_1]. \quad (5\text{-}46)$$

Notice that **A** can also be formed by taking the block-matrix convolution of the linear detector $\mathbf{C}_k$ with the stacked-correlation matrix $\Phi_r = [\Phi_{-1}\ \Phi_0\ \Phi_1]$, *i.e.*,

$$\mathbf{A} = \mathbf{C}_k * \Phi_r. \quad (5\text{-}47)$$

We should emphasize that (5-47) is an efficient method for calculating the non-redundant terms in (5-45). Also the matrix **A** can used to efficiently construct all of the matrices given by (5-45).

Using this example, we can develop an algorithm to efficiently compute $\mathbf{C}_k\Phi_{\mathrm{R},l}$ for $|l| \leq M+N$. Before, we can state the algorithm, we need the following definition:

Definition 5-9. Let $\mathbf{Z} = [\mathbf{Z}_0\ \mathbf{Z}_1\ \ldots\ \mathbf{Z}_N]$ where $\mathbf{Z}_i$ is an $n \times m$ matrix and let $\mathbf{Z}_{N+1}$ also be an $n \times m$ matrix. Define the block-shifting function L: $\mathbb{C}^{n\times(N+1)m}$, $\mathbb{C}^{n\times m} \rightarrow \mathbb{C}^{n\times(N+1)m}$ as follows:

$$L(\mathbf{Z}, \mathbf{Z}_{N+1}) = [\mathbf{Z}_1\ \ldots\ \mathbf{Z}_N\ \mathbf{Z}_{N+1}]. \quad (5\text{-}48)$$

The algorithm for computing $\mathbf{C}_k\Phi_{\mathrm{R},l}$ for $|l| \leq M+N$ is given by the following steps:

Step 1. Compute the block-matrix convolution of the linear detector $\mathbf{C}_k = [\mathbf{C}_0 \ \ldots \ \mathbf{C}_N]$ with the stacked-correlation matrix $\Phi_r = [\Phi_{-M} \ \ldots \ \Phi_0 \ \ldots \ \Phi_M]$. The augmented matrix $\mathbf{A}$ is formed by appending the block-matrix convolution with N $n \times m$ zero matrices, *i.e.*,

$$\mathbf{A} = [\mathbf{A}_{-(M+N)} \ \ldots \ \mathbf{A}_0 \ \ldots \ \mathbf{A}_{M+N}] = [\mathbf{C}_k * \Phi_r \ \mathbf{0}_{n\times m} \ \mathbf{0}_{n\times m} \ \ldots \ \mathbf{0}_{n\times m}]. \tag{5-49}$$

Step 2. Initialize $\mathbf{C}_k\Phi_{\mathrm{R},-(M+N)-1}$ to be an $n \times (N+1)m$ zero matrix.

Step 3. The matrix $\mathbf{C}_k\Phi_{\mathrm{R},l}$ for $|\,l\,| \le M+N$ is given by the following recursive update:

$$\mathbf{C}_k\Phi_{\mathrm{R},l} = L(\mathbf{C}_k\Phi_{\mathrm{R},l-1}, \mathbf{A}_l), \tag{5-50}$$

where $\mathbf{A}_l$ is defined in Step 1.

This method for computing $\mathbf{C}_k\Phi_{\mathrm{R},l}$ is very efficient because it only requires the computation of one block-matrix convolution.

In general, the stacked-correlation matrix is unknown, but since we have assumed that both input vector $\boldsymbol{x}_k$ and the channel $\mathbf{H}(z)$ are stationary, Φ_r can be estimated using a simple running average:

$$\hat{\Phi}_r = \frac{1}{k-M} \sum_{i\,=\,M\,+\,1}^{k} [\boldsymbol{r}_{i-M}\boldsymbol{r}_i^* \ \ldots \ \boldsymbol{r}_{i-1}\boldsymbol{r}_i^* \ \boldsymbol{r}_i\boldsymbol{r}_i^* \ \boldsymbol{r}_i\boldsymbol{r}_{i-1}^* \ \ldots \ \boldsymbol{r}_i\boldsymbol{r}_{i-M}^*]. \tag{5-51}$$

As k gets large, this sum approaches Φ_r. In a practical implementation, $\hat{\Phi}_r$ should be used instead of Φ_r in (5-49).

In summary, the vector constant-modulus algorithm with GSC is defined by (5-25), (5-43), Steps 1–3, and (5-51).

5.6 PERFORMANCE IN NOISE

In the previous section, we determined the local minima of the vector CMA cost function with GSC in the absence of noise. This analysis, though valid and informative, is only of academic interest since in any real-world application, noise is always present and must be included in the analysis. The analysis of the noisy vector CMA cost function, presented in the previous chapter, when extended to channels with memory unfortunately becomes intractable. We could use high SNR approximations in order to obtain a closed-form expression for the noisy local minima, but these results would not describe the cost function for all SNR.

Since it is difficult to obtain a closed-form expression of the noisy local minima, we have chosen to quantify the performance of the vector CMA with GSC detector experimentally. We can determine the minimum mean-squared error (MMSE) performance of this detector by implementing the stochastic algorithm for a large number of symbols and a very small step size. Unfortunately, the update algorithm may affect the performance of the detector. To minimize this effect, we initialized the linear detector with the zero-forcing solution, averaged the steady-state MSE over a large number of symbols, and used an ideal rotator to resolve the unitary ambiguity in the detector output. Initializing the linear detector at the zero-forcing solution allows the update algorithm to converge more quickly and more accurately to the noisy local minimum. By averaging the steady-state MSE over a large number of symbols, we reduce the effects of the update algorithm wan-

dering about the local minimum. Finally, using an ideal rotator instead of an adaptive algorithm, such as the MPLL, ensures that the only contribution to the MSE comes from the vector CMA with GSC detector. In spite of these measures to reduce the effects of the stochastic algorithm, the measured MMSE will still be higher than the true MMSE. However, by correctly choosing the number of symbols and the step size, the difference between these two values can be minimized.

In the following experiment, we quantify the performance of the vector CMA with GSC detector in the presence of noise and compare it with the theoretical performance of both the MMSE detector and the forward-backward LP detector [54]. We restrict the focus of this experiment to tall channels with memory, because in other cases the LP does not exist. To ensure that comparison among these detectors is fair, we assume that all three linear detectors have the same length and that the delay for each detector is optimal.

To facilitate the comparison between the vector CMA with GSC detector and the forward-backward linear predictor, we use an example similar to the one presented in [54].

> **Experiment 5-1.** Consider an 8×2 random channel with memory $M = 2$. The coefficients of the channel are drawn independently from a zero-mean unit-variance complex Gaussian distribution. The columns of the channel are scaled so that the energy of user 1 is 10 dB below that of user 2; in other words, $\mathrm{SIR}_1 = -10$ dB. The input alphabet is assumed to be 16-QAM. For each channel, we determined the optimal 3-tap vector CMA with GSC detector, the optimal 3-tap MMSE detector, and the optimal minimum-order forward-backward LP detector, which had at most 3 taps. The optimal vector

CMA with GSC detector was found by implementing the stochastic algorithm for all possible delays. For each delay, the stochastic algorithm was initialized with the zero-forcing solution and implemented for 200,000 symbols. The step size for each implementation decreased with time according to $\mu_k = 0.001/(1 + k/20000)$. The MMSE for each delay was calculated by averaging the MSE over the last 20,000 symbols. The optimal vector CMA with GSC detector was the one which produced the smallest MMSE. In Fig. 5-3, we plot the best MMSE_1 versus $\text{SNR}_1 = \sum_{j=1}^{m} \left\| \boldsymbol{h}_j^{(1)} \right\| / \sigma^2$, where $\boldsymbol{h}_j^{(1)}$

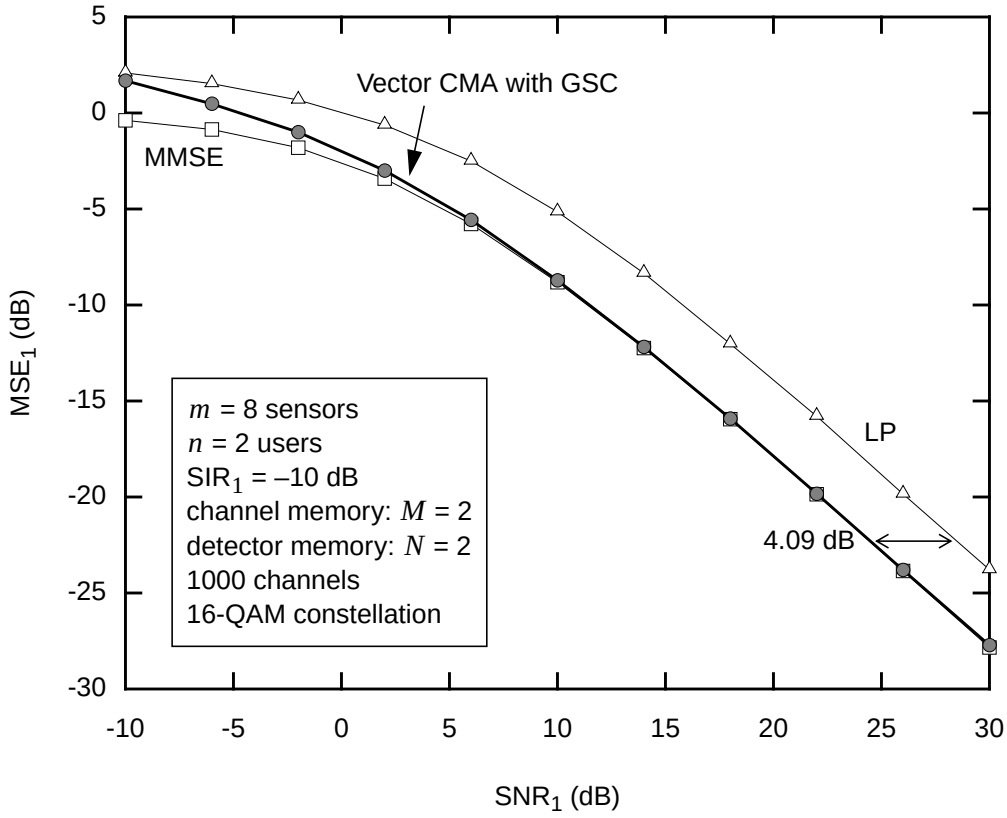


Fig. 5-3. A comparison of the mean-squared error versus SNR of the optimal vector CMA with GSC detector, the optimal minimum-MSE detector, and optimal forward-backward LP detector.

denotes the first column of the j-th channel tap, for each detector. The curves are the ensemble average of 1000 random channels. From these curves, we observe that the performance of the vector CMA with GSC detector is essentially similar to that of the MMSE detector, except at extremely low SNR values where the vector CMA with GSC detector starts to break down. We also see from the horizontal gap between the curves that the LP detector suffers an SNR penalty relative to both the MMSE detector and the vector CMA with GSC detector. The size of this penalty is roughly equivalent to the amount of energy discarded by the LP. In this particular example, the LP discarded an average of 39% of the total channel energy, which is equivalent to a gap of $1 / 0.39 = 4.09$ dB.

An important result of this experiment is that whereas the optimal MMSE detector and the optimal vector CMA with GSC detector achieve nearly identical performance, the optimal LP detector suffers a 4.09 dB penalty at high SNR. The relatively low performance of the LP detector is due to the fact that it inverts only a single tap of the channel, thereby throwing away the energy from the remaining $(M - 1)$ taps. It was stated in [54] that the performance of the LP detector suffers when the energy of the chosen tap is small and the noise is nonzero or when the channel energy is evenly distributed across all taps. In both cases an under-utilization of the channel energy occurs. In contrast, both the MMSE and the vector CMA with GSC detectors invert the entire channel, thereby using all of the channel energy.

5.7 EXPERIMENTAL RESULTS

We conclude this chapter with several computer experiments. In the first experiment, we consider a two-user, two-sensor uniform linear array with half-wavelength spacing, multipath, and a 16-QAM input alphabet. Since this channel is square, we compare the performance of the vector CMA with GSC detector with that of the decorrelation CMA based detector [101,102]. In the next experiment, we consider a 16×2 asynchronous CDMA application with a 16-QAM input alphabet. For this channel, we compare the performance of the vector CMA with GSC detector with that of the decorrelation CMA detector and of the forward linear predictor (LP) detector [54]. Finally, in the last two experiments, specific channels where the vector CMA with GSC detector outperforms the LP detector are considered.

5.7.1 Uniform Linear Array with Multipath

In the following experiment, we consider a 2-user, 2-sensor, uniform linear array with $\lambda/2$-spacing and multipath. This channel is a generalization of the memoryless channel presented in Section 4.7.2. We assume that each user draws symbols independently and uniformly from a 16-QAM input alphabet. The energy from each user is received at the linear array along two paths: a direct line-of-sight (LOS) path and a reflected path. Each path is characterized by its amplitude A, its propagation delay τ, and its angle of incidence θ. For user 1, the parameters are $(A, \tau, \theta) = (0.5, 0.4, -35°)$ and $(A, \tau, \theta) = (0.4, 1.1, 80°)$ for the LOS path and reflected path, respectively. For user 2, the parameters are $(A, \tau, \theta) = (0.8, 0.2, 10°)$ and $(A, \tau, \theta) = (0.1, 1.3, -65°)$ for the LOS path and reflected path, respectively. The transmit pulse for both users is a raised-cosine pulse with 100% excess bandwidth. In general, this channel has an infinite memory, but since more than 99% of the

energy is contained in first three taps, we can truncate the channel to have a memory of M = 2, without losing any significant amount of energy. We set SNR_1 = 30 dB and SNR_2 = 33 dB for this channel, so that SIR_1 = –3 dB.

Experiment 5-2. In Fig. 5-4, we plot MSE_1 versus time, for the optimal delay, of both the vector CMA with GSC detector and the decorrelation CMA detector. Each curve is an ensemble average of 500 different random input and noise sequences. The parameters for each were optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram, or equiva-

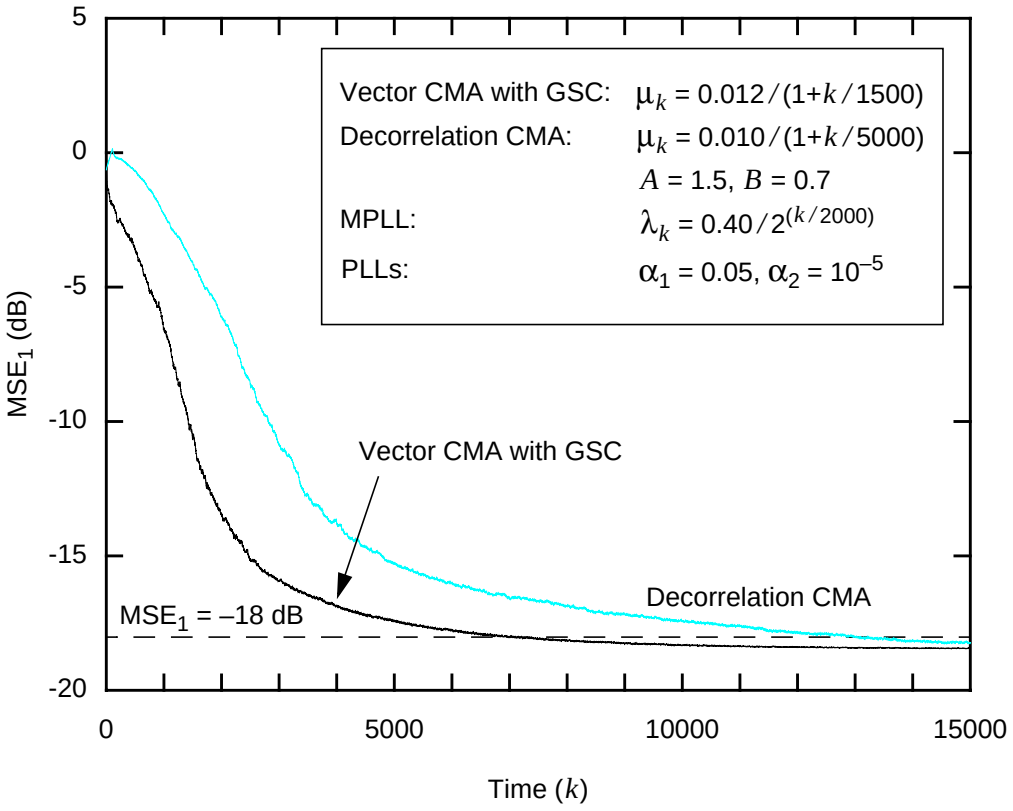


Fig. 5-4. Comparison of the vector CMA with GSC detector and decorrelation CMA detector, in terms of MSE_1 versus time, for a 2×2 uniform linear with multipath.

lently an $MSE_1 = -18$dB. The step size for the vector CMA with GSC detector and the decorrelation CMA detector ($A = 1.5$ and $B = 0.7$) were $\mu_{k,\text{vec}} = 0.012/(1+k/1500)$ and $\mu_{k,\text{dec}} = 0.010/(1+k/5000)$, respectively. We used a 100-point causal rectangular window to estimate the cross-correlation terms for the decorrelation CMA detector. The MPLL step size was decreased with time according to $\lambda_k = 0.40\ /2^{(-k/2000)}$. The bank of scalar PLLs parameters used by the decorrelation CMA detector were $\alpha_1 = 0.05$ and $\alpha_2 = 10^{-5}$. From these curves, we observe that both of these detector can open the eye diagram for this square channel, but the vector CMA with GSC detector needed only half as many symbols as did the decorrelation CMA detector.

The vector CMA with GSC detector converges much faster than the decorrelation CMA detector because the former detector requires an estimate of a stationary signal that can be determined fairly quickly and reasonably accurately, whereas the latter detector requires an estimate of a non-stationary signal that is somewhat difficult to obtain and which is usually not very accurate. The better the estimate, the more quickly each of these detectors converge. Another reason for the fast convergence of the vector CMA with GSC detector is due to the additional term in the cost function, which makes two contributions: it penalizes any cross-correlation among the users in both time and space, and it also aids the vector CMA cost function in restoring the modulus for each user. In contrast, the additional term in the decorrelation CMA cost function only penalizes the cross-correlations among the users, but the restoration of the moduli is left to the pointwise CMA cost function. Thus, the additional term in the vector CMA cost function with GSC not only eliminates the undesirable local minima, but also aids in reducing the convergence time.

5.7.2 Asynchronous CDMA

In the following experiment, we consider a two-user asynchronous direct-sequence CDMA application as shown in Fig. 5-5. This channel [54] is a generalization of the memoryless channel presented in Section 4.7.3. We assume that each users draws symbols independently and uniformly from a 16-QAM input alphabet. The spreading sequence for the *i*-th users is given by

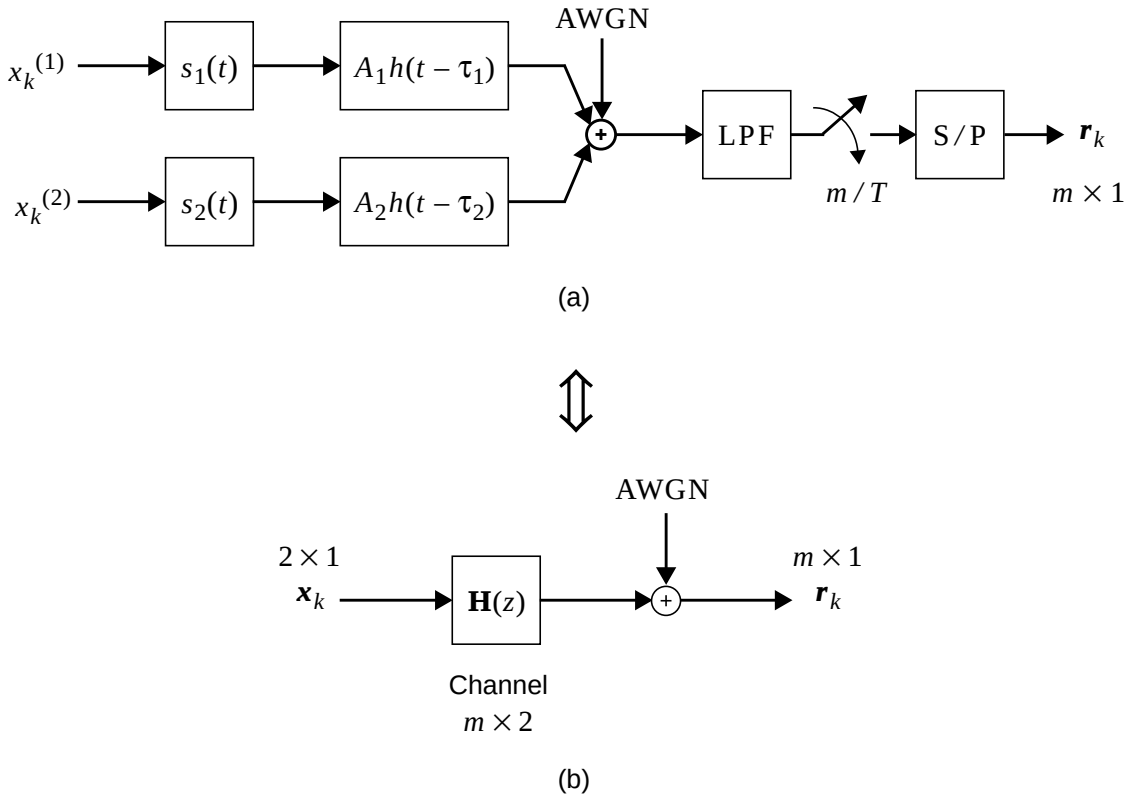


Fig. 5-5. Models for a two-user asynchronous CDMA application: (a) continuous-time model with a chip-rate receiver; (b) equivalent discrete-time model.

$$s_i(t) = \sum_{j=0}^{m-1} c_j^{(i)} p(t - j\frac{T}{m}), \tag{5-52}$$

where the chip-pulse shape $p(t) = \frac{sin(\pi t\, m/T)}{(\pi t\, m/T)}$ is an ideal sinc function with bandwidth equal to half the chip rate m/T and $c_j^{(i)}$ is the binary chip sequence for the *i*-th user. Each chip sequence has a length $m = 16$ and a period equal to one baud interval T. For this channel, the two binary chip sequences were generated randomly:

$$\{c_j^{(1)}\} = \{1, 1, 1, 1, 1, -1, -1, 1, 1, -1, -1, 1, -1, -1, 1, -1\} \tag{5-53}$$

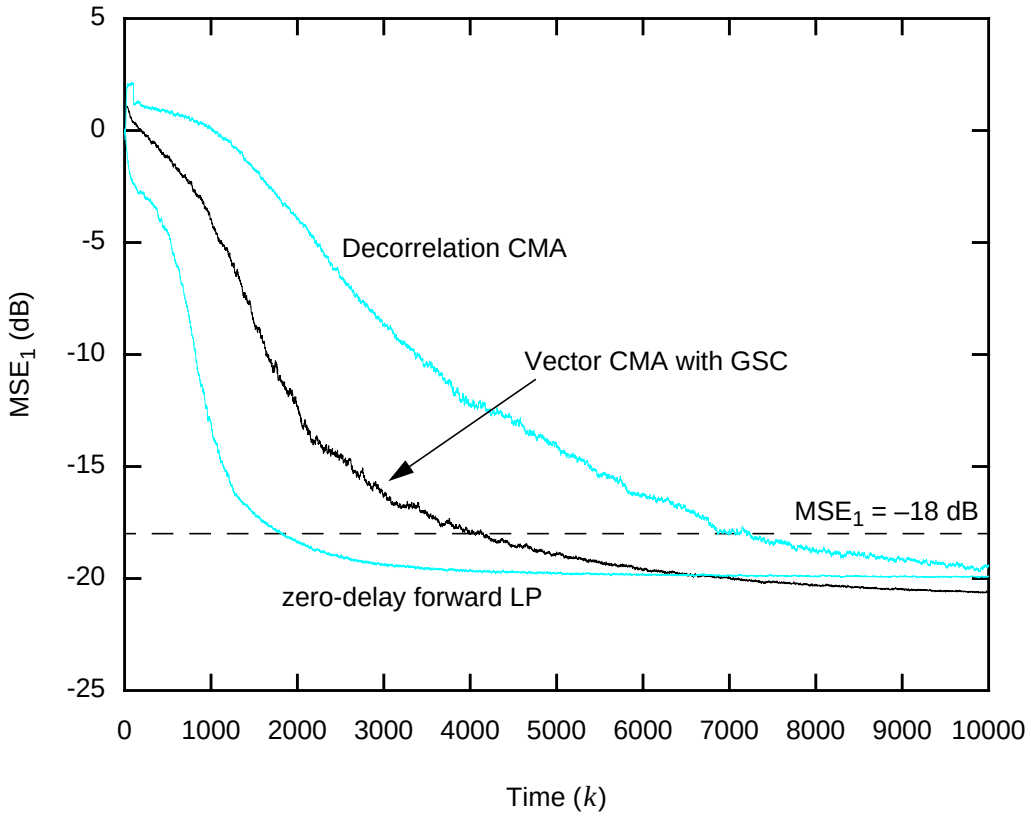
$$\{c_j^{(2)}\} = \{1, -1, -1, 1, 1, 1, -1, 1, 1, 1, -1, -1, -1, -1, -1, 1\} \tag{5-54}$$

The normalized correlation between the two binary chip sequences is given by $\rho = -0.125$.

Each CDMA signal is then passed through a channel with severe dispersion. This channel is modeled by a first-order low pass filter $h(t)$ that has a 3 dB bandwidth equal to one-fourth the chip rate, *i.e.*, $W = 1/4T_c$, where $T_c = T/m$ is the period of each chip. To generate an asynchronous model, we delay the first user by $\tau_1 = 0.7\ T_c$ and the second user by $\tau_2 = 5.2\ T_c$. The front-end of the receiver consists of an anti-aliasing filter followed by a chip-rate sampler and a serial-to-parallel (S/P) converter. Since the transmit filters have zero-excess bandwidth, the anti-aliasing filter is matched to the transmit filter. The serial-to-parallel converter groups the chip-rate samples into blocks of $m = 16$ to generate the baud-rate received vector $\boldsymbol{r}_k$. The resulting discrete-time channel transfer function matrix

$\mathbf{H}(z)$ is a 16×2 channel with memory $M = 1$. The amplitudes for each user A_1 and A_2 and the noise variance σ^2 were selected in such a way that $\text{SNR}_1 = 25$ dB and $\text{SNR}_2 = 35$ dB.

Experiment 5-3. In Fig. 5-6, we plot MSE_1 versus time, for the optimal delay of the vector CMA with GSC detector, the decorrelation CMA detector, and the forward LP detector. Each curve is an ensemble average of 500 different random input and noise sequences. The parameters for each detector were optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram, or equivalently an $\text{MSE}_1 = -18$dB. The step size for the vector CMA with GSC and the decorrelation CMA detector ($A = 1.0$ and $B = 1.0$) were $\mu_{k,\text{vec}} = 0.11/(1+k/1600)$ and $\mu_{k,\text{dec}} = 0.12/(1+k/4500)$, respectively. We used a 100-point causal rectangular window to estimate the cross-correlation terms for the decorrelation CMA detector. The step size for the forward LP detector was given by $\mu_{k,\text{lp}} = 0.06/(1+k/500)$ and the step size for the AGC was given by $\mu_{k,\text{agc}} = 0.01 \ / \ (1 + k/500)$. The step size for the MPLL was decreased with time according to $\lambda_k = 0.40/2^{(-k/2000)}$. The bank of scalar PLLs parameters used by the decorrelation CMA detector were $\alpha_1 = 0.05$ and $\alpha_2 = 10^{-5}$. From the curves in Fig. 5-6, we observe that all three detector can open the eye diagram for this tall channel. The forward LP detector is the fastest to converge, followed by the vector CMA with GSC detector and the decorrelation CMA detector in that order. Both the forward LP detector and the vector CMA with GSC detector can open the eye diagram in less than 4000 symbols, while the decorrelation CMA detector needs almost twice as many symbols.



Vector CMA with GSC:	$\mu_k = 0.011/(1+k/1600)$
Decorrelation CMA:	$\mu_k = 0.12/(1+k/4500)$
	$A = 1,\ B = 1$
LP:	$\mu_k = 0.06/(1+k/500)$
	$\beta_k = 0.01/(1+k/500)$
MPLL:	$\lambda_k = 0.40/2^{(k/2000)}$
PLLs:	$\alpha_1 = 0.05,\ \alpha_2 = 10^{-5}$

Fig. 5-6. Comparison of the vector CMA with GSC detector, decorrelation CMA detector, and a zero-delay forward LP detector, in terms of MSE_1 versus time, for a two-user asynchronous CDMA application.

The forward LP detector is designed to isolate and invert the first tap of the channel. This detector works well when channel energy is concentrated primarily in the first tap of the channel, as is the case for this particular two-tap channel in the above experiment. However, if the channel energy were concentrated primarily in the second tap and the energy of the first tap were significant relative to the noise power ($tr(\mathbf{H}_0\mathbf{H}_0^*) \approx \sigma^2$), then the performance of a forward LP detector would be unacceptable. (We will demonstrate this case in the next experiment.) We should point out that the vector CMA with GSC detector should have acceptable performance for either channel configuration.

5.7.3 Small First Tap

In the following experiment, we consider a two-tap 4×2 channel $\mathbf{H}(z)$ such that the channel energy is concentrated primarily in the second tap and the energy of the first tap is significant relative to the noise power ($tr(\mathbf{H}_0\mathbf{H}_0^*) \approx \sigma^2$). This channel is given as follows:

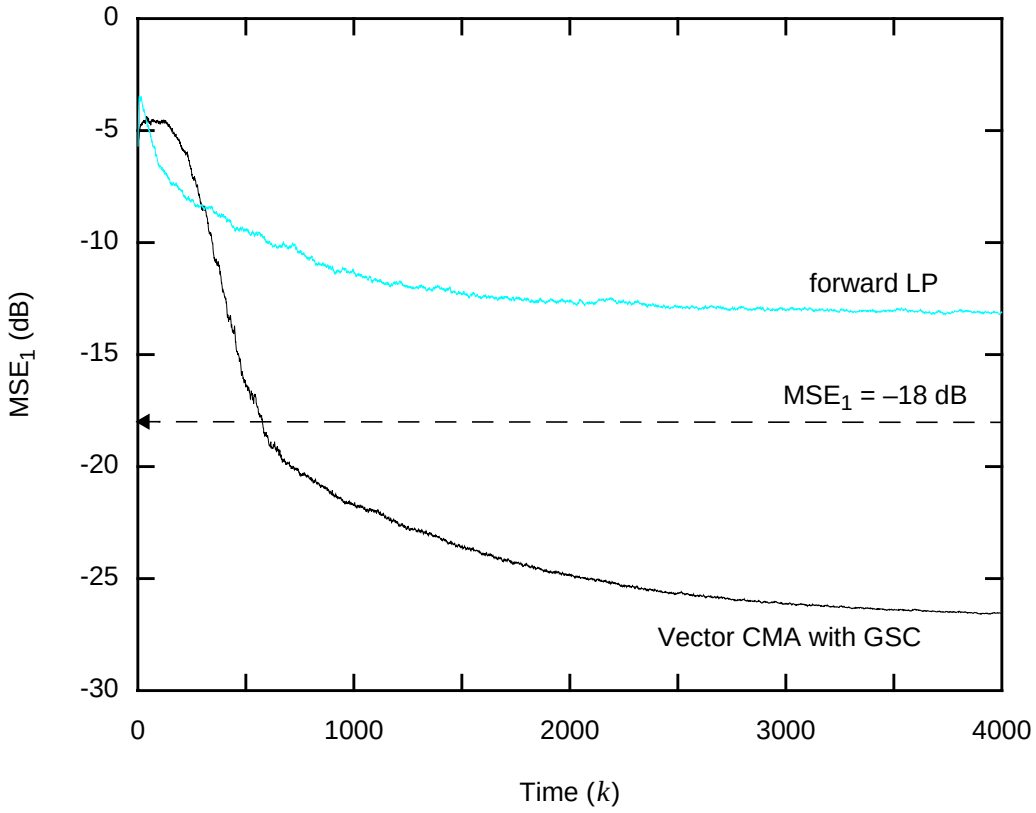
$$\mathbf{H}(z) =$$

$$\begin{bmatrix} 0.031 - j0.082 & 0.057 + j0.024 \\ 0.027 - j0.002 & -0.058 + j0.082 \\ 0.049 - j0.038 & -0.001 + j0.029 \\ 0.025 + j0.023 & -0.008 - j0.031 \end{bmatrix} + \begin{bmatrix} -0.606 + j0.144 & -0.310 + j0.00 \\ 0.097 - j0.381 & 0.204 - j0.124 \\ -0.399 - j0.007 & 0.085 + j0.422 \\ 0.534 - j0.018 & -0.355 - j0.722 \end{bmatrix} z^{-1}. \quad (5\text{-}55)$$

For this channel, we compare the performance of the vector CMA with GSC detector with that of the forward LP detector. The amplitudes for each user A_1 and A_2 and the noise variance σ^2 were selected in such a way that $\text{SNR}_1 = \text{SNR}_2 = 24$ dB, so that $\text{SIR}_1 = 0$ dB.

Experiment 5-4. In Fig. 5-7, we plot MSE_1 versus time, for the optimal delay, of the vector CMA with GSC detector and the forward LP detector. Each curve is an ensemble average of 500 different random input and noise sequences. The parameters for each detector were optimized to provide the fastest rate of convergence so as to achieve an open-eye diagram, or equivalently an $MSE_1 = -18$ dB. The step size was $\mu_{k,\text{vec}} = 0.01/(1+k/500)$ for the vector CMA with GSC detector, it was $\mu_{k,\text{lp}} = 0.05/(1+k/500)$ for the forward LP detector, and it was $\mu_{k,\text{agc}} = 0.02/(1+k/100)$ for the AGC. The step size for the MPLL was decreased with time according to $\lambda_k = 0.40/2^{(-k/2000)}$. From the curves in Fig. 5-7, we see that only the vector CMA with GSC detector was able to open the eye diagram. In fact, to open the eye diagram, this detector needed slightly more than 500 symbols. We also observe that the forward LP detector reaches a steady-state MSE of –13 dB. This value of the MSE implies that the eye diagram is not open and therefore the forward LP detector cannot recover the transmitted data.

This experiment demonstrates that the forward LP detector is not a viable option on a channel where the energy of the first tap is significant relative to the noise power ($tr(\mathbf{H}_0\mathbf{H}_0^*) \approx \sigma^2$). In fact, this result also extends to the forward-backward LP; the performance of this detector will also be unacceptable if the channel energy of the selected tap is significant relative to the noise power. Ideally, the forward-backward LP detector must choose the channel tap with the greatest energy. Since the channel is unknown, the channel energy distribution is also unknown. Hence, it is impossible choose *a priori* the channel tap with the greatest energy and as a result, the performance of any forward-backward



Vector CMA with GSC:	$\mu_k = 0.01/(1+k/500)$
LP:	$\mu_k = 0.05/(1+k/500)$
	$\beta_k = 0.02/(1+k/100)$
MPLL:	$\lambda_k = 0.40/2^{(k/2000)}$

Fig. 5-7. Comparison of the vector CMA with GSC detector and the forward LP, in terms of MSE_1 versus time, for a channel with a small first tap.

linear predictor will suffer when compared to the MMSE detector and the vector CMA with GSC detector, both of which work well on all channels irrespective of the channel energy distribution.

5.7.4 Non-Minimum Square Right Factor

A linear prediction detector is based on the concept that a tall channel is almost always minimum-phase. It is possible for a channel $\mathbf{H}(z)$ to be physically tall ($m > n$) and still not be minimum-phase. In this case, the channel is said to be *reducible*, which implies that there exists a square right-factor that is non-minimum phase, *i.e.*,

$$\mathbf{H}(z) = \tilde{\mathbf{H}}(z)\mathbf{G}(z), \tag{5-56}$$

where $\tilde{\mathbf{H}}(z)$ is an $m \times n$ minimum-phase channel and $\mathbf{G}(z)$ is a $n \times n$ non-minimum phase channel. For the channel described by (5-56), the LP detector will only be able to resolve the minimum-phase portion of this channel $\tilde{\mathbf{H}}(z)$, thus leaving the non-minimum phase portion of the channel unresolved. Hence, the output of the LP detector would still be corrupted by both intersymbol interference and multiuser interference. Thus, the LP detector is unable to resolve a reducible tall channel. In contrast, the vector CMA with GSC detector should still be able to invert this particular channel and recover the transmitted data.

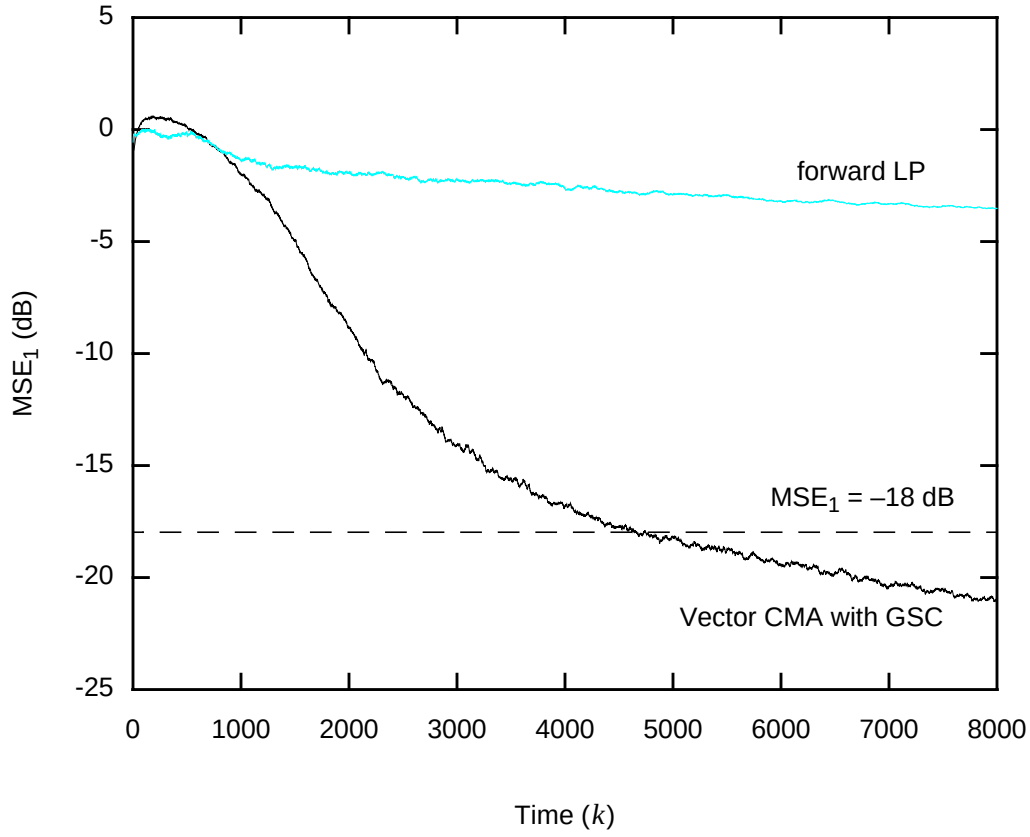
To demonstrate this point, we consider the following three-tap 4×2 channel $\mathbf{H}(z)$, which is a cascade of a two-tap 4×2 minimum-phase channel and a two-tap 2×2 non-minimum-phase channel:

$$\mathbf{H}(z) = \begin{bmatrix} 0.071 - j0.070 & -0.090 + j0.088 \\ 0.058 + j0.039 & -0.088 - j0.045 \\ -0.253 - j0.272 & 0.244 + j0.301 \\ -0.019 + j0.554 & -0.041 - j0.602 \end{bmatrix} + \begin{bmatrix} -0.093 + j0.232 & 0.024 - j0.238 \\ -0.086 - j0.210 & 0.170 + j0.257 \\ 0.072 + j0.230 & 0.200 + j0.180 \\ 0.006 + j0.075 & 0.191 - j0.391 \end{bmatrix} z^{-1} +$$

$$\begin{bmatrix} 0.023 + j0.170 & -0.002 - j0.017 \\ -0.147 - j0.042 & 0.031 + j0.027 \\ 0.253 + j0.119 & 0.089 - j0.018 \\ 0.386 - j0.288 & 0.001 - j0.191 \end{bmatrix} z^{-2}. \qquad (5\text{-}57)$$

We observe that the majority of the energy in this channel is concentrated in the first tap. For this channel, we compare the performance of the vector CMA with GSC detector with that of the forward LP detector. The amplitudes for each user, A_1 and A_2 were selected in such a way that $\text{SIR}_1 = -3$ dB.

Experiment 5-5. In Fig. 5-8, we plot MSE_1 versus time, for the optimal delay, of the vector CMA with GSC detector and the forward LP detector, assuming no noise. Each curve is an ensemble average of 500 different random input sequences. The parameters for each detector were optimized as to provide the fastest rate of convergence to achieve an open-eye diagram, or equivalently an $\text{MSE}_1 = -18$dB. The step size was $\mu_{k,\text{vec}} = 0.01/(1+k/4000)$ for the vector CMA with GSC detector, it was $\mu_{k,\text{lp}} = 0.05/(1+k/2000)$ for the vector CMA with GSC detector, and it was $\beta_{k,\text{agc}} = 0.01/(1+k/500)$ for the step size for the AGC. The step size for the MPLL was decreased with time according to $\lambda_k = 0.40/2^{(-k/2000)}$. From these curves, we see that only the vector CMA with GSC detector was able to open the eye diagram. In fact, this detector can open the eye diagram in less than 5000 symbols. We also observe



Vector CMA with GSC:	$\mu_k = 0.02/(1+k/4000)$
LP:	$\mu_k = 0.05/(1+k/2000)$
	$\beta_k = 0.01/(1+k/500)$
MPLL:	$\lambda_k = 0.40/2^{(k/2000)}$

Fig. 5-8. Comparison of the vector CMA with GSC detector and the forward LP, in terms of MSE_1 versus time, for a reducible tall channel.

that MSE of the forward LP detector approaches –4 dB in steady state. This value of the MSE implies that the transmitted data can never be recovered and therefore the forward LP detector is unacceptable.

These experiments demonstrate that the vector CMA with GSC detector can recover the transmitted data for a wide variety of channels: square, tall, minimum-phase, and non-minimum phase, so long as these channels are linearly separable and the input alphabet is sub-Gaussian. As shown in [114], the error surface for the CMA cost function becomes progressively "flatter" as the input alphabet approaches Gaussianity. Hence, one can expect prolonged convergence times. This result also extends to the vector CMA cost function with GSC. In contrast, a forward-backward LP detector can recover the transmitted data *only* for a specific class of channels: tall and minimum-phase, and it can do so irrespective of the distribution of the input alphabet.

5.8 SUMMARY

In this chapter, we have extended the definition for the vector CMA detector to channels with memory and have provided a detailed implementation of this blind detector. We have shown in Theorems 5-2 and 5-3 that, in the absence of noise, the vector CMA cost function is minimized by both unitary and non-unitary matrices when the input alphabet is sub-Gaussian. The unitary matrices are desirable because the unitary ambiguity can be estimated and resolved by the MPLL, which we have already developed. The non-unitary matrices, unfortunately, limit the usefulness of the vector CMA detector in the case of sub-Gaussian input alphabets. For a meso-Gaussian and super-Gaussian input alphabets, as shown in Theorems 5-4 and 5-5, the vector CMA cost function proved to be useless.

By exploiting the properties of the desirable minima, we were able to add an additional term to the vector CMA cost function that penalized the undesirable local minima. The modified cost function was referred to as the vector CMA cost function with Gram-Schmidt constraint (GSC). This cost function for a sub-Gaussian input alphabet is minimized by only unitary matrices. The additional term in this cost function, which is based solely on second-order statistics, assists in reducing the convergence time. The elimination of the undesirable local minima, unfortunately, is accompanied by an increase in computational complexity. We have shown that this computational burden can be reduced by exploiting the redundancy in the update equation. We have included a detailed implementation of the reduced-complexity vector CMA with GSC detector.

The local minima of the vector CMA cost function with GSC in the presence of noise could not be theoretically determined. Instead, we chose to experimentally quantify the performance. We have shown that the performance of the optimal vector CMA with GSC detector is nearly identical to that of the optimal MMSE detector.

Finally, we have shown through computer simulations that the vector CMA with GSC detector compares favorably in terms of performance and complexity to other known CMA-based blind multiuser detectors. Specific cases where this detector outperforms a LP detector were identified.

APPENDIX 5.1

PROOF OF THEOREM 5-2

The noiseless vector CMA cost function can be expressed in terms of the stacked-channel input vector $\mathbf{X}$ and the block-matrix transfer function $\mathbf{F}$ as follows[1]:

$$J_V(\mathbf{F}) = E\left[(\mathbf{X}^*\mathbf{F}^*\mathbf{F}\mathbf{X})^2 - 2M_V(\mathbf{X}^*\mathbf{F}^*\mathbf{F}\mathbf{X}) + M_V{}^2 \right]. \tag{5-58}$$

Since $\mathbf{F}^*\mathbf{F}$ is an Hermitian[2] matrix and the $rank(\mathbf{F}^*\mathbf{F}) \leq n$, this matrix has a unique truncated eigendecomposition: $\mathbf{F}^*\mathbf{F} = \mathbf{V}\mathbf{D}\mathbf{V}^*$, where $\mathbf{V}$ is an $(M+N+1)n \times n$ truncated unitary matrix and $\mathbf{D}$ is an $n \times n$ diagonal matrix. We see that $\mathbf{F}^*\mathbf{F}$ is also a positive-semidefinite[3] matrix, and hence the diagonal elements of $\mathbf{D}$ must be real and non-negative.

Let $\boldsymbol{u} = \mathbf{V}^*\mathbf{X}$ and let $\boldsymbol{w}$ denote an $n \times 1$ vector whose i-th component is given by

$$w_i = |u_i|^2 = |\boldsymbol{v}_i{}^*\mathbf{X}|^2, \tag{5-59}$$

where $\boldsymbol{v}_i$ is the i-th column of $\mathbf{V}$. The first and second terms of (5-58) can be simplified using the previous two definitions:

---

1. The dependence on time has been suppressed to simplify the notation.

2. The matrix $\mathbf{G}$ is Hermitian matrix if and only if $\mathbf{G} = \mathbf{G}^*$.

3. An Hermitian matrix $\mathbf{G}$ is positive semi-definite if and only if $\boldsymbol{r}^*\mathbf{G}\boldsymbol{r} \geq 0$ for all $\boldsymbol{r} \in \mathbb{C}^n$.

$$E[\mathbf{X}^*\mathbf{F}^*\mathbf{F}\mathbf{X}] = E[\boldsymbol{u}^*\mathbf{D}\boldsymbol{u}] = \boldsymbol{d}^T E[\boldsymbol{w}], \tag{5-60}$$

$$E[(\mathbf{X}^*\mathbf{F}^*\mathbf{F}\mathbf{X})^2] = E[(\boldsymbol{u}^*\mathbf{D}\boldsymbol{u})^2] = \boldsymbol{d}^T E[\boldsymbol{w}\boldsymbol{w}^T]\boldsymbol{d}, \tag{5-61}$$

where $\boldsymbol{d}$ is an $n \times 1$ vector composed of the diagonal elements of $\mathbf{D}$. The expectation of w_i is given by:

$$E[w_i] = E[\,|\,\boldsymbol{v}_i^*\mathbf{X}\,|^2] = \boldsymbol{v}_i^* E[\mathbf{X}\mathbf{X}^*]\boldsymbol{v}_i = m_2\,|\,\boldsymbol{v}_i\,|^2 = m_2, \tag{5-62}$$

where the third equality is due to the assumption that all users are independent and identically distributed with $E[\boldsymbol{x}\boldsymbol{x}^*] = m_2\mathbf{I}$. Therefore, $E[\boldsymbol{w}] = m_2\mathbf{1}_n$, where the $n \times 1$ vector $\mathbf{1}_n = [1 \ldots 1]^T$. Using the previous three equations, (5-58) can be written as:

$$J_v(\mathbf{V}, \boldsymbol{d}) = \boldsymbol{d}^T\mathbf{R}_{ww}\boldsymbol{d} - 2m_2 M_v \boldsymbol{d}^T\mathbf{1}_n + M_v^2, \tag{5-63}$$

where $\mathbf{R}_{ww} = E[\boldsymbol{w}\boldsymbol{w}^T]$. The exact structure and rank of $\mathbf{R}_{ww}$ are summarized by the following two lemmas.

Lemma 5-1: The matrix $\mathbf{R}_{ww}$ can be written as a linear combination of three matrices:

$$\mathbf{R}_{ww} = (m_2)^2\left[\mathbf{1}_n\mathbf{1}_n{}^T + \mathbf{I} + (\kappa - 2)\mathbf{B}^T\mathbf{B}\right], \tag{5-64}$$

where the (i, j)-th component of $\mathbf{B}$ is given by $[\mathbf{B}]_{ij} = |\,v_{ij}\,|^2$.

Proof: See Appendix 5.2.

Lemma 5-2: If $\kappa > 1$, then $\mathbf{R}_{ww}$ is a positive-definite matrix.

Proof: See Appendix 5.3.

We see that (5-63) is completely parameterized by $\mathbf{V}$ and $\boldsymbol{d}$. Since these two variables are independent, the local minima of (5-63) can be determined by first minimizing the cost function with respect to $\boldsymbol{d}$, and then with respect to $\mathbf{V}$. The gradient of (5-63) with respect to $\boldsymbol{d}$ is given by:

$$\nabla_d J_v = 2\mathbf{R}_{ww}\boldsymbol{d} - 2m_2 M_v \mathbf{1}_n. \tag{5-65}$$

The inflection points of the cost function occur when the gradient is equal to the zero vector, or equivalently when:

$$\boldsymbol{d} = m_2 M_v \mathbf{R}_{ww}{}^{-1}\mathbf{1}_n, \tag{5-66}$$

where the inverse of $\mathbf{R}_{ww}$ exists because the matrix is positive definite, and therefore full rank. The solution to this equation depends upon the exact value of $\mathbf{R}_{ww}$. The matrix $\mathbf{R}_{ww}$ that minimizes the cost function can be determined by substituting (5-66) into (5-63). After rearranging some of terms, we find that the vector CMA cost function reduces to:

$$J_v(\mathbf{V}) = M_v{}^2\left[1 - (m_2)^2 \mathbf{1}_n{}^T \mathbf{R}_{ww}{}^{-1}\mathbf{1}_n\right]. \tag{5-67}$$

This equation is only a function of $\mathbf{R}_{ww}$, which in turn is a function of $\mathbf{V}$. Therefore, minimizing the cost function is equivalent to maximizing $(m_2)^2 \mathbf{1}_n{}^T \mathbf{R}_{ww}{}^{-1} \mathbf{1}_n$ with respect to $\mathbf{V}$.

Using the matrix inversion lemma[4], the quantity $(m_2)^2 \mathbf{1}_n{}^T \mathbf{R}_{ww}{}^{-1} \mathbf{1}_n$ can be written as:

$$(m_2)^2 \mathbf{1}_n{}^T \mathbf{R}_{ww}{}^{-1} \mathbf{1}_n = (1 + \frac{1}{\gamma})^{-1}, \tag{5-68}$$

where $\gamma = \mathbf{1}_n{}^T [\mathbf{I} + (\kappa - 2)\mathbf{B}^T \mathbf{B}]^{-1} \mathbf{1}_n$. We observe that maximizing $(m_2)^2 \mathbf{1}_n{}^T \mathbf{R}_{ww}{}^{-1} \mathbf{1}_n$ is equivalent to maximizing γ. Since $\mathbf{B}^T \mathbf{B}$ is an Hermitian and a positive-semidefinite matrix, it has a unique eigendecomposition: $\mathbf{B}^T \mathbf{B} = \mathbf{Q} \Sigma \mathbf{Q}^T$, where $\mathbf{Q}$ is an $n \times n$ unitary matrix and Σ is an $n \times n$ diagonal matrix with non-negative real entries. The quantity γ can be expressed in terms of this eigendecomposition as follows:

$$\gamma = \sum_{j=1}^{n} \frac{1}{1 + (\kappa - 2)\sigma_{jj}} (\mathbf{1}_n{}^T \boldsymbol{q}_j)^2, \tag{5-69}$$

where σ_{jj} is the j-th diagonal element of Σ and where $\boldsymbol{q}_j$ is the j-th column of $\mathbf{Q}$. Notice that maximizing γ is equivalent to simultaneously maximizing $\mathbf{1}_n{}^T \boldsymbol{q}_j$ and simultaneously minimizing $[1 + (\kappa - 2)\sigma_{jj}] \; \forall \; j$. Clearly, $\boldsymbol{q}_1 = \frac{1}{\sqrt{n}} \mathbf{1}_n$ maximizes the numerator, and since the columns of $\mathbf{Q}$ are orthogonal, (5-69) reduces to:

$$\gamma = \frac{n}{1 + (\kappa - 2)\sigma_{11}}. \tag{5-70}$$

---

4. The matrix inversion lemma is $(\mathbf{A} + \mathbf{BCD})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}(\mathbf{DA}^{-1}\mathbf{B} + \mathbf{C}^{-1})^{-1}\mathbf{DA}^{-1}$.

For a non-CM sub-Gaussian input alphabet ($1 < \kappa < 2$), the denominator is minimized by choosing σ_{11} as large as possible; the largest eigenvalue corresponding to this eigenvector occurs when $\sigma_{11} = 1$. Thus, γ is maximized when

$$\mathbf{B}^T\mathbf{B}\mathbf{1}_n = \mathbf{1}_n. \tag{5-71}$$

The question remains: for what values of **V** does (5-71) hold?

Multiplying (5-71) on the right-hand side by $\mathbf{1}_n{}^T$, we find that

$$\left\|\mathbf{B}\mathbf{1}_n\right\|^2 = n \Leftrightarrow \sum_{i=1}^{T}\left|\sum_{j=1}^{n}|v_{ij}|^2\right|^2 = n, \tag{5-72}$$

where $T = (M+N+1)n$. A property of the matrix **B** is that all of the components sum to n:

$$\sum_{i=1}^{T}\sum_{j=1}^{n}|v_{ij}|^2 = n. \tag{5-73}$$

Subtracting (5-73) from (5-72), and rearranging some of the terms, we obtain:

$$\sum_{i=1}^{T}\left[\sum_{j=1}^{n}|v_{ij}|^2\left(\sum_{j=1}^{n}|v_{ij}|^2 - 1\right)\right] = 0, \tag{5-74}$$

which holds if and only if the i-th row of **B** sums to zero or the i-th row sums to one. From (5-73), we deduce that only n rows of **B** can sum to one, while the remaining rows must sum to zero; in other words,

$$\sum_{j=1}^{n} |v_{i_p j}|^2 = \begin{cases} 1 & \text{where } p = 1,\ldots, n, \text{ and } i_p \in \{1,\ldots,T\} \text{ s.t. } i_p \neq i_q \text{ when } p \neq q \\ 0 & \text{otherwise} \end{cases}. \quad (5\text{-}75)$$

Thus, the matrix **V**, which satisfies (5-75) and equivalently (5-71), is given by:

$$\mathbf{V} = \mathbf{P}\begin{bmatrix} \mathbf{I}_{n\times n} \\ \mathbf{0}_{(T-n)\times n} \end{bmatrix}\mathbf{Q}, \quad (5\text{-}76)$$

where **Q** is an arbitrary $n \times n$ unitary matrix and **P** is a $(M+N+1)n \times (M+N+1)n$ real permutation matrix. Substituting (5-76) into (5-66), we find that the optimal $\boldsymbol{d}$ is given by $\boldsymbol{d} = \mathbf{1}_n$, or equivalent $\mathbf{D} = \mathbf{I}$. This inflection point is a local minima because the Hessian of (5-65) with respect to $\boldsymbol{d}$, which is given by

$$\nabla_{dd} J_v = 2\mathbf{R}_{ww}, \quad (5\text{-}77)$$

is positive definite for a non-CM sub-Gaussian input alphabet (see Lemma 5-2).

Using the fact that $\mathbf{F}^*\mathbf{F} = \mathbf{V}\mathbf{D}\mathbf{V}^*$, we find that the vector CMA cost function for a non-CM sub-Gaussian input alphabet is minimized if and only if

$$\mathbf{F} = \mathbf{U}\begin{bmatrix} \mathbf{I}_{n\times n} & \mathbf{0}_{n\times(T-n)} \end{bmatrix}\mathbf{P}^T, \quad (5\text{-}78)$$

where **U** is an arbitrary $n \times n$ unitary matrix. Finally, mapping the block-matrix notation back to the z-domain, we see that the vector CMA cost function for a non-CM sub-Gaussian input alphabet is minimized if and only if

$$\mathbf{F}(z) = \mathbf{U}\mathbf{P}(z), \tag{5-79}$$

where $\mathbf{P}(z)$ is an $n \times n$ matrix which has only one nonzero entry of the form z^{-D}, where D is an arbitrary delay value per row. If there is more than one nonzero entry in any column, then the delay values in that column must all be different. ❑

APPENDIX 5.2

PROOF OF LEMMA 5-1

We recall from (5-59) that the i-th component of $\boldsymbol{w}$ is given by:

$$w_i = |\boldsymbol{v}_i^* \mathbf{X}|^2 = \left| \sum_{l=1}^{T} v_{li}^* x_l \right|^2, \tag{5-80}$$

where $T = (M+N+1)n$ is the total number of elements in the vector $\mathbf{X}$. Thus, the (i,j)-th component of $\mathbf{R}_{ww}$ can be written as:

$$[\mathbf{R}_{ww}]_{ij} = E[w_i w_j^*], \tag{5-81}$$

$$= E\left[\left| \sum_{l=1}^{T} v_{li}^* x_l \right|^2 \right] \left[\left| \sum_{p=1}^{T} v_{pj}^* x_p \right|^2 \right], \tag{5-82}$$

$$= \sum_{l=1}^{T} \sum_{m=1}^{T} \sum_{p=1}^{T} \sum_{q=1}^{T} v_{li}^* v_{mi} v_{pj}^* v_{qj} E[x_l x_m^* x_p x_q^*]. \tag{5-83}$$

We can use the following identity: $E[x_k x_m^* x_p x_q^*] = (m_2)^2[\delta_{km}\delta_{pq}+\delta_{kq}\delta_{mp}+(\kappa-2)\delta_{kmpq}]$, to simplify (5-83):

$$[\mathbf{R}_{ww}]_{ij} = (m_2)^2\Bigg[\sum_{l=1}^{T}|v_{li}|^2\sum_{p=1}^{T}|v_{pj}|^2 + \sum_{l=1}^{T}v_{lj}v_{li}^{*}\sum_{p=1}^{T}v_{pi}v_{pj}^{*} + (\kappa-2)\sum_{l=1}^{T}|v_{li}|^2|v_{lj}|^2\Bigg], \tag{5-84}$$

$$= (m_2)^2\Bigg[1 + \delta_{ij} + (\kappa-2)\Bigg(\sum_{l=1}^{T}|v_{li}|^2|v_{lj}|^2\Bigg)\Bigg], \tag{5-85}$$

$$= (m_2)^2\Big[[\mathbf{1}_n\mathbf{1}_n{}^T]_{ij} + [\mathbf{I}]_{ij} + (\kappa-2)[\mathbf{B}^T\mathbf{B}]_{ij}\Big], \tag{5-86}$$

where we have defined $[\mathbf{B}]_{ij} = |v_{ij}|^2$.

Hence, $\mathbf{R}_{ww}$ can be expressed as a linear combination of three matrices:

$$\mathbf{R}_{ww} = (m_2)^2\Big[\mathbf{1}_n\mathbf{1}_n{}^T + \mathbf{I} + (\kappa-2)\mathbf{B}^T\mathbf{B}\Big], \tag{5-87}$$

where $[\mathbf{B}]_{ij} = |v_{ij}|^2$. ❑

APPENDIX 5.3

PROOF OF LEMMA 5-2

We recall from (5-64) that the matrix $\mathbf{R}_{ww}$ can be written as follows:

$$\mathbf{R}_{ww} = (m_2)^2\Big[\mathbf{1}_n\mathbf{1}_n{}^T + (\mathbf{I} - \mathbf{B}^T\mathbf{B}) + (\kappa{-}1)\mathbf{B}^T\mathbf{B}\Big], \tag{5-88}$$

where $[\mathbf{B}]_{ij} = |v_{ij}|^2$. The matrix $\mathbf{1}_n\mathbf{1}_n{}^T$ is clearly positive semidefinite, because its eigenvalues are zero, with multiplicity n–1, and n. The matrix $(\kappa{-}1)\mathbf{B}^T\mathbf{B}$ is also positive semidefinite, because $\boldsymbol{r}^*\mathbf{B}^T\mathbf{B}\boldsymbol{r} = \|\mathbf{B}\boldsymbol{r}\|^2 \geq 0$ for all $\boldsymbol{r} \in \mathbb{C}^n$ and $(\kappa{-}1) > 0$ for a non-CM input alphabets.

The third matrix in (5-88) is positive semidefinite if $\boldsymbol{r}^*(\mathbf{I} - \mathbf{B}^T\mathbf{B})\boldsymbol{r} = \|\boldsymbol{r}\|^2 - \|\mathbf{B}\boldsymbol{r}\|^2 \geq 0$ for all $\boldsymbol{r} \in \mathbb{C}^n$. We observe that:

$$\|\mathbf{B}\boldsymbol{r}\|^2 = \sum_{i=1}^{T}\left|\sum_{j=1}^{n}|v_{ij}|^2 r_j\right|^2. \tag{5-89}$$

Define $p_j = \frac{1}{\alpha_i}|v_{ij}|^2$, where $\alpha_i = \sum_{j=1}^{n}|v_{ij}|^2$, so that $\sum_{j=1}^{n}p_j = 1$. Since $\mathbf{V}$ is a truncated unitary matrix, $0 < \alpha_i \leq 1$. We can view the sum $\sum_{j=1}^{n}p_j r_j$ as an expectation $E[R]$, where R is a random variable over the set $\{r_j\}$ with probability mass function $\{p_j\}$. From Jensen's inequality [107], the square of the mean cannot exceed the second moment:

$$\left| \sum_{j=1}^{n} p_j r_j \right|^2 = |\mathrm{E}[R]|^2 \leq \mathrm{E}[|R|^2] = \sum_{j=1}^{n} p_j |r_j|^2. \tag{5-90}$$

Using the fact that $p_j = \frac{1}{\alpha_i} |v_{ij}|^2$ and that α_i is positive, we simplify the inequality given by (5-90) to the following:

$$\left| \sum_{j=1}^{n} |v_{ij}|^2 r_j \right|^2 \leq \alpha_i \sum_{j=1}^{n} |v_{ij}|^2 |r_j|^2 \leq \sum_{j=1}^{n} |v_{ij}|^2 |r_j|^2, \tag{5-91}$$

where the last inequality is due to the fact that $0 < \alpha_i \leq 1$. Hence, (5-89) is upper-bounded by:

$$\|\mathbf{B}\boldsymbol{r}\|^2 \leq \sum_{i=1}^{T} \sum_{j=1}^{n} |v_{ij}|^2 |r_j|^2 = \sum_{j=1}^{n} |r_j|^2 \sum_{i=1}^{T} |v_{ij}|^2 = \sum_{j=1}^{n} |r_j|^2 = \|\boldsymbol{r}\|^2, \tag{5-92}$$

where $\sum_{i=1}^{T} |v_{ij}|^2 = 1$ because the columns of $\mathbf{V}$ have unit length. This equation implies that $\|\boldsymbol{r}\|^2 - \|\mathbf{B}\boldsymbol{r}\|^2 \geq 0$ for all $\boldsymbol{r} \in \mathbb{C}^n$ and so $(\mathbf{I} - \mathbf{B}^T\mathbf{B})$ is a positive-semidefinite matrix.

Since the sum of positive-semidefinite matrices is positive semidefinite [106], $\mathbf{R}_{ww}$ is a positive-semidefinite matrix. The matrix $\mathbf{R}_{ww}$ therefore can be either singular or nonsingular. If this matrix is assumed to be singular, then there exists a nonzero vector $\boldsymbol{r}$ such that:

$$\boldsymbol{r}^* \mathbf{R}_{ww} \boldsymbol{r} = 0 \quad \Leftrightarrow \quad (m_2)^2 \boldsymbol{r}^* \left[\mathbf{1}_n \mathbf{1}_n{}^T + (\mathbf{I} - \mathbf{B}^T\mathbf{B}) + (\kappa - 1)\mathbf{B}^T\mathbf{B} \right] \boldsymbol{r} = 0, \tag{5-93}$$

$$\Leftrightarrow (m_2)^2\left[\, |\, \mathbf{1}_n{}^T\boldsymbol{r}\,|^{\,2} + (\|\boldsymbol{r}\|^2 - \|\mathbf{B}\boldsymbol{r}\|^2) + (\kappa-1)\|\mathbf{B}\boldsymbol{r}\|^2 \,\right] = 0, \qquad (5\text{-}94)$$

$$\Leftrightarrow \mathbf{1}_n{}^T\boldsymbol{r} = 0 \text{ and } \|\boldsymbol{r}\|^2 = \|\mathbf{B}\boldsymbol{r}\|^2 \text{ and } \|\mathbf{B}\boldsymbol{r}\|^2 = 0, \qquad (5\text{-}95)$$

where the third implication is due to the fact that each term in (5-94) is non-negative. From (5-95), we see that the last two conditions imply that $\|\boldsymbol{r}\|^2 = 0$, which is impossible since we have assumed that $\boldsymbol{r}$ to be nonzero. Hence, $\mathbf{R}_{ww}$ can never be singular. The matrix $\mathbf{R}_{ww}$ must therefore be nonsingular and also positive definite for $\kappa > 1$. ❑

APPENDIX 5.4

PROOF OF THEOREM 5-3

In Appendix 5.1, we showed that minimizing the vector CMA cost function is equivalent to maximizing (5-69). For a CM input alphabet ($\kappa = 1$), γ is maximized if and only if $(\mathbf{I} - \mathbf{B}^T\mathbf{B})$ is singular. We observe that $(\mathbf{I} - \mathbf{B}^T\mathbf{B})$ is singular if and only if there exists a nonzero vector $\boldsymbol{r} \in \mathbb{C}^n$ such that:

$$\boldsymbol{r}^*(\mathbf{I} - \mathbf{B}^T\mathbf{B})\boldsymbol{r} = 0 \Leftrightarrow \left\|\mathbf{B}\boldsymbol{r}\right\|^2 = \left\|\boldsymbol{r}\right\|^2. \tag{5-96}$$

The question remains: for what values of $\mathbf{V}$ does (5-96) hold?

In Appendix 5.3, we showed that

$$\left\|\mathbf{B}\boldsymbol{r}\right\|^2 = \sum_{i=1}^{T}\left|\sum_{j=1}^{n}|v_{ij}|^2 r_j\right|^2 \le \sum_{i=1}^{T}\sum_{j=1}^{n}|v_{ij}|^2|r_j|^2 = \left\|\boldsymbol{r}\right\|^2. \tag{5-97}$$

Equality in (5-97) is achieved when the random variable R is no longer random, but is deterministic. This random variable becomes deterministic if the components of $\boldsymbol{r}$ are equal to a constant for all nonzero entries on the i-th row of $\mathbf{V}$. For example, R is deterministic when $\boldsymbol{r} = \alpha\mathbf{1}_n$ for some nonzero constant α. This vector achieves equality in (5-97) only when each row of $\mathbf{B}$ sums to either one or zero. Because of the way $\mathbf{B}$ is constructed,

this type of matrix can never exists, and hence $\boldsymbol{r} = \alpha \mathbf{1}_n$ cannot lie in the nullspace of ($\mathbf{I} - \mathbf{B}^T\mathbf{B}$).

If $\mathbf{V}$ is a partial block-diagonal matrix, then the vector $\boldsymbol{r}$ can be subdivided into disjoint subvectors corresponding to the nonzero blocks of $\mathbf{V}$. In this case, the random variable R is deterministic if the components of each subvector are equal to a constant. By choosing the constant for each subvector carefully, we can achieve equality in (5-97). If a row of $\mathbf{V}$ has unit length, then the constant for the subvector corresponding to the nonzero entries on that row should be nonzero. However, if the row does not have unit length, then the constant for the subvector corresponding to the nonzero entries on that row must be zero. Using this insight, we see that ($\mathbf{I} - \mathbf{B}^T\mathbf{B}$) is singular if and only if the matrix $\mathbf{V}$ has the form:

$$\mathbf{V} = \mathbf{P}_L \left[\begin{array}{ccc|c} \mathbf{Q}_1 & & & \\ & \mathbf{Q}_2 & & \\ & & & \mathbf{0} \\ & & \mathbf{Q}_P & \\ \hline & \mathbf{0} & & \mathbf{W} \end{array}\right] \mathbf{P}_R, \tag{5-98}$$

where $\mathbf{Q}_i$ is a $k_i \times k_i$ unitary matrix with $k_i \in \{1, \ldots, n\}$ satisfying $1 \le k = \sum_{i=1}^{P} k_i \le n$, $\mathbf{W}$ is an $((M+N+1)n-k) \times (n-k)$ truncated unitary matrix, and $\mathbf{P}_L$ and $\mathbf{P}_R$ are $(M+N+1)n \times (M+N+1)n$ and $n \times n$ real permutation matrices, respectively. The vector $\boldsymbol{r}$ that lies in the nullspace of ($\mathbf{I} - \mathbf{B}^T\mathbf{B}$) has the form:

$$\boldsymbol{r} = \mathbf{P}_R{}^T \begin{bmatrix} \alpha_1 \mathbf{1}_{k_1} \\ \alpha_2 \mathbf{1}_{k_2} \\ \vdots \\ \alpha_P \mathbf{1}_{k_P} \\ \mathbf{0}_{n-k} \end{bmatrix}, \tag{5-99}$$

where α_i is some nonzero constant.

Substituting (5-98) into (5-65), we find that the optimal d is given by:

$$\boldsymbol{d} = \mathbf{P}_R{}^T \left\{ \begin{bmatrix} \mathbf{1}_k \\ \mathbf{0}_{n-k} \end{bmatrix} + \begin{bmatrix} \alpha_1 \mathbf{1}_{k_1} \\ \alpha_2 \mathbf{1}_{k_2} \\ \vdots \\ \alpha_P \mathbf{1}_{k_P} \\ \mathbf{0}_{n-k} \end{bmatrix} \right\}, \tag{5-100}$$

where α_i is a nonzero constant satisfying $\sum_{i=1}^{P} \alpha_i k_i = 0$ and $d_i \geq 0$. We observe that all solutions for $\boldsymbol{d}$ satisfy the following property: $\sum_{i=1}^{n} d_i = n$, or equivalently, $tr(\mathbf{D}) = n$. Thus, the vector CMA for a CM input alphabet is minimized if and only if $\mathbf{F}^*\mathbf{F} = \mathbf{V}\mathbf{D}\mathbf{V}^*$, where $\mathbf{V}$ is given by (5-98) and where $\mathbf{D}$ is a non-negative diagonal matrix satisfying $tr(\mathbf{D}) = n$. By expanding the product $\mathbf{F}^*\mathbf{F}$, it easy to show that $\mathbf{F}^*\mathbf{F} = \tilde{\mathbf{D}}$, where $\tilde{\mathbf{D}}$ is a diagonal matrix, whose diagonal entries are a permutation of the diagonal entries of $\mathbf{D}$.

Finally, the vector CMA cost function for a CM input alphabet is minimized if and only if

$$\mathbf{F} = \mathbf{U}\mathbf{D}^{1/2}\left[\mathbf{I}_{n\times n}\ \mathbf{0}_{n\times(T-n)}\right]\mathbf{P}^{T}, \tag{5-101}$$

where $\mathbf{U}$ is an arbitrary $n \times n$ unitary matrix, $\mathbf{D}$ is a non-negative real diagonal matrix satisfying $tr(\mathbf{D}) = n$, and $\mathbf{P}$ is an $(M+N+1)n \times (M+N+1)n$ real permutation matrix. Mapping the block-matrix notation back to the z-domain, we then see that the vector CMA cost function is minimized if and only if

$$\mathbf{F}(z) = \mathbf{U}\mathbf{D}^{1/2}\mathbf{P}(z), \tag{5-102}$$

where $\mathbf{P}(z)$ is an $n \times n$ matrix which has only one nonzero entry of the form z^{-D}, where D is an arbitrary delay value per row. If there is more than one nonzero entry in any column, then the delay values in that column must all be different. ❑

APPENDIX 5.5

PROOF OF THEOREM 5-4

If the input alphabet is meso-Gaussian ($\kappa = 2$), then $\mathbf{R}_{ww}$ given in (5-64) reduces to:

$$\mathbf{R}_{ww} = (m_2)^2\left[\mathbf{1}_n\mathbf{1}_n{}^T + \mathbf{I}\right]. \tag{5-103}$$

Observe that $\mathbf{R}_{ww}$ is independent of $\mathbf{V}$, and therefore, the solution to (5-66) is also independent of $\mathbf{V}$. Substituting (5-103) into (5-66), we find that the optimal $\boldsymbol{d}$ is given by $\boldsymbol{d} = \mathbf{1}_n$, or equivalently $\mathbf{D} = \mathbf{I}$. This inflection is a local minima because the Hessian of (5-65) with respect to $\boldsymbol{d}$, which is given by

$$\nabla_{dd}J_v = 2\mathbf{R}_{ww}, \tag{5-104}$$

is positive definite for a meso-Gaussian input alphabet (see Lemma 5-2).

Using the fact that $\mathbf{F}^*\mathbf{F} = \mathbf{V}\mathbf{D}\mathbf{V}^*$, we find that the vector CMA cost function for a meso-Gaussian input alphabet is minimized if and only if

$$\mathbf{F} = \mathbf{U}, \tag{5-105}$$

where $\mathbf{U}$ is an $(M+N+1)n \times n$ truncated unitary matrix. Observe that each row of $\mathbf{F}$ has unit length, and therefore, $\mathbf{F}$ is an energy-preserving filter. Mapping the block-matrix notation back to the z-domain, we see that the vector CMA cost function for a meso-Gaussian input alphabet is minimized if and only $\mathbf{F}(z)$ is a energy-preserving filter. ❑

APPENDIX 5.6

PROOF OF THEOREM 5-5

In Appendix 5.1, we showed that minimizing the vector CMA cost function is equivalent to maximizing (5-69). Notice that maximizing γ is equivalent to simultaneously maximizing $\mathbf{1}_n{}^T \boldsymbol{q}_j$ and minimizing $[1 + (\kappa - 2)\sigma_{jj}]\ \forall\ j$. Clearly, $\boldsymbol{q}_1 = \frac{1}{\sqrt{n}}\mathbf{1}_n$ maximizes the numerator, and since the columns of $\mathbf{Q}$ are orthogonal, (5-69) reduces to:

$$\gamma = \frac{n}{1 + (\kappa - 2)\sigma_{11}}. \tag{5-106}$$

For a super-Gaussian input alphabet ($\kappa > 2$), the denominator is minimized by choosing σ_{11} as small as possible; the smallest eigenvalue corresponding to this eigenvector occurs when $\sigma_{11} = \frac{1}{M + N + 1}$. Thus, γ is maximized when

$$\mathbf{B}^T\mathbf{B}\mathbf{1}_n = \frac{1}{M + N + 1}\mathbf{1}_n. \tag{5-107}$$

The question remains: for what values of $\mathbf{V}$ does (5-107) hold?

Multiplying (5-107) on the right-hand side by $1_n{}^T$, we find that

$$\left\| \mathbf{B}\mathbf{1}_n \right\|^2 = \frac{n}{M+N+1} \Leftrightarrow \sum_{i=1}^{T} \left| \sum_{j=1}^{n} |v_{ij}|^2 \right|^2 = \frac{n}{M+N+1}, \tag{5-108}$$

where $T = (M+N+1)n$. Recall that all of the components of $\mathbf{B}$ must sum to n:

$$\sum_{i=1}^{T} \sum_{j=1}^{n} |v_{ij}|^2 = n. \tag{5-109}$$

Subtracting (5-109) from (5-108), and rearranging some of the terms, we obtain:

$$\sum_{i=1}^{T} \left[\sum_{j=1}^{n} |v_{ij}|^2 \left((M+N+1) \sum_{j=1}^{n} |v_{ij}|^2 - 1 \right) \right] = 0, \tag{5-110}$$

which holds if and only if the i-th row of $\mathbf{B}$ sums to zero or the i-th row sums to $\frac{1}{M+N+1}$. From (5-109), we deduce that all of the rows of $\mathbf{B}$ must have length $\frac{1}{M+N+1}$. Thus, the matrix $\mathbf{V}$, which satisfies (5-110) and equivalently (5-107), is given by:

$$\mathbf{V} = \frac{1}{\sqrt{M+N+1}} \begin{bmatrix} \mathbf{Q}_0 \\ \mathbf{Q}_2 \\ \vdots \\ \mathbf{Q}_{M+N} \end{bmatrix}, \tag{5-111}$$

where $\mathbf{Q}_i$ is an $n \times n$ unitary matrix. Substituting (5-111) into (5-66), we find that the optimal $\boldsymbol{d}$ is given by $\boldsymbol{d} = \alpha \mathbf{1}_n$, or equivalent $\mathbf{D} = \alpha \mathbf{I}$, where

$$\alpha = = \frac{M_v}{n + 1 + \frac{(\kappa - 2)}{M + N + 1}}. \tag{5-112}$$

This inflection point is a local minima because the Hessian of (5-65) with respect to $\boldsymbol{d}$, which is given by

$$\nabla_{dd} J_v = 2\mathbf{R}_{ww}, \tag{5-113}$$

is positive definite for a super-Gaussian input alphabet (see Lemma 5-2).

Using the fact that $\mathbf{F}^*\mathbf{F} = \mathbf{V}\mathbf{D}\mathbf{V}^*$, we find that the vector CMA cost function for a super-Gaussian input alphabet is minimized if and only if

$$\mathbf{F} = \frac{1}{\sqrt{M + N + 1}} \left(\frac{M_v}{n + 1 + \frac{(\kappa - 2)}{M + N + 1}} \right) \left[\mathbf{U}_0 \ \mathbf{U}_1 \cdots \mathbf{U}_{M+N} \right], \tag{5-114}$$

where $\mathbf{U}_i$ is an arbitrary $n \times n$ unitary matrix. Mapping the block-matrix notation back to the z-domain, (5-114) can be written as:

$$\mathbf{F}(z) = \frac{1}{\sqrt{M + N + 1}} \left(\frac{M_v}{n + 1 + \frac{(\kappa - 2)}{M + N + 1}} \right) (\mathbf{U}_0 + \mathbf{U}_1 z^{-1} + \ldots + \mathbf{U}_{M+N} z^{-M-N}). \tag{5-115}$$

This transfer function, which has maximal multiuser and intersymbol interference, minimizes the vector CMA cost function for a super-Gaussian input alphabet. ❑

APPENDIX 5.7

PROOF OF THEOREM 5-6

We recall from (5-39) that the noiseless vector CMA cost function with Gram-Schmidt constraint is given by:

$$J_{GS} = J_v + \sum_{l=-M-N}^{M+N} \left\| E[\boldsymbol{y}_k \boldsymbol{y}_{k-l}^*] - m_2 \mathbf{I}\delta_l \right\|_F^2, \tag{5-116}$$

where $\boldsymbol{y}_k = \mathbf{F}\mathbf{X}_k$. Using the fact that $E[\boldsymbol{y}_k \boldsymbol{y}_{k-l}^*] = m_2 \mathbf{F}\mathbf{J}_l\mathbf{F}^*$, where $\mathbf{J}_l$ is a block-diagonal matrix with a block-identity matrix on the l-th diagonal, and that $\|\mathbf{F}\mathbf{J}_l\mathbf{F}^*\|_F^2 = \|\mathbf{F}\mathbf{J}_{-l}\mathbf{F}^*\|_F^2$, we can express (5-116) as follows:

$$J_{GS} = J_v + (m_2)^2 \|\mathbf{F}\mathbf{F}^* - \mathbf{I}\|_F^2 + 2(m_2)^2 \sum_{l=1}^{M+N} \|\mathbf{F}\mathbf{J}_l\mathbf{F}^*\|_F^2. \tag{5-117}$$

Since $rank(\mathbf{F}) \le n$, the matrix $\mathbf{F}$ has a unique truncated singular-value decomposition: $\mathbf{F} = \mathbf{U}\mathbf{D}^{1/2}\mathbf{V}^*$, where $\mathbf{U}$ is an $n \times n$ unitary matrix, $\mathbf{D}$ is an $n \times n$ non-negative diagonal matrix, and $\mathbf{V}$ is an $(M+N+1)n \times n$ truncated unitary matrix. In Appendix 5.1, we showed that the first term of (5-117) can be written as follows:

$$J_v(\mathbf{V}, \boldsymbol{d}) = \boldsymbol{d}^T \mathbf{R}_{ww} \boldsymbol{d} - 2 m_2 M_v \boldsymbol{d}^T \mathbf{1}_n + {M_v}^2, \tag{5-118}$$

where $\boldsymbol{d}$ is an $n \times 1$ vector composed of the diagonal elements of $\mathbf{D}$ and where $\mathbf{R}_{ww} = E[\boldsymbol{w}\boldsymbol{w}^T]$. The i-th component of the $n \times 1$ vector $\boldsymbol{w}$ is given by:

$$w_i = |\, \boldsymbol{v}_i^* \mathbf{X} \,|^2, \tag{5-119}$$

where $\boldsymbol{v}_i$ is the i-th column of $\mathbf{V}$.

We can express the second term of (5-117) in terms of the vector $\boldsymbol{d}$ as follows:

$$\|\mathbf{F}\mathbf{F}^* - \mathbf{I}\|_F^2 = \|\mathbf{U}\mathbf{D}\mathbf{U}^* - \mathbf{I}\|_F^2 = \|\mathbf{D} - \mathbf{I}\|_F^2 = \boldsymbol{d}^T \boldsymbol{d} - 2\boldsymbol{d}^T \mathbf{1}_n + n, \tag{5-120}$$

where the second equality is due to the fact that the Frobenius norm is invariant to a unitary transformation. Let $\mathbf{W}_l$ denote an $n \times n$ matrix whose (i, j)-th component is given by:

$$[\mathbf{W}_l]_{ij} = |\, [\mathbf{V}^* \mathbf{J}_l \mathbf{V}]_{ij} \,|^2. \tag{5-121}$$

Using this definition, the third term of (5-117) can be written compactly as:

$$\|\mathbf{F}\mathbf{J}_l\mathbf{F}^*\|_F^2 = \|\mathbf{U}\mathbf{D}^{1/2}\mathbf{V}^*\mathbf{J}_l\mathbf{V}\mathbf{D}^{1/2}\mathbf{U}^*\|_F^2 = \|\mathbf{D}^{1/2}\mathbf{V}^*\mathbf{J}_l\mathbf{V}\mathbf{D}^{1/2}\|_F^2 = \boldsymbol{d}^T \mathbf{W}_l \boldsymbol{d}. \tag{5-122}$$

Again, the second equality is due to the fact that the Frobenius norm is invariant to a unitary transformation. Now, we can rewrite (5-117) as follows:

$$J_{GS} = \boldsymbol{d}^T\mathbf{R}_{ww}\boldsymbol{d} - 2m_2M_v\boldsymbol{d}^T\mathbf{1}_n + {M_v}^2 + (m_2)^2(\boldsymbol{d}^T\boldsymbol{d} - 2\boldsymbol{d}^T\mathbf{1}_n + n)$$

$$+ 2(m_2)^2\sum_{l=1}^{M+N}(\boldsymbol{d}^T\mathbf{W}_l\boldsymbol{d}). \tag{5-123}$$

Observe that this equation is completely parameterized by $\mathbf{V}$ and $\boldsymbol{d}$. Since these two variables are independent, the local minima can be determined by first minimizing the cost function with respect to $\boldsymbol{d}$ and then with respect to $\mathbf{V}$. The gradient of (5-123) with respect to $\boldsymbol{d}$ is given by:

$$\nabla_d J_{GS} = \mathbf{R}_{ww}\boldsymbol{d} - 2m_2M_v\mathbf{1}_n + 2(m_2)^2(\boldsymbol{d} - \mathbf{1}_n) + 2(m_2)^2\mathbf{A}\boldsymbol{d}, \tag{5-124}$$

where

$$\mathbf{A} = \sum_{l=1}^{M+N}\mathbf{W}_l + \mathbf{W}_l{}^T, \tag{5-125}$$

is a real non-negative symmetric matrix.

The inflection points of the cost function occur when the gradient is equal to the zero vector, or equivalently when $\boldsymbol{d}$ satisfies the following equation:

$$[\mathbf{R}_{ww} + (m_2)^2\mathbf{I} + (m_2)^2\mathbf{A}]\boldsymbol{d} = [m_2M_v + (m_2)^2]\mathbf{1}_n. \tag{5-126}$$

Substituting (5-126) into (5-123), we find that the vector CMA cost function with GSC reduces to:

$$J_{GS} = [M_v^2 + n(m_2)^2 - (m_2 M_v + (m_2)^2)\boldsymbol{d}^T\mathbf{1}_n] + (m_2)^2\boldsymbol{d}^T\left[\sum_{l=1}^{M+N}(\mathbf{W}_l - \mathbf{W}_l^T)\right]\boldsymbol{d}. \quad (5\text{-}127)$$

Since $\sum_{l=1}^{M+N}(\mathbf{W}_l - \mathbf{W}_l^T)$ is a real skew-symmetric matrix, the last term in (5-127) is identically zero for all $\boldsymbol{d}$. Therefore, minimizing the vector CMA cost function with GSC is equivalent to maximizing $\boldsymbol{d}^T\mathbf{1}_n = \sum_{i=1}^{n} d_i = n$.

The question remains: for what values of $\mathbf{V}$ is $\boldsymbol{d}^T\mathbf{1}_n$ maximized? In Appendix 5.1, we showed that the vector CMA cost function for a non-CM sub-Gaussian input alphabet is minimized if and only if

$$\mathbf{V} = \mathbf{P}\begin{bmatrix}\mathbf{I}_{n\times n}\\ \mathbf{0}_{(T-n)\times n}\end{bmatrix}\mathbf{Q}, \quad (5\text{-}128)$$

where $\mathbf{Q}$ is an arbitrary $n \times n$ unitary matrix and $\mathbf{P}$ is an $(M+N+1)n \times (M+N+1)n$ real permutation matrix. Substituting (5-128) into (5-126) and multiplying (5-126) on the right-hand side by $\mathbf{1}_n^T$, we find that (5-126) reduces to:

$$(M_{\mathrm{v}} + 1)\sum_{i=1}^{n} d_i + \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}d_j = (M_{\mathrm{v}} + 1)n, \quad (5\text{-}129)$$

where a_{ij} is the (i, j)-th component of $\mathbf{A}$ and where we have used the fact that, for the $\mathbf{V}$ specified by (5-128), the columns of $\mathbf{R}_{ww}$ sum to M_v. Rearranging (5-129), we find that

$$(M_{\mathrm{v}} + 1)\left[n - \sum_{i=1}^{n} d_i \right] = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} d_j \geq 0, \tag{5-130}$$

where the right-hand side is non-negative because both $\mathbf{A}$ and $\boldsymbol{d}$ are non-negative. Since $\boldsymbol{d} \neq \mathbf{0}$, $\sum_{i=1}^{n} d_i$ is maximized when $\mathbf{A} = \mathbf{0}$.

If we view $\mathbf{V}$ in term of its taps, *i.e.*, $\mathbf{V}^T = [\mathbf{V}_0{}^T \, \mathbf{V}_1{}^T \, \ldots \, \mathbf{V}_{M+N}{}^T]$, where $\mathbf{V}_i$ is an $n \times n$ matrix, then the condition that $\mathbf{A} = \mathbf{0}$ implies that

$$\sum_{i=0}^{M+N-1} \sum_{j=i+1}^{M+N} \mathbf{V}_i^* \mathbf{V}_j = \mathbf{0}. \tag{5-131}$$

This condition in turn implies that there cannot be any correlation between the users in the detector output. Substituting, $\mathbf{A} = \mathbf{0}$ in (5-126), we find that the optimal $\boldsymbol{d}$ is given by $\boldsymbol{d} = \mathbf{1}_n$, or equivalent $\mathbf{D} = \mathbf{I}$. The Hessian of (5-124) with respect to $\boldsymbol{d}$ is given by

$$\nabla_{dd} J_v = 2\mathbf{R}_{ww} + 2(m_2)^2 \mathbf{I}. \tag{5-132}$$

This matrix is clearly positive definite, because the identity matrix is positive definite and the sum of a positive-definite matrix and a positive-semidefinite matrix is always positive definite. Therefore, this inflection point is a local minima for all sub-Gaussian input alphabets, including the CM input alphabet.

Using the fact that $\mathbf{F}^*\mathbf{F} = \mathbf{V}\mathbf{D}\mathbf{V}^*$, we find that the vector CMA cost function with GSC for a sub-Gaussian input alphabet is minimized if and only if

$$\mathbf{F} = [\mathbf{F}_0\ \mathbf{F}_1\ \ldots\ \mathbf{F}_{M+N}] = \mathbf{U}\left[\mathbf{I}_{n\times n}\ \ \mathbf{0}_{n\times(T-n)}\right]\mathbf{P}^T, \tag{5-133}$$

where $\mathbf{U}$ is an arbitrary $n \times n$ unitary matrix, $\mathbf{P}$ is an $(M+N+1)n \times (M+N+1)n$ real permutation matrix, and $\mathbf{F}$ satisfies the following constraint:

$$\sum_{i=0}^{M+N-1}\ \sum_{j=i+1}^{M+N} \mathbf{F}_i^{*}\mathbf{F}_j = \mathbf{0}. \tag{5-134}$$

Finally, mapping the block-matrix notation back to the z-domain, we see that the vector CMA cost function with GSC for a sub-Gaussian input alphabet is minimized if and only if

$$\mathbf{F}(z) = \mathbf{U}\mathbf{D}(z)\mathbf{P}, \tag{5-135}$$

where $\mathbf{U}$ is an arbitrary $n \times n$ unitary matrix, $\mathbf{D}(z)$ is an $n \times n$ diagonal matrix with elements of the form z^{-D} where D is an arbitrary delay value, and $\mathbf{P}$ is an $n \times n$ real permutation matrix. ❑

CHAPTER 6

CONCLUSIONS AND FUTURE RESEARCH

6.1 CONCLUSIONS

We have proposed several new algorithms for blind multiuser detection. The inspiration for these algorithms is drawn from two time-tested single-user blind algorithms: the constant-modulus algorithm (CMA) and the decision-directed phase-locked loop (PLL). The proposed detectors have good performance, low complexity, and fast convergence. We have demonstrated the effectiveness of these algorithms in a wide variety of contexts including a uniform linear array application and a code-division multiple-access system.

In Chapter 3, we have reviewed the basic structure of a first-order and a second-order phase-locked loop. We have analyzed the dynamics, both theoretically and experimentally, of these PLLs. We have shown that there exists a minimum step size that guarantees convergence of the PLL within a finite number of symbols. Unfortunately, the structure of the conventional PLL does not extend to multiple dimensions, so we have manipulated the update equations to develop an alternative model for the PLL, which is shown in Fig. 3-7. Using this alternative model, we have proposed the multidimensional phase-locked loop

(MPLL), which is illustrated in Fig. 3-9. The MPLL is a decision-directed algorithm that exploits the discrete nature of digital communication signals in order to blindly estimate and resolve a unitary ambiguity. We have experimentally analyzed the dynamics of both a first-order and second-order MPLL. There exists a minimum step size, as was the case for the PLL, that guarantees convergence of the MPLL within a finite number of symbols. Finally, when compared to JADE and EASI, for various unitary channels, we have shown that the MPLL offers fast convergence, excellent steady-state performance, and low complexity.

In Chapter 4, we have proposed the vector CMA cost function, which is based on a unique generalization of the CMA cost function. For this cost function, we have determined the local minima both in the absence and in the presence of noise. We have shown that in the absence of noise, this cost function is minimized only by unitary matrices when the input alphabet is non-CM (Theorem 4-3), and by both unitary and non-unitary matrices when the input alphabet is CM (Theorem 4-4). A consequence of Theorem 4-3 is that the vector CMA detector is compatible with highly shaped input alphabets and can, therefore, be used on system that approach Shannon capacity. In the presence of noise, we have demonstrated that the vector CMA has near-MMSE like performance. For a CM input alphabet, we have proposed the vector CMA cost function with Gram-Schmidt constraint (GSC). This cost function is minimized only by unitary matrices for all input alphabets. Unfortunately, the elimination of the undesirable minima comes at the expense of higher complexity. We have also presented detailed implementations of both of these detectors. Finally, we have compared the performance and complexity of both the vector

CMA and vector CMA with GSC detectors to other known CMA-based blind multiuser detectors for a multisensor receiver and a synchronous CDMA application.

In Chapter 5, we have extended the vector CMA cost function to channels with memory. In the absence of noise, we have shown that the vector CMA cost function is minimized by both unitary and non-unitary matrices when the input alphabet is sub-Gaussian. For non-sub-Gaussian input alphabets, this cost function proves to be useless, as expected. In order to eliminate the undesirable local minima, we have also extended the vector CMA cost function with GSC to channels with memory. In the absence of noise, this cost function is minimized only by unitary matrices for a sub-Gaussian input alphabet. As with memoryless channels, the elimination of the undesirable local minima comes at the expense of increased computational complexity. We have included a detailed implementation of a reduced-complexity vector CMA with GSC detector. We have also shown that in the presence of noise, the performance of optimal vector with GSC detector is nearly identical to that of the optimal MMSE detector. Finally, using a multisensor receiver with multipath and an asynchronous CDMA application, we have compared the performance of the vector CMA with GSC detector to that of a forward-backward LP and decorrelation CMA detector.

6.2 Future Research

6.2.1 MPLL Convergence

The computer simulations in Chapter 3 suggest that the first-order MPLL does indeed converge to the desired stable point for an appropriately chosen step size. We currently lack a rigorous mathematical proof of convergence for this first-order MPLL, which may

have provided us insight into the nonlinear behavior of higher order systems, such as the second-order MPLL. It may be possible to derive it by using a multidimensional generalization of the Fokker-Planck equation, which has been used to provide convergence for a first-order PLL [32].

6.2.2 Fading Channels

Throughout this dissertation, we assumed that the channel transfer function is stationary. We have, in fact, exploited this property in the implementation of the vector CMA with GSC detector. While this assumption is valid in some cases, it does not always reflect all real-world applications. In fact, many applications are described by either fading channels or rapidly time-varying channels. For these channels, there is a need to speed up the convergence of the proposed detectors. All of the adaptive algorithm described in this dissertation are designed in the spirit of the LMS algorithm; the instantaneous estimates of the gradient are used in place of their true values. One possible approach to increase the rate of convergence is to design algorithms similar to the recursive least-squares algorithm. The resulting algorithms would be relatively more complex, but the greater speed of convergence would enable them to work well on both fading channels and rapidly time-varying channels.

6.2.3 Undesirable Local Minima for FIR Linear Detectors

The fundamental work of Benveniste *et al*., Godard, Foschini, and Shalvi *et al*. on the convergence characteristics of the constant-modulus algorithm for infinite-length equalizers has recently been examined and extended to finite-length equalizers by Ding et al. [89,90] and Johnson [91]. Ding and colleagues have shown that for a QAM input alphabet

and a broad class of non-pathological (minimum-phase channels) channels, there exists undesirable local minima, which do not correspond to an open eye diagram. The CMA therefore suffers from undesirable local minima when the channel has memory and the equalizer is FIR.

An open area of research is to extend the analysis presented by Ding and his colleagues to the vector constant-modulus algorithm. These results would apply only to square multiuser channels and finite-length linear detectors. In the analysis, it may be possible to determine the class of channels for which vector CMA fails to converge to the desired local minima and also to develop techniques which can prevent misconvergence. For a tall multiuser channel, as we have shown in Chapter 5, we do not expect that there will exist undesirable local minima because it can be inverted by a FIR linear detector. This conclusion still needs to be confirmed.

REFERENCES

[1] E.A. Lee and D. G. Messerschmitt, *Digital Communication*, Second Edition, Kluwer, Boston, 1994.

[2] J. G. Proakis, *Digital Communications*, Third Edition, McGraw-Hill, New York, 1995.

[3] S. Haykin, *Communication Systems*, Third Edition, John Wiley & Sons, New York, 1994.

[4] N. Abramson, ed., *Multiple Access Communications*: *Foundations for Emerging Technologies*, IEEE Press, New York, 1993.

[5] G. L. Stüber, *Principles of Mobile Communication*, Kluwer, Boston, 1996.

[6] A. J. Viterbi, *CDMA*: *Principles of Spread Spectrum Communication*, Addison-Wesley, Reading, MA, 1995.

[7] P. A. Voois and J. M. Cioffi, "Multichannel signal processing for multiple-head digital magnetic recording," *IEEE Transactions on Magnetics*, vol. 30, no. 6, pp. 5100-5114, November 1994.

[8] P. A. Voois and J. M. Cioffi, "Multichannel digital magnetic recording," *1992 International Conference on Communications*, vol. 1, pp. 125-130, Chicago, IL, June 1992.

[9] J. Salz, "Digital transmission over cross-coupled linear channels," *AT&T Technical Journal*, vol. 64, no. 6, pp. 1147-1158, July-August 1985.

[10] M. L. Honig, P. Crespo, and K. Steiglitz, "Suppression of near- and far-end crosstalk by linear pre- and post-filter," *IEEE Journal on Selected Areas in Communications*, vol. 10, no. 3, pp. 614-629, April 1992.

[11] M. Kavehrad and J. Salz, “Cross-polarization cancellation and equalization in digital transmission over dually polarized multipath fading channels,” *Bell System Technical Journal*, vol. 64, no. 10, pp. 2211-45, December 1985.

[12] S. Verdú, *Multiuser Detection*, Cambridge University Press, Cambridge, UK, 1998.

[13] S. Verdú, “Recent progress in multi-user detection,” in *Advances in Communications and Signal Progressing*, W. A. Porter and S. C. Kak, Eds. Springer-Verlag, New York (1989).

[14] W. Van Etten, “Maximum likelihood receiver for multiple channel transmission Systems,” *IEEE Transactions on Communications*, vol. COM-24, no. 2, pp. 276-283, February 1976.

[15] W. Van Etten, “An optimum linear receiver for multiple channel digital transmission systems,” *IEEE Transactions on Communications*, vol. COM-23, no. 8, pp. 828-834, August 1975.

[16] R. Lupas and S. Verdú, “Linear multiuser detectors for synchronous code-division multiple-access channels,” *IEEE Transactions on Information Theory*, vol. 35, no. 1, pp. 123-136, January 1989.

[17] N. Amitay and J. Salz, “Linear equalization theory in digital data transmission over dually polarized fading radio channels,” *AT&T Bell Laboratories Technical Journal*, vol. 63, no. 10, pp. 2215-2259, December 1984.

[18] A. Duel-Hallen, “Equalizers for multiple input/multiple output channels and PAM systems with cyclostationary input sequences,” *IEEE Journal on Selected Areas in Communications*, vol. 10, no. 3, pp. 630-639, April 1992.

[19] K. D. Senne, “Adaptive linear discrete-time estimation,” *Technical Report No*. 6778-5, Systems Theory Laboratory, Stanford University, June 1968.

[20] A. Batra and J. R. Barry, “Blind cancellation of co-channel interference,” *IEEE Global Telecommunications Conference*, Singapore, vol. 1, pp. 157-162, November 1995.

[21] M. Abdulrahman, A. Sheik, and D. Falconer, “Decision feedback equalization for CDMA in indoor wireless communication,” *IEEE Journal on Selected Areas in Communications*, vol. 12, no. 4, pp. 698-706, May 1994.

[22] A. Duel-Hallen, “Decorrelating decision-feedback multiuser detector for synchronous CDMA,” *IEEE Transactions on Communications*, vol. 41, no. 2, pp. 285-290, February 1993.

[23] A. Duel-Hallen, “A family of multiuser decision-feedback detectors for asynchronous code-division multiple-access channels,” *IEEE Transactions on Communications*, vol. 43, no. 2-4, pp. 421-434, February-April 1995.

[24] J. Yang and S. Roy, “Joint transmitter-receiver optimization for multi-input multi-output systems with decision feedback,” *IEEE Transactions on Information Theory*, vol. 40, no. 5, pp. 1334-1347, September 1994.

[25] S. Haykin, *Blind Deconvolution*, Prentice Hall, 1994.

[26] J. R. Treichler and B. G. Agee, “A new approach to multipath correction of constant modulus signals,” I*EEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-31, no. 2, pp. 459-472, April 1983.

[27] F. M. Gardner, *Phaselock Techniques*, Second Edition, John Wiley & Sons, New York, 1979.

[28] A. J. Viterbi, *Principles of Coherent Communication*, McGraw-Hill, New York, 1996.

[29] U. Mengali and A. N. D’Andrea, *Synchronization Techniques for Digital Receivers*, Plenum Press, New York, 1997.

[30] J. B. Encinas, *Phase Locked Loops*, Chapman & Hall, London, 1993.

[31] A. Blanchard, *Phase-Locked Loops*: *Application to Coherent Receiver Design*, John Wiley & Sons, New York, 1976.

[32] H. Meyr and G. Ascheid, *Synchronization in Digital Communications*, John Wiley & Sons, New York, 1990.

[33] N. Delfosse and P. Loubaton, “Adaptive separation of independent sources: a deflation approach,” *Signal Processing*, vol.45, no.1, pp.59-83, July 1995.

[34] A. Gorokhov, P. Loubaton, and E. Moulines, “Second order blind equalization in multiple input multiple output FIR systems: a weighted least squares approach”, *1996 International Conference on Acoustics, Speech, and Signal Processing*, Atlanta, GA, vol. 5, pp. 2415-2418, May 1996.

[35] A. Gorokhov and P. Loubaton, “Blind identification of MIMO-FIR systems: a generalized linear prediction approach,” *Signal Processing*, vol. 73, no. 1-2, pp. 105-124, 1998.

[36] P. Comon, “Contrasts for multichannel blind deconvolution”, *IEEE Signal Processing Letters*, vol. 3, no. 7, pp. 209-211, July 1996.

[37] E. Moreau and J.-C. Pesquet, “Generalized contrasts for multichannel blind deconvolution of linear systems,” *IEEE Signal Processing Letters*, vol. 4, no. 6, pp. 182-193, June 1997.

[38] 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. 40, no. 2, pp. 340-349, March 1994.

[39] D. T. M. Slock, “Blind fractionally-spaced equalization, perfect-reconstruction filter banks and multichannel linear prediction”, *1994 International Conference on Acoustics, Speech, and Signal Processing*, Adelaide, Australia, vol. 4, pp. 585-588, April 1994.

[40] D. Slock and C. Papadias, “Blind fractionally spaced equalization based on cyclostationarity,” *Vehicular Technology Conference*, pp. 1286-1290, Stockholm, 1994.

[41] D. Slock and C. Papadias, “Further results on blind identification and equalization of multiple FIR channels,” *1995 International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 1964-1967, Detroit, 1995.

[42] K. Abed Meraim et al., “Prediction error methods for time-domain blind identification of multichannel FIR filters”, *1995 International Conference on Acoustics, Speech, and Signal Processing*, Detroit, MI, vol. 3, pp. 1968-1971, May 1995.

[43] S. Icart and R. Gautier, “Blind separation of convolutive mixtures using second and fourth order moments”, *1996 International Conference on Acoustics, Speech, and Signal Processing*, Atlanta, GA, vol. 5, pp. 3018-3021, May 1996.

[44] M. Nájar *et al*., “Blind wideband source separation”, *1994 International Conference on Acoustics, Speech, and Signal Processing*, Adelaide, Australia, vol. 4, pp. 65-68, April 1994.

[45] Z. Ding, “A blind channel identification algorithm based on matrix outer-product”, *1996 International Conference on Communications*, Dallas, TX, vol. 2, pp. 852-856, June 1996.

[46] R. T. Causey and J. R. Barry, “Blind multiuser detection using linear prediction,” *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 9, pp. 1702-1710, December 1998.

[47] J. F. Cardoso and A. Souloumiac, “Blind beamforming for non-Gaussian signals”, *IEE Proceedings F* (*Radar and Signal Processing*), vol. 140, no. 6, pp. 362-370, December 1993.

[48] J. F. Cardoso, “Eigen-structure of the fourth-order cumulant tensor with application to the blind source separation problem”, *1990 International Conference on Acoustics, Speech, and Signal Processing*, Albuquerque, NM, vol. 5, pp. 2655-2658, April 1990.

[49] J. F. Cardoso and P. Comon, “Tensor-based independent component analysis,” in *Signal Processing V*: *Theory and Applications*, vol. 1, pp. 673-6, September 1990.

[50] J. F. Cardoso and B. H. Laheld, “Equivariant adaptive source separation”, *IEEE Transactions on Signal Processing*, vol. 44, no.12, pp. 3017-3030, December 1996.

[51] J. R. Barry and A. Batra, “A multidimensional phase-locked loop for blind equalization of multi-input multi-output channels,” *International Conference on Communications*, Dallas, TX, vol. 3, pp. 1307-1312, June 1996.

[52] A. Batra and J. R. Barry, “Blind unitary source separation using a multidimensional phase-locked loop,” *First IEEE Signal Processing Workshop on Signal Processing Advances in Wireless Communications*, Paris, pp. 61-64, April 1997.

[53] J. R. Barry and A. Batra, “Co-channel demodulation using multi-input multi-output equalization,” *CRASP Annual Report*, July 30, 1996.

[54] Richard T. Causey, “Blind multiuser detection based on second-order statistics,” Ph.D. Thesis, Georgia Institute of Technology, July 1999.

[55] A. V. Oppenhiem and R. W. Shafer, *Discrete-Time Signal Processing*, Prentice Hall, Englewood Cliffs, NJ, 1989.

[56] T. Kaliath, *Linear Systems*, Prentice Hall, Englewood Cliffs, NJ, 1980.

[57] Y. Li and K. J. R. Liu, “Adaptive blind source separation and equalization for multiple-i input/multiple-output systems,” *IEEE Transactions on Information Theory*, vol. 44, no. 7, pp. 2864-76, November 1998.

[58] G. D. Forney, Jr., “Minimal bases of rational vector spaces, with applications to multivariable linear systems,” *SIAM Journal on Control*, vol. 13, pp. 493-520, May 1975.

[59] G. D. Forney, Jr., “Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference,” *IEEE Transactions on Information Theory*, vol. 18, no. 3, pp. 363-377, May 1972.

[60] G. D. Forney, Jr., “The Viterbi algorithm,” *Proceedings of IEEE*, vol. 61, no. 3, pp. 268-278, March 1973.

[61] J. R. Barry and A. Batra, “Co-channel demodulation using multi-input multi-output equalization,” *CRASP Annual Report*, August 1, 1995.

[62] N. Seshadri, “Joint data and channel estimation using fast blind trellis search techniques,” *Proceedings of the IEEE Global Telecommunications Conference*, pp. 1659-1663, San Diego, December 1990.

[63] M. Ghosh and C. L.Weber, “Maximum likelihood blind equalization,” *Proceedings of the SPIE Conference*, vol. 1565, pp. 188-195, Bellingham, WA, 1991.

[64] O. Macchi and A. Hachicha, “Self-adaptive equalization based on a prediction principle,” *IEEE Global Telecommunications Conference*, Houston, TX, vol.3, pp. 1641-1645, December 1986.

[65] L. Tong and D. Liu, “Blind predictive decision-feedback equalization via the constant modulus algorithm,” *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Munich, Germany, vol. 5, pp. 3901-4, April 1997.

[66] C. F. Wong and T. L. Fine, “Adaptive blind equalization using artificial neural networks,” *Proceedings of the International Conference on Neural Networks*, Washington, DC, vol. 4, pp. 1974-1979, June 1996.

[67] T. Chen and R. Chen, “Neural network approach to blind identification of stochastic and deterministic signals,” *Conference Record of the Twenty-Eighth Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, vol. 2, pp. 892-896, November 1994.

[68] C. L. Nikias and J. G. Proakis, “Blind equalization algorithms,” Final Report, Communications and Digital Signal Processing (CDSP), Center for Research and Graduate Studies, Northeastern University, Boston, MA, May 1990.

[69] D. Hatzinakos and C. L. Nikias, “Blind equalization using a tricepstrum based algorithm,” *IEEE Transactions on Communications*, vol. 39, no. 5, pp. 669-682, May 1991.

[70] A. G. Bessios and C. L. Nikias, “POTEA: The power cepstrum and tricoherence equalization algorithm,” *IEEE Transactions on Communications*, vol. 43, no. 11, pp. 2667-2671, November 1995.

[71] D. H. Brooks and C. L. Nikias, “Multichannel adaptive blind deconvolution using the complex cepstrum of higher order cross-spectra,” *IEEE Transactions on Signal Processing*, vol. 41, no. 9, pp. 2928-2934, September 1993.

[72] F.-C. Zheng, S. McLaughlin, and B. Mulgrew, “Blind equalization of multilevel PAM data for nonminimum phase channels via second- and fourth-Order cumulants,” *Signal Processing*, vol. 31, no. 3, pp. 313-327, April 1993.

[73] B. Porat and B. Friedlander, “Blind adaptive equalization of digital communication channels using high-order moments,” *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pp. 1372-1373, Glasgow, Scotland, 1989.

[74] B. Porat and B. Friedlander, “Blind equalization of digital communication channels using high-order moments,” *IEEE Transactions on Signal Processing*, vol. 39, no. 2, pp. 522-526, February 1991.

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

[76] S. Haykin, *Adaptive Filter Theory*, Second Edition, Prentice Hall, Englewood Cliffs, NJ, 1991.

[77] J. J. Bussgang, “Crosscorrelation functions of amplitude-distorted Gaussian signals,” *MIT Technical Report No*. 216, March 1952.

[78] S. Bellini, “Bussgang techniques for blind equalization,” *IEEE Global Telecommunications Conference*, Houston, vol.3, pp. 1634-1640, December 1986.

[79] S. Bellini, “Bussgang techniques for blind deconvolution and equalization”, in *Blind Deconvolution*, S. Haykin, Ed., Prentice Hall (1994), pp. 8-59.

[80] 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.

[81] Y. Sato et al., “Blind suppression of time dependency and its extension to multi-dimensional equalization,” *IEEE Global Telecommunications Conference*, Houston, TX, vol.3, pp. 1652-1656, December 1986.

[82] A. Benveniste et al., “Robust identification of a nonminimum phase system: blind adjustment of a linear equalizer in data communications,” *IEEE Transactions on Automatic Control*, vol. AC-25, no. 3, pp. 385-399, June 1980.

[83] A. Benveniste and M. Goursat, “Blind equalizers,” *IEEE Transactions on Communications*, vol. COM-32, no. 8, pp 871-883, August 1984.

[84] 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, no. 9, pp. 877-887, September 1987.

[85] 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.

[86] G. J. Foschini, “Equalizing without altering or detecting data,” *AT&T Technical Journal*, vol. 64, no. 8, pp. 1885-1911, October 1985.

[87] O. Shalvi and E. Weinstein, “New criteria for blind deconvolution of nonminimum phase systems (channels),” *IEEE Transactions on Information Theory*, vol. 36, pp. 312-321, March 1990.

[88] J. R. Treichler et al. “Observed misconvergence in the constant modulus adaptive algorithm,” *Conference Record of the Twenty-Fifth Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, vol. 2, pp. 663-667, November, 1991.

[89] Z. Ding *et al*., “Ill-convergence of Godard blind equalizers in data communication systems,” *IEEE Transactions on Communications*, vol. 39, no. 9, September 1991.

[90] Z. Ding et al., “On the (non)existence of undesirable equilibria of Godard blind equalizers,” *IEEE Transactions on Signal Processing*, pp. 2425-2432, October 1992.

[91] C. Johnson, “Admissibility in blind adaptive equalization,” *IEEE Control Systems Magazine*, 11:3-15, January 1991.

[92] Y. Chen, C.L. Nikias, and J. G. Proakis, “CRINMO: criterion with memory nonlinearity for blind equalization,” *Conference Record of the Twenty-Fifth Asilomar Conference on Signals, Systems, and Computers*, November 4-6, 1991.

[93] P. Tsakalides and C. L. Nikias, “A new criterion for blind deconvolution of colored input signals,” *Conference Record of the Twenty-Seventh Asilomar Conference on Signals, Systems, and Computers*, pp. 746-750, November 4-6, 1993.

[94] J. F. Cardoso, “Source separation using higher order moments”, *1989 International Conference on Acoustics, Speech, and Signal Processing*, Glasgow, UK, vol. 4, pp. 2109-2112, May 1989.

[95] P. Comon, “Independent component analysis, a new concept?”, *Signal Processing*, vol. 36, no. 3, pp. 287-314, 1994.

[96] D. Donoho, “On minimum entropy deconvolution,” in *Applied Time-Series Analysis II*, D. Findley, ed., Academic Press, pp. 565-608, 1981.

[97] G. B. Giannakis et al., “Cumulant based information of multichannel moving-average models”, *IEEE Transactions on Automatic Control*, vol.34, no.7, pp. 783-787, July 1989.

[98] A. Swami et al., “Multichannel ARMA processes”, *IEEE Transactions on Signal Processing*, vol. 42, no. 4, pp. 898-913, April 1994.

[99] J. T. Tugnait, “On blind MIMO channel estimation and blind signal separation in unknown additive noise,” *First IEEE Signal Processing Workshop on Signal Processing Advances in Wireless Communications*, Paris, pp. 53-60, April 1997.

[100] L. Castedo, C. J. Escudero, and A. Dapena, “A blind signal separation Method for multiuser communications,” *IEEE Transactions on Signal Processing*, vol. 45, no. 5, pp. 1343-1348, May 1997.

[101] C. B. Papadias and A. J. Paulraj, “A constant modulus algorithm for multiuser signal separation in presence of delay spread using antenna arrays,” *IEEE Signal Processing Letters*, vol. 4, no. 6, pp. 178-181, June 1997.

[102] H. Oda and Y. Sato, “A method of multi-dimensional blind equalization,” *International Symposium on Information Theory*, San Antonio, TX, p. 327, 1993.

[103] J. R. Treichler and M. G. Larimore, “New processing techniques based on the constant modulus adaptive algorithm,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 33, no. 2, pp. 420-431, April 1985.

[104] M. K. Simon and J. G. Smith, “Carrier synchronization and detection of QASK signal sets,” *IEEE Transactions on Communications*, vol. 22, no. 2, pp. 98-106, February 1974.

[105] W. H. Press *et al*., *Numerical Recipes in C*, Cambridge University Press, New York, 1988.

[106] C. R. Johnson and R. A. Horn, *Matrix Analysis*, Cambridge University Press, 1990.

[107] T. M. Cover and Joy. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, New York, 1991.

[108] C. Escudero *et al*., “Performance analysis of a CMA-based adaptive multiuser detector,” *First IEEE Signal Processing Workshop on Signal Processing Advances in Wireless Communications*, Paris, pp.281-4, April 1997.

[109] H. H. Zeng and Lang Tong, “The MSE performance of constant modulus receivers,” *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 5, pp.3577-80, April 1997.

[110] S. Zeng *et al*., “Blind channel equalization via multiobjective optimization,” *Proceedings of 8th Workshop on Statistical Signal and Array Processing*, pp.160-3, June 1996.

[111] H.H. Zeng *et al*., “Blind equalization using the constant modulus algorithm,” *1996 Third International Conference on Signal Processing Proceedings*, vol. 1, pp.400-3, October 1996.

[112] A. R. Calderbank and L. H. Ozarow, “Nonequiprobable signaling on the Gaussian channel,” *IEEE Transactions on Information Theory*, vol. 36, no. 4, July 1990.

[113] D. D. Falconer, M. Abdulrahman, N. W. K. Lo, B. R. Petersen, and A. U. H. Sheikh, “Advances in equalization and diversity for portable wireless systems,” *Digital Signal Processing*, vol. 3, No. 3, pp. 148-162, July 1993.

[114] J. P. LeBlanc *et al*., "Fractionally-spaced constant modulus algorithm blind equalizer error surface characterization: effects of source distributions," *1996 International Conference on Acoustics, Speech, and Signal Processing*, vol. 5, pp. 2944-2947, Atlanta, May 1996.

VITA

Anuj Batra was born on April 27, 1970 in Chapel Hill, North Carolina. He received his Bachelor of Science Degree, with distinction, in Electrical Engineering from Cornell University in January 1992. While at Cornell, he was the recipient of the *John McMullen Dean's Prize* and of the *Outstanding Senior Award in Electrical Engineering*. During his senior year, he served as President of Eta Kappa Nu and Treasurer of the student chapter of IEEE. After graduation, he worked at Raytheon E-Systems where he contributed in the development of software to track mobile AMPS cellular signals. In the fall of 1992, he enrolled in the graduate school at Stanford University, where he was awarded a Graduate Fellowship. He received his Master of Science Degree in Electrical Engineering from Stanford in June 1993. In September 1993, he enrolled in the School of Electrical Engineering at the Georgia Institute of Technology to begin his doctoral program, where he was a recipient of the Georgia Tech President's Fellowship. He was awarded his Doctorate in Philosophy in April 2000. He is a member of IEEE, Eta Kappa Nu, Tau Beta Pi and Golden Key Honor Society. His current research interests include multiuser communication theory, wireless communications, equalization, and coding.