uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Jongsoo Choi

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Ph.D. (Electrical Engineering)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Recursive Filtering Approaches to Channel Equalization and Estimation in Communications

TITRE DE LA THÈSE / TITLE OF THESIS

Martin Bouchard

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

Tet Yeap

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Claude D'Amours

Richard Dansereau

Saeed Gazor

Abbas Yongacoglu

Gary W. Slater

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES /
DEAN OF THE FACULTY OF GRADUATE AND POSTDOCORAL STUDIES

Recursive Filtering Approaches to Channel Equalization and Estimation in Communications

Recursive Filtering Approaches to Channel Equalization and Estimation in Communications

Jongsoo Choi

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements for the degree of
Doctor of Philosophy
In Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa

February 2006

Your file Votre référence
ISBN: 978-0-494-15011-5
Our file Notre référence
ISBN: 978-0-494-15011-5

Abstract

Communication poses challenges in the presence of interference channel environments. In order to attain performance gains, signal processing techniques at the receiver need to detect the most likely transmitted signal based on the knowledge of the received signal, the channel state information and the statistics of noise. This thesis develops such practical schemes based on a Kalman filter framework, assuming that the channel knowledge is unknown to the receiver. The communication contexts addressed in this thesis include the equalization of intersymbol interference (ISI) channels in single-transmit single-receive (STSR) systems and the channel estimation combined with the decoding of multiple-input multiple-output (MIMO) systems in fading channels. Signal processing based on a Bayesian filtering framework, built on a state-space model of the given communication system, plays a critical role in the equalization and the estimation of the channel.

We develop adaptive channel equalization techniques utilizing recurrent neural networks and Kalman filters for ISI cancellation in STSR systems, which include nonlinear distortions, additive white Gaussian noise and additive white non-Gaussian impulsive noise. In uncoded systems, the extended Kalman filter (EKF) and the unscented Kalman filter (UKF) are used to train the recurrent neural equalizers and achieve the performance improvement in terms of bit error rate and convergence rate, with a shorter training sequence over the conventional equalization schemes.

The integration of semiblind channel estimation into a turbo receiver, the so-called turbo-BLAST[1], is developed for narrowband MIMO wireless channels. For quasi-static MIMO channels, we present iterative channel estimation schemes based on adaptive filtering algorithms such as least-mean square, recursive least-squares and the Kalman filter. The iterative strategy with adaptive filtering leads to a computationally efficient solution to iterative channel estimation, compared to the conventional snapshot approaches. For time-varying MIMO channels, we present the use of particle filtering in order to track the time variations of the channels. The improved performance by the particle filtering channel tracking is demonstrated for both Gaussian and non-Gaussian noise environments.

[1]BLAST: Bell-labs LAyered Space-Time architectures.

Acknowledgments

I am indebted to my supervisors Prof. Martin Bouchard and Prof. Tet Yeap for their support throughout the development of this thesis. Dr. Bouchard's outstanding expertise and tireless advice greatly improved the technical content and the presentation of this thesis. It is a pleasure for me to thank Dr. Yeap who considerably contributed to this thesis by means of countless constructive advice and encouragement.

I wish to thank Prof. C. D'Amours and Prof. A. Yongaçoglu at the University of Ottawa, Prof. R. Dansereau at Carleton University, and Prof. S. Gazor at Queen's University for serving as my committee and for their invaluable suggestions and corrections.

I am grateful to all my colleagues at Bell University Laboratories, Ottawa and the Signal Processing Oriented Technologies (SPOT) group at the University of Ottawa for their friendship. In particular, I want to thank Dafu Lou and Simon Robidas for many entertaining hours.

Special thanks go to the people who I met at Ottawa Korean Community Church for sharing their deep devotion to God with me over the years. Finally, I would like to thank my wife, Youmi, and my lovely kids, Chiwon and Yoona, for their unconditional understanding and constant support.

My gratitude also goes to Dr. Yeap, the Faculty of Engineering, and the Faculty of Graduate and Postdoctoral Studies for financial support.

Contents

List of Figures

List of Tables

List of Abbreviations

3G	Third Generation
4G	Fourth Generation
ANSE	Average Normalized Square Error
APP	*a posteriori* Probability
AR	AutoRegressive
AWGN	Additive White Gaussian Noise
BCJR	Bahl-Cocke-Jelinek-Raviv
BE	Bayesian Equalizer
BEM	Basis Extension Model
BER	Bit Error Rate
BLAST	Bell-labs LAyered Space-Time
BLUE	Best Linear Unbiased Estimate
BP	Back-Propagation
BPSK	Binary Phase-Shift Keying
CAI	Co-Antenna Interference
CCI	Co-Channel Interference
CDMA	Code Division Multiple Access
CIR	Channel Impulse Response
CRB	Cramér-Rao Bound
CSI	Channel State Information
DEKF	Decoupled Extended Kalman Filter
DFE	Decision-Feedback Equalizer
DFRNE	Decision-Feedback Recurrent Neural Equalizer
DSSM	Dynamic State-Space Model
DS/SS	Direct-Sequence Spread-Spectrum

EDGE	Enhanced Data rates for Global Evolution
EKF	Extended Kalman Filter
EM	Expectation-Maximization
FER	Frame Error Rate
FIR	Finite Impulse Response
FLOS	Fractional Lower-Order Statistics
FNN	Forward Neural Networks
GEKF	Global EKF
GPF	Gradient proposal Particle Filter
GSM	Global System for Mobile
GSNR	Geometric Signal-to-Noise Ratio
IDD	Iterative Detection and Decoding
ISI	Intersymbol Interference
KF	Kalman Filter
LLR	Log Likelihood Ratio
LMP	Least Mean p-norm
LMS	Least Mean Square
LS	Least Square
LTE	Linear Transversal Equalizer
MAP	Maximum *a posteriori* Probability
MC	Monte Carlo
MCMC	Markov Chain Monte Carlo
MIMO	Multi-Input Multi-Output
MISO	Multi-Input Single-Output
ML	Maximum Likelihood
MLP	MultiLayer Perceptron
MLSE	Maximum Likelihood Sequence Estimation
MMSE	Minimum Mean-Squared Error
MSE	Mean Square Error
NLMP	Normalized LMP
NN	Neural Networks
OFDM	Orthogonal Frequency Division Multiplexing
PAT	Pilot-Assisted Transmission

PSAM	Pilot Symbol Assisted Modulation
PSIC	Parallel Soft-Interference Cancelation
QoS	Quality of Service
QPSK	Quaternary Phase-Shift Keying
RBF	Radial Basis Function
RLS	Recursive Least Square
RLST	Random Layered Space-Time
RMLP	Recurrent MultiLayer Perceptron
RNN	Recurrent Neural Networks
RPP	Regular Periodic Placement
RTRL	Real-Time Recurrent Learning
SαS	Symmetric α-Stable
SIR	Sampling Importance Resampling
SIS	Sequential Importance Sampling
SISO	Soft-Input Soft-Output
SMC	Sequential Monte Carlo
SNR	Signal-to-Noise Ratio
SOS	Second-Order Statistics
ST	Space-Time
STSR	Single-Transmit Single-Receive
TDM	Time Division Multiplexing
UKF	Unscented Kalman filter
UT	Unscented Transform
VA	Viterbi Algorithm
WSS	Wide-Sense Stationary
ZF	Zero-Forcing
ZMCSCG	Zero-Mean Circularly Symmetric Complex Gaussian

List of Symbols

a	Scalar
$\mathbf{a}$	Column vector
$\mathbf{I}_n$	Identity matrix of dimension $n \times n$
$\mathbf{A}$	Matrix $\mathbf{A}$
$\mathbf{A}^{-1}$	Inverse of matrix $\mathbf{A}$
$\text{tr}(\mathbf{A})$	Trace of matrix $\mathbf{A}$
$()^*$	Complex conjugate
$()^T$	Transpose
$()^H$	Hermitian transpose
$\mathcal{E}\{\mathbf{x}\}$	Expectation of the random variable $\mathbf{x}$
$\delta(t)$	Dirac delta function
$\arg\max_{\mathbf{x}}$	The argument $\mathbf{x}$ that maximizes the operand
$\mathbf{x} \sim p(\mathbf{x})$	$\mathbf{x}$ is distributed according to distribution $p(\mathbf{x})$
$\mathcal{N}(m, \sigma^2)$	Gaussian distribution with mean m and variance σ^2
$\mathcal{CN}(m, \sigma^2)$	Complex Gaussian distribution with mean m and variance σ^2
$\mathcal{O}(L)$	Operations in order L
$\mathcal{U}[a, b]$	Uniform distribution on the interval $[a, b]$
$\sim$	sampled from
$\prod$	Interleaver
$\prod^{-1}$	Deinterleaver

Symbols Used in Chapter 2

$\mathbf{d}$	Process noise vector

$\mathbf{v}$	Measurement noise vector
$\mathbf{x}$	State vector of system
$\mathbf{y}$	Observed measurement vector
$\mathbf{f_k}, \mathbf{g_k}$	Generic vector-valued functions at time k
$p(\mathbf{x})$	Probability density of variable $\mathbf{x}$
$\mathbf{K}$	Kalman gain
$\mathbf{F}$	Transition matrix
$\mathbf{G}$	Measurement matrix
$\mathbf{P}$	Covariance matrix
$\mathbf{Q}$	Covariance matrix of process noise
$\mathbf{R}$	Covariance matrix of measurement noise
χ	Sigma point
W	Associated weight with sigma point
L	Dimension of the state $\mathbf{x}$
N	Number of particles (samples)
$q(\cdot)$	Importance sampling distribution
w	Importance weight

Symbols Used in Chapter 3

$\mathbf{s}$	Symbol sequence
r	Received baseband signal
n	Additive noise
T_s	Symbol period
T_m	Channel multipath spread
$h, \mathbf{h}, \mathbf{H}$	Channel coefficient, vector, and matrix
$g(t;\tau)$	Composite channel
g_{TX}	Transmit filter
g_{RX}	Receive filter
f_c	Carrier frequency
f_D	Doppler frequency
J_0	Zero-order Bessel function of the first kind
p	Order of AR model

L	Length of channel response
M_T	Number of transmit antennas
M_R	Number of receive antennas
E_s	Symbol energy
B	Bandwidth of the signal $s(t)$
$H(f)$	Frequency response of channel
$S(f)$	Frequency response of transmitted signal
$R(f)$	Frequency response of received signal
$N(f)$	Frequency response of noise
w	Weight of equalizer (or neural networks)
M	Number of tap weights
v	Input to equalizer or output of channel
y	Output of equalizer (or neural networks)
$W(z)$	Forward filter of DFE
$V(z)$	Feedback filter of DFE
N_s	Number of channel states
$\mathbf{W}$	Weight matrix of neural networks
$\mathbf{P}$	Inverse correlation matrix in RLS algorithm
$\mathbf{P}$	Error correlation matrix in Kalman filter algorithm
$\mathbf{k}$	Gain vector in RLS algorithm
$\mathbf{K}$	Kalman gain in Kalman filter algorithm
ξ	*a priori* estimation error
e	*a posteriori* estimation error
$\mathcal{L}$	Log likelihood ratio
$\mathbf{F}$	Transition matrix
$\mathbf{G}$	Measurement matrix
$P_\alpha(x)$	Characteristic function of α-stable random variable x
$p_\alpha(x)$	PDF of α-stable random variable x
$S_\alpha(\cdot)$	α-stable noise distribution
$q(\cdot)$	Clipping function
N_0	Noise power
S_0	Geometric noise power

Symbols Used in Chapters 4 and 5

s	Transmitted symbol
v	Additive noise
r	Received signal
$g(\cdot)$	Nonlinear distortion function
h	FIR of the channel
N	Length of FIR
m	Feedforward order of DFE
n	Feedback order of DFE
d	Decision delay of DFE
$\mathbf{w}$	Weight vector of recurrent network
$\mathbf{W}$	Weight matrix of recurrent network
φ	Activation function of recurrent network
η	Learning rate
$\mathbf{e}$	Error vector
$\boldsymbol{\omega}$	Process noise vector
$\boldsymbol{\nu}$	Measurement noise vector
$\mathbf{Q}$	Covariance matrix of process noise
$\mathbf{R}$	Covariance matrix of measurement noise
L	Dimension of weights
$\mathcal{W}$	Sigma point matrix of weights
W	Weights associated with sigma points

Symbols Used in Chapters 6 and 7

$\mathbf{s}, \mathbf{S}$	Transmitted symbol vector and matrix
$\mathbf{v}$	Additive noise vector
$\mathbf{w}$	Driving noise of process
$\mathbf{r}$	Received signal vector
$\mathbf{h}, \mathbf{H}$	Channel vector and matrix
M_T	Number of transmit antennas
M_R	Number of receive antennas

I	Mutual information
C	Channel capacity
$\mathcal{L}$	Log likelihood ratio
$\mathcal{D}(\cdot)$	Decision device
$\mathbf{F}$	State transition matrix
$p(\mathbf{x})$	Distribution of the state $\boldsymbol{x}$
$q(\mathbf{x})$	Proposal distribution (or importance density) of the state $\boldsymbol{x}$
w	Importance weight
$\tilde{w}$	Normalized importance weight
N	Number of particles (or samples)

Chapter 1

Introduction

1.1 Signal Processing for Communications

Signal processing is a key research area in communication systems. Currently, the telecommunication industries deploy third generation (3G) systems worldwide. The next generation systems, called fourth generation (4G), will be commercialized in around 2010. Signal processing will encounter the new challenges found in future communication systems. These include high data transmission rates up to 1 Giga bits/second, low bit error rates, multimedia communications, seamless global roaming, quality of service (QoS) management, high user capacity, as well as integration and compatibility between 3G and 4G systems. To resolve these challenges, researchers concentrate their attentions on different signal processing issues.

In digital communications, the transmitter sends signals through distortive channels such as phone, cable, and wireless radio. At the receiver, the signals transmitted are recovered using signal processing techniques. High speed data transmission over communication channels is subject to distortion and noise. Linear distortion such as intersymbol interference (ISI) is usually the result of the limited bandwidth allocated to the channel and/or the presence of multipath propagation. In addition to ISI, in wireless communication systems there exist more impairments such as the effect of mobility of the transmitter and the receiver, interference from other users in the same or adjacent frequency spectrums, as well as time-variation, more commonly known as fading. Wireless communication channels, which are the transfer functions between the transmitter and the receiver, are time, frequency and space selective. Therefore, channel impulse responses (CIR)

change as a function of time, frequency and location.

The receiver has to compensate for the distortion and noise introduced by the channel environments in order to reliably detect the transmitted information data. There are two main streams of signal processing techniques for the receiver. One approach is to equalize the received signals by looking for the inverse system of the CIR. Equalization is a process which reconstructs the transmitted data by jointly combating the effect of ISI and noise in communication links. The optimum solution for equalization is a Bayesian approach known as the maximum *a posteriori* probabilities (MAP) symbol-by-symbol decision equalizer [1]. The other approach is to estimate the CIR and to perform the data detection based on the channel estimate. Maximum likelihood sequence estimation (MLSE) is the optimum solution for data sequence detection in the presence of a perfectly known channel and (white) Gaussian noise [2].

Although signal processing methods for the receivers have been thoroughly studied in the literature, by and large, the results are based on the assumption that the underlying noise and interference processes are Gaussian. However, in many physical channels, such as urban, indoor radio and underwater acoustic channels, the ambient noise is known through experimental measurements to be non-Gaussian, mainly due to the impulsive nature of man-made electromagnetic interference [3]. It is well known that non-Gaussian noise can cause significant performance degradation in traditional communication systems designed under the Gaussian assumption. When the performance degradation due to the ideal Gaussian assumption in non-Gaussian environments cannot be tolerated, the underlying signal processing methods must be revisited and redesigned taking into account the statistics of non-Gaussian noise.

In the meantime, the use of multiple-antenna arrays at both transmit and receive sides, which has recently emerged as one of the most significant technical breakthroughs in modern communications, has opened a new view of possibilities in the design of transmit and receive algorithms. A key feature of multiple-input multiple-output (MIMO) links is the ability to turn multipath propagation, which is traditionally regarded as the chief pitfall to achieve high data rates in wireless transmission, into a benefit for the user in a tremendous improvement of channel capacity [4],[5]. MIMO systems, equipped with multiple antenna elements at both the transmitter and the receiver, can take advantage of random fading with rich multipath scattering [6] and multipath delay spread [7] when available. The performance improvements resulting from the use of MIMO systems

are due to array gain, diversity gain, spatial multiplexing gain, and interference reduction. These have prompted progress of new research topics in areas as diverse as signal processing, channel modeling, information theory and coding, antenna design, and fixed or mobile wireless systems [8].

Due to the potential of MIMO systems, the efforts of the signal processing community have been contributed to the development of MIMO-oriented architectures. Remarkable examples typically fall into two categories: data rate maximization with spatial multiplexing and diversity maximization with space-time coding schemes. In spatial multiplexing, often referred to in the literature as BLAST (Bell-labs LAyered Space-Time) architectures [5],[9], streams of independent data are transmitted independently over different antennas, maximizing the average data rate over wireless MIMO channels. Space-time codes [10],[11] introduce certain redundancy in both space and time domains to achieve diversity maximization. In a viewpoint of signal processing, advanced algorithms for MIMO receivers have been developed, yet further research is in great demand. These include signal separation, channel equalization, channel estimation as well as detection and decoding techniques at the receiver side.

In this thesis, we address two problems of signal processing in communication systems: channel equalization for single-transmit single-receive (STSR) links and channel estimation for MIMO links. Signal processing algorithms to be developed throughout this thesis are based on a recursive filtering framework, represented in a dynamic state-space model of the systems.

1.2 Recursive Filtering Framework

In many engineering applications, it is needed to extract unknown quantities of interest from the data corrupted by additive noise and interferences of different kinds. The data often arrive sequentially in time. Therefore, sequential data processing is required to make on-line decisions. In this case, recursive filtering is a convenient solution. A recursive filtering framework for estimation of the state of a discrete-time dynamic system is built on the state-space model of a dynamic system,

consisting of two equations

$$\mathbf{x}[k+1] = \mathbf{f}(\mathbf{x}[k], \mathbf{u}[k], \mathbf{d}[k]) \quad (1.1)$$

$$\mathbf{y}[k] = \mathbf{g}(\mathbf{x}[k], \mathbf{v}[k]) \quad (1.2)$$

which describe the process and measurement equations, respectively. The vector $\mathbf{x}[k]$ represents the unobserved state of the system, $\mathbf{u}[k]$ is a known exogeneous input, and $\mathbf{y}[k]$ is the observed measurement signal. The process noise $\mathbf{d}[k]$ drives the dynamic system, and the measurement (or observation) noise is given by $\mathbf{v}[k]$. The system dynamical model $\mathbf{f}$ and $\mathbf{g}$ are two generic vector-valued (or matrix) functions, which are potentially time-varying and assumed known. The state equation (1.1) characterizes the state transition probability $p(\mathbf{x}[k+1]|\mathbf{x}[k])$, whereas the measurement equation (1.2) describes the likelihood (or measurement probability) $p(\mathbf{y}[k]|\mathbf{x}[k])$.

For the standard Kalman filter (KF) applied to a linear dynamic system, the state-space model is stochastic owing to the additive presence of the process noise and the measurement noise, which are assumed to be Gaussian with zero mean and known covariance matrices. Applications of KF theory can be extended to nonlinear dynamic systems, in which the extended Kalman filter (EKF) [12] is usually applied. The derivation of the EKF hinges on linearization of the nonlinear state-space model, on the assumption that deviation from linearity is of first order. The EKF has been used as the standard technique for performing recursive nonlinear estimation. However, the EKF algorithm provides only an approximation to the optimal nonlinear equation, because the state distribution is approximated by a Gaussian random variable and is then propagated analytically through the first-order linearization of the nonlinear system. The unscented Kalman filter (UKF) [13],[14] using a deterministic sampling approach and an unscented transform (UT) can approximate a nonlinear dynamic system accurately to the second-order for any nonlinearity. In the UKF, the state distribution is also approximated by a Gaussian random variable, but it is represented using a set of carefully chosen sample points, often referred as *sigma points*. The sigma points can capture the true mean and covariance of the Gaussian random variable. Furthermore, the UKF can be applied to non-Gaussian random variables to get approximations which are accurate to at least the second order, with the accuracy of third- and higher-order moments being determined by the choice of parameters required in the UT [15].

The sequential Monte Carlo methods [16], also known as particle filtering, have recently emerged in the field of statistics and engineering, and are believed to be a key technology whenever nonlinearity or non-Gaussianity is involved in the system of interest. Particle filters allow for a more accurate representation of the state distribution using sequential sampling and resampling. The use of particle filters can be used as a tool for tracking the time variations in the channel state in wireless communications [17].

The recursive filtering framework can be extended to parameter estimation, sometimes referred as system identification or machine learning, by writing a new state-space representation,

$$\mathbf{x}[k+1] = \mathbf{x}[k] + \mathbf{d}[k] \tag{1.3}$$

$$\mathbf{y}[k] = \mathbf{g}\left(\mathbf{x}[k], \mathbf{u}[k]\right) + \mathbf{v}[k] \tag{1.4}$$

where $\mathbf{x}[k]$ corresponds to a stationary process with identity state transition matrix, driven by the process noise $\mathbf{d}[k]$, the output $\mathbf{y}[k]$ corresponds to a nonlinear observation of the parameter vector $\mathbf{x}[k]$ and the input $\mathbf{u}[k]$, and the nonlinear mapping $\mathbf{g}(\cdot)$ is parameterized by the vector $\mathbf{x}[k]$. For example, the nonlinear mapping can be represented by neural networks where the vector $\mathbf{x}$ is the weights, with various applications in regression, classification, approximation, and dynamic modeling. Learning (or training) of the neural networks corresponds to estimating the parameters $\mathbf{x}$, with a training set of known input-output pairs.

In this thesis, recurrent neural networks (RNNs) are employed for the design of adaptive nonlinear equalizers which compensate for the effects of ISI and noise. The RNNs are inherently nonlinear dynamic systems and can be represented by state-space models that serve as the basis for parameter estimation in the recursive filtering framework. The EKF and the UKF algorithms are applied to training recurrent neural equalizers for both time-invariant and time-varying ISI channels, over Gaussian or non-Gaussian noise environments. On the other hand, channel estimation for MIMO wireless systems can be treated as a state estimation problem. With an iterative receiver structure, adaptive filtering-based iterative channel estimation techniques are investigated for narrowband MIMO wireless channels. Particle filtering is incorporated into the iterative receiver in order to estimate fast fading MIMO channels over both Gaussian and non-Gaussian noise environments.

1.3 Organization and Contributions

This thesis is organized as three major parts. First of all, Part I consists of two chapters, Chapter 2 and Chapter 3, in which some background and review are provided. Then, Part II consists of two chapters, Chapter 4 and Chapter 5, in which we study adaptive neural equalization for single transmit single receive (STSR) systems in ISI channels. We start this part by proposing a recurrent neural equalizer trained with the EKF algorithms in Chapter 4. In order to deal with non-Gaussian noise, we present a recurrent multilayer perceptron equalizer trained with the UKF algorithm in Chapter 5. Finally, Part III consists of another two chapters, Chapter 6 and Chapter 7. We study adaptive filtering-based iterative channel estimation in MIMO wireless communications in Chapter 6. Particle filtering-based channel tracking is proposed in Chapter 7.

The contributions of each chapter are as follows:

Chapter 2: This chapter presents an overview of recursive Bayesian filtering. This includes a state-space system model, Bayesian estimation, and recursive filtering solutions, that are applied throughout this thesis.

Chapter 3: This chapter gives an in-depth literature review and presents the fundamental concepts of channel models, channel equalization and estimation techniques, and non-Gaussian noise models for communication systems. Classification of training-based channel equalizers is given in terms of types, structures and algorithms. For channel estimation, training-based and semiblind estimation techniques are covered in detail. Moreover, an in-depth treatment of two non-Gaussian noise models used to model practical communication channels is given.

Chapter 4: In Chapter 4, two versions of the EKF algorithm, the global EKF (GEKF) and the decoupled EKF (DEKF), are investigated for the training of a decision-feedback recurrent neural equalizer (DFRNE) using the fully connected recurrent network. Based on real-time recurrent learning (RTRL), the GEKF and the DEKF algorithms are developed for the complex-valued DFRNE over dispersive Gaussian channels with nonlinear distortions. The main properties of the EKF-trained DFRNEs include a fast convergence, a lower bit error probability and a better time-varying tracking ability, compared to the conventional RTRL-trained DFRNE. The DEKF leads to a computationally efficient implementation over the GEKF, without performance degradation.

The technical contributions of this chapter have been published in the following journal paper:

- J. Choi, M. Bouchard, and T. H. Yeap, "Decision Feedback Recurrent Neural Equalization with Fast Convergence Rate," *IEEE Transactions on Neural Networks*, Vol. 16, No. 3, pp. 699-708, May 2005.

Chapter 5: In this chapter, an adaptive nonlinear equalizer using a recurrent multilayer perceptron (RMLP) is presented to effectively compensate for non-Gaussian noise and ISI. The UKF algorithm is employed to train the RMLP equalizer, offering more accurate approximations for non-Gaussian noise than the EKF. The results demonstrate that the RMLP-UKF is a suitable solution for compensating the effects of nonlinear distortions and α-stable impulsive noise channels.

The work of this chapter has been published in a conference paper and submitted to a journal for publication:

- J. Choi, M. Bouchard, and T. H. Yeap, "Recurrent Neural Equalization for Communication Channels in Impulsive Noise Environments," *International Joint Conference on Neural Networks* (IJCNN), Montreal, Canada, July 31-August 4, 2005.

- J. Choi, M. Bouchard, and T. H. Yeap, "Adaptive Neural Equalization for Non-Gaussian Channels in Digital Communication Systems," Submitted to *IEEE Transactions on Neural Networks* (2nd round).

There are more technical contributions related to the work of this chapter. These include channel equalization for Gaussian noise channels and nonlinear system modeling, based on recurrent neural networks and the UKF algorithm. The results are not included in the thesis, but can be found in:

- J. Choi, M. Bouchard, T. H. Yeap and O. Kwon, "A Derivative-Free Kalman Filter for Parameter Estimation of Recurrent Neural Networks and Its Applications to Nonlinear Channel Equalization," *Proceedings of Fourth International ICSC Symposium on Engineering of Intelligent Systems* (EIS), Madeira, Portugal, Feb 29 - Mar 2, 2004.

- J. Choi, T. H. Yeap and M. Bouchard, "Nonlinear State-Space Modeling using Recurrent Multilayer Perceptrons with Unscented Kalman Filter," *Proceedings of IEEE International Conference on Systems, Man and Cybernetics* (SMC), pp. 3427-3432, The Hague, Netherlands, October 10-13, 2004.

- J. Choi, T. H. Yeap, and M. Bouchard, "Online State-Space Modeling using Recurrent Multilayer Perceptrons with Unscented Kalman Filter," *Neural Processing Letters*, Vol. 22, No. 1, pp. 69-84, August 2005.

Chapter 6: In this chapter, iterative channel estimation approaches combined with turbo-BLAST, which is a promising iterative MIMO receiver, are investigated over block Rayleigh fading channels. On the framework of semiblind channel estimation, both snapshot approaches and adaptive filtering approaches are used as iterative MIMO channel estimators. A finding through experiments is that iterative channel estimation performed with hard-decision feedback from the decoder achieves more accurate channel estimates and better performance gain than their counterpart with soft-decision feedback. In the iterative strategy making use of the entire received data for estimating the channel, we find that adaptive filters (LMS, RLS and KF) can be used for a computationally efficient implementation of channel estimation, avoiding the matrix inversion of snapshot approaches.

Part of the technical contributions have been published as a conference paper:

- J. Choi, M. Bouchard, and T. H. Yeap, "Adaptive Filtering-Based Iterative Channel Estimation for MIMO Wireless Communications," *Proceedings of IEEE International Symposium on Circuits and Systems* (ISCAS), Kobe, Japan, May 23-26, 2005.

Chapter 7: Particle filtering techniques are exploited in order to iteratively track time-varying MIMO channels. The turbo-BLAST receiver with the inclusion of two particle filters for channel tracking, the sampling importance resampling (SIR) and the gradient proposal particle filter (GPF), achieves an improved performance in both bit error rate and channel estimation errors, compared to the static receiver with no channel tracking in fast fading environments. In particular, the use of the GPF algorithm leads to overcoming the error floor phenomenon over non-Gaussian noise.

Part of the results have been submitted to:

- J. Choi, M. Bouchard, and T. H. Yeap, "Iterative Channel Tracking using Particle Filtering for MIMO Wireless Communications," In preparation for publication.

Chapter 8: In this chapter, the main results undertaken in this thesis are summarized and discussed, and possible directions for further research are presented.

Part I

Background and Review

Chapter 2

Recursive Bayesian Filtering

2.1 State-Space Model

We concentrate on a probabilistic estimation (or inference) problem with a discrete-time formulation of the systems. A framework for estimation of the state of a discrete-time dynamic system is built on a generic state-space model of the system. In this thesis we consider a dynamic state-space model (DSSM), which can be described by a graphical representation as in Fig. 2.1. To make estimation about a dynamic system, two models are needed: the system model and the measurement model. The system model is a model describing the evolution of the state with time, whereas the measurement model is a model relating the noisy measurement to the state. The DSSM consists of two equations

$$\mathbf{x}[k+1] = \mathbf{f}_k\left(\mathbf{x}[k], \mathbf{u}[k], \mathbf{d}[k]\right) \tag{2.1}$$

$$\mathbf{y}[k] = \mathbf{g}_k\left(\mathbf{x}[k], \mathbf{u}[k], \mathbf{v}[k]\right) \tag{2.2}$$

which describe the process (or state) and measurement (or observation) equations, respectively. In the DSSM above, $\mathbf{x}[k]$ represents the unobserved state vector of the system, $\mathbf{u}[k]$ is a known exogeneous input vector that is often provided to process or measurement, or both equations, and $\mathbf{y}[k]$ is the observed measurement vector. $\mathbf{d}[k]$ and $\mathbf{v}[k]$ are the process and measurement noise vector, respectively, with a known variance. The system functions $\mathbf{f}_k(\cdot)$ and $\mathbf{g}_k(\cdot)$ are two generic vector-valued (or matrix) functions, which are potentially time-varying. The analytical forms of the functions $\mathbf{f}_k(\cdot)$ and $\mathbf{g}_k(\cdot)$ are assumed to be known. We also assume that the initial probability

density of the state $p(\mathbf{x}[0])$[1] is known. The state transition density $p(\mathbf{x}[k+1]|\mathbf{x}[k])$ is specified by the state transition function $\mathbf{f}_k(\cdot)$ and the process noise distribution $p(\mathbf{d}[k])$, whereas the likelihood (or measurement density) $p(\mathbf{y}[k]|\mathbf{x}[k])$ is fully specified by the measurement function $\mathbf{g}_k(\cdot)$ and the measurement noise distribution $p(\mathbf{v}[k])$.

To sum up, the model is described by

$$\begin{aligned} p(\mathbf{x}[0]) & \\ p(\mathbf{x}[k]|\mathbf{x}[k-1]) & \quad \text{for } k \geq 1 \\ p(\mathbf{y}[k]|\mathbf{x}[k]) & \quad \text{for } k \geq 1. \end{aligned}$$

The DSSM provides a basic framework for algorithmic forms of probabilistic inference, i.e. state estimation, parameter estimation and dual estimation. In what follows, we denote the signals and measurements up to time k as, respectively

$$\begin{aligned} \mathbf{x}[0:k] &\equiv \{\mathbf{x}[0], \mathbf{x}[1], \ldots, \mathbf{x}[k]\} \\ \mathbf{y}[1:k] &\equiv \{\mathbf{y}[1], \mathbf{y}[2], \ldots, \mathbf{y}[k]\}. \end{aligned}$$

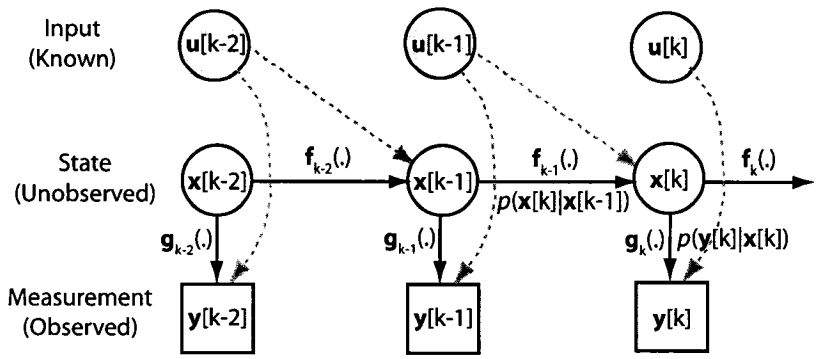


Fig. 2.1 Graphical representation of a dynamic state-space model.

[1]For simplicity, we express discrete probability distributions using $p(\mathbf{x}[k])$ instead of $Pr(\mathbf{X}[k] = \mathbf{x}[k])$, where we use $\mathbf{x}[k]$ to denote both the random variable and its realization.

2.2 Bayesian Estimation

The objective of the estimation procedure is to gather information about the value of the state $\mathbf{x}[k]$, given a set of all available measurements $\mathbf{y}[1:k]$ up to time k. The Bayesian estimation approach provides an elegant and consistent method of dealing with uncertainties. In a Bayesian framework, our aim is to estimate recursively in time the *posterior* distribution $p(\mathbf{x}[0:k]|\mathbf{y}[1:k])$, its associated features (including the marginal distribution $p(\mathbf{x}[k]|\mathbf{y}[1:k])$, known as the *filtering* distribution), and the expectations

$$E(f) = \mathcal{E}_{p(\mathbf{x}[0:k]|\mathbf{y}[1:k])}\{f(\mathbf{x}[0:k])\} \equiv \int f(\mathbf{x}[0:k])p(\mathbf{x}[0:k]|\mathbf{y}[1:k])d\mathbf{x}[0:k] \tag{2.3}$$

where $f(\mathbf{x}[0:k])$ is a function of interest, which is integrable with respect to $p(\mathbf{x}[0:k]|\mathbf{y}[1:k])$. Examples of the function $f(\mathbf{x}[0:k])$ include the conditional mean or the conditional covariance.

The Bayesian *filtering* problem is to calculate $p(\mathbf{x}[k]|\mathbf{y}[1:n])$ for $n = k$, i.e. to find the minimum mean-square error (MMSE) estimate of the state $\mathbf{x}[k]$, given the measurements up to time k. The related problems of evaluating $p(\mathbf{x}[k]|\mathbf{y}[1:n])$ is called *prediction* if $n < k$ and *smoothing* if $n > k$.

The posterior distribution at time k is given by *Bayes' theorem*

$$p(\mathbf{x}[0:k]|\mathbf{y}[1:k]) = \frac{p(\mathbf{y}[1:k]|\mathbf{x}[0:k])p(\mathbf{x}[0:k])}{p(\mathbf{y}[1:k])} \tag{2.4}$$

which reflects all the information about the state of the system $\mathbf{x}[0:k]$, conditioned in the measurements $\mathbf{y}[1:k]$, and the prior density $p(\mathbf{x}[0:k])$. Taking into account that the measurements up to time k are independent given $\mathbf{x}[0:k]$, the likelihood $p(\mathbf{y}[1:k]|\mathbf{x}[0:k])$ in (2.4) can be factorized as

$$p(\mathbf{y}[1:k]|\mathbf{x}[0:k]) = \prod_{i=1}^{k} p(\mathbf{y}[i]|\mathbf{x}[0:k]) \tag{2.5}$$

$$= \prod_{i=1}^{k} p(\mathbf{y}[i]|\mathbf{x}[i]). \tag{2.6}$$

Equation (2.6) reflects that the measurement $\mathbf{y}[i]$ is independent of the states at all other times since

it is conditional on $\mathbf{x}[i]$. Furthermore, because the state equation (2.1) satisfies a Markov chain, the prior density $p(\mathbf{x}[0:k])$ can be as follows:

$$p(\mathbf{x}[0:k]) = p(\mathbf{x}[0]) \prod_{i=1}^{k} p(\mathbf{x}[i]|\mathbf{x}[i-1]). \tag{2.7}$$

By applying (2.6) and (2.7) to (2.4), the posterior density can then be as follows:

$$p(\mathbf{x}[0:k]|\mathbf{y}[1:k]) = \frac{p(\mathbf{x}[0]) \prod_{i=1}^{k} p(\mathbf{y}[i]|\mathbf{x}[i]) p(\mathbf{x}[i]|\mathbf{x}[i-1])}{p(\mathbf{y}[1:k])}. \tag{2.8}$$

The posterior filtering density of interest $p(\mathbf{x}[k]|\mathbf{y}[1:k])$ can be obtained by marginalization of (2.8). However, the dimension of the integration grows as k increases. This can be avoided by using a recursion procedure. It is possible to obtain straightforwardly a recursive formula for the joint probability density $p(\mathbf{x}[0:k]|\mathbf{y}[1:k])$ from (2.8),

$$p(\mathbf{x}[0:k]|\mathbf{y}[1:k]) = p(\mathbf{x}[0:k-1]|\mathbf{y}[1:k-1]) \frac{p(\mathbf{y}[k]|\mathbf{x}[k]) p(\mathbf{x}[k]|\mathbf{x}[k-1])}{p(\mathbf{y}[k]|\mathbf{y}[1:k-1])}. \tag{2.9}$$

The posterior (marginal) density $p(\mathbf{x}[k]|\mathbf{y}[1:k])$ can be obtained recursively in two stages, called *prediction* and *update* [18]. In the following the detailed recursion is presented under the two assumptions: 1) The states follows a first-order Markov process $p(\mathbf{x}[k]|\mathbf{x}[0:k-1]) = p(\mathbf{x}[k]|\mathbf{x}[k-1])$; and 2) The measurements are independent of the given states.

Prediction Stage:

In the prediction stage, the filtering density $p(\mathbf{x}[k]|\mathbf{y}[1:k-1])$ is propagated into the future by the transition density $p(\mathbf{x}[k-1]|\mathbf{x}[1:k-1])$ and it is obtained as follows:

$$p(\mathbf{x}[k]|\mathbf{y}[1:k-1]) = \int p(\mathbf{x}[k]|\mathbf{x}[k-1]) p(\mathbf{x}[k-1]|\mathbf{y}[1:k-1]) d\mathbf{x}[k-1]. \tag{2.10}$$

This relation is also referred to as the *Chapman-Kolmogorov* equation. The transition density $p(\mathbf{x}[k]|\mathbf{x}[k-1])$ is defined in terms of the probabilistic model governing the state's evolution and the process noise statistics.

Update Stage:

From Bayes' rule, the marginal density becomes

$$\begin{aligned} p(\mathbf{x}[k]|\mathbf{y}[1:k]) &= \frac{p(\mathbf{y}[1:k]|\mathbf{x}[k])p(\mathbf{x}[k])}{p(\mathbf{y}[1:k])} \\ &= \frac{p(\mathbf{y}[k],\mathbf{y}[1:k-1]|\mathbf{x}[k])p(\mathbf{x}[k])}{p(\mathbf{y}[k],\mathbf{y}[1:k-1])} \\ &= \frac{p(\mathbf{y}[k]|\mathbf{y}[1:k-1],\mathbf{x}[k])p(\mathbf{y}[1:k-1]|\mathbf{x}[k])p(\mathbf{x}[k])}{p(\mathbf{y}[k]|\mathbf{y}[1:k-1])p(\mathbf{y}[1:k-1])} \\ &= \frac{p(\mathbf{y}[k]|\mathbf{y}[1:k-1],\mathbf{x}[k])p(\mathbf{x}[k]|\mathbf{y}[1:k-1])p(\mathbf{y}[1:k-1])p(\mathbf{x}[k])}{p(\mathbf{y}[k]|\mathbf{y}[1:k-1])p(\mathbf{y}[1:k-1])p(\mathbf{x}[k])} \\ &= \frac{p(\mathbf{y}[k]|\mathbf{x}[k])p(\mathbf{x}[k]|\mathbf{y}[1:k-1])}{p(\mathbf{y}[k]|\mathbf{y}[1:k-1])} \end{aligned} \tag{2.11}$$

where the normalizing constant

$$p(\mathbf{y}[k]|\mathbf{y}[1:k-1]) = \int p(\mathbf{y}[k]|\mathbf{x}[k])p(\mathbf{x}[k]|\mathbf{y}[1:k-1])d\mathbf{x}[k]. \tag{2.12}$$

In summary, the recursion is described by three terms: the prior density $p(\mathbf{x}[k]|\mathbf{y}[1:k-1])$, the likelihood $p(\mathbf{y}[k]|\mathbf{x}[k])$, and the evidence $p(\mathbf{y}[k]|\mathbf{y}[1:k-1])$. Calculation of the three terms is the fundamentals of Bayesian filtering and inference. Fig. 2.2 illustrates how to recursively estimate the posterior filtering density in two stages.

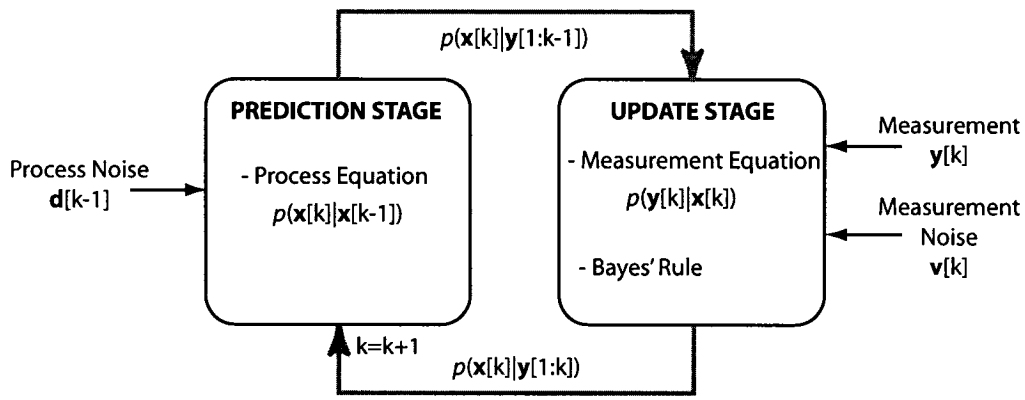


Fig. 2.2 Prediction and update stages in the recursion.

2.3 Recursive Filtering Solutions

The recursive expressions in (2.10) and (2.11) are deceptively simple because the integrals involved are usually intractable. Because the requirement of evaluating complex high-dimensional integrals, one can not typically compute the marginal densities of the posterior density $p(\mathbf{x}[0:k]|\mathbf{y}[1:k])$, the normalizing constant $p(\mathbf{y}[k]|\mathbf{y}[1:k])$ and the expectations $E(f)$, except for several special cases. This is why, from the mid-1960s, a great number of different approaches and filters have been devoted to obtaining approximations for these distributions. These approximations include the Kalman filters, the mixture model (e.g., Gaussian-sum filter), grid-based methods, and sequential Monte Carlo integration methods. In particular, due to the limitation of computational power available in the 1960s, sequential Monte Carlo methods were overlooked and forgotten. In the late of 1980s, the great increase of computational power made rapid advances possible in sequential Monte Carlo methods for recursive Bayesian filtering. In the following subsections, we introduce some of these approximations for recursive Bayesian filtering.

2.3.1 Linear Gaussian Approximation: Kalman Filter

Since the optimal recursive Bayesian solution, requiring the propagation of the full posterior density, does not put any restriction on the form of the posterior density, it can not generally be described by a finite number of parameters. Therefore, any practical estimator has to make some form of approximating assumptions with regard to the shape of the posterior density and the form of the Bayesian recursion. A group of Gaussian approximate solutions is based on the Kalman filter framework [19].

The recursion in (2.10) and (2.11) can be solved by the celebrated Kalman filter [19], which is the closed-form recursive solution of Bayesian estimation for linear, Gaussian systems. If a DSSM is linear, with uncorrelated system and measurement Gaussian noise and a Gaussian prior density, it is of the following form:

$$\mathbf{x}[k+1] = \mathbf{F}[k+1,k]\mathbf{x}[k] + \mathbf{d}[k] \tag{2.13}$$

$$\mathbf{y}[k] = \mathbf{G}[k]\mathbf{x}[k] + \mathbf{v}[k] \tag{2.14}$$

where $\mathbf{F}[k+1,k]$ is the transition matrix taking the state $\mathbf{x}[k]$ from time k to $k+1$ and $\mathbf{G}[k]$ is the measurement matrix. The covariance matrices of the process and measurement noise are respectively defined by

$$\mathcal{E}\{\mathbf{d}[n]\mathbf{d}^T[k]\} = \delta_{nk}\mathbf{Q}[k] \tag{2.15}$$

$$\mathcal{E}\{\mathbf{v}[n]\mathbf{v}^T[k]\} = \delta_{nk}\mathbf{R}[k], \quad \forall n,\, k \tag{2.16}$$

where the process noise $\mathbf{d}[k]$ is uncorrelated with the measurement noise $\mathbf{v}[k]$, i.e. $\mathcal{E}\{\mathbf{d}[n]\mathbf{v}^T[k]\} = \mathbf{0}$. Eventually, the prior probability density $p(\mathbf{x}[k]|\mathbf{y}[1:k-1])$ and the posterior probability density $p(\mathbf{x}[k]|\mathbf{y}[1:k])$ at time k are Gaussian random variables. We may denote the mean and covariance of the prior Gaussian density $p(\mathbf{x}[k]|\mathbf{y}[1:k-1])$ by $\hat{\mathbf{x}}^-[k]$ and $\mathbf{P}^-[k]$, and denote the mean and covariance of the posterior Gaussian density $p(\mathbf{x}[k]|\mathbf{y}[1:k])$ by $\hat{\mathbf{x}}[k]$ and $\mathbf{P}[k]$, respectively.

The Kalman filter is the optimal Gaussian approximate linear Bayesian update, which is called the Kalman update for short, of the conditional (posterior) mean of the state and its (posterior) covariance. The recursive form is as follows:

$$\hat{\mathbf{x}}[k] = \hat{\mathbf{x}}^-[k] + \mathbf{K}[k](\mathbf{y}[k] - \hat{\mathbf{y}}^-[k]) \tag{2.17}$$

$$\begin{aligned}\mathbf{P}[k] &= \mathcal{E}\{(\mathbf{x}[k] - \hat{\mathbf{x}}[k])(\mathbf{x}[k] - \hat{\mathbf{x}}[k])^T\} \\ &= (\mathbf{I} - \mathbf{K}[k]\mathbf{G}[k])\mathbf{P}^-[k]\end{aligned} \tag{2.18}$$

where the error between the true measurement and the predicted measurement, $\tilde{\mathbf{y}}[k] = \mathbf{y}[k] - \hat{\mathbf{y}}^-[k]$, is called the *innovation*. The optimal terms in this recursion are given by

$$\hat{\mathbf{x}}^-[k] = \mathbf{F}[k,k-1]\hat{\mathbf{x}}^-[k-1] \tag{2.19}$$

$$\hat{\mathbf{y}}^-[k] = \mathbf{G}[k]\hat{\mathbf{x}}^-[k] \tag{2.20}$$

$$\mathbf{P}^-[k] = \mathbf{F}[k,k-1]\mathbf{P}[k-1]\mathbf{F}^T[k,k-1] + \mathbf{Q}[k-1] \tag{2.21}$$

$$\mathbf{K}[k] = \mathbf{P}^-[k]\mathbf{G}^T[k](\mathbf{G}[k]\mathbf{P}^-[k]\mathbf{G}^T[k] + \mathbf{R}[k])^{-1}. \tag{2.22}$$

The prior estimate $\hat{\mathbf{x}}^-[k]$ is the optimal prediction of the state $\mathbf{x}[k]$ and is defined in terms of the "old" posterior estimate. Similarly $\hat{\mathbf{y}}^-[k]$ is the optimal prediction of $\mathbf{y}[k]$. The posterior

covariance $\mathbf{P}^{-}[k]$ is dependent on the "old" posterior covariance matrix $\mathbf{P}[k-1]$. Lastly, $\mathbf{K}[k]$ is called the optimal Kalman gain.

The expectations, which requires for evaluation of the covariance terms, can in general be calculated exactly in an analytical sense for a linear DSSM and Gaussian random noises. In fact, under these linear and Gaussian assumptions the Kalman filter is an exact solution of the optimal Bayesian recursion. The recursive algorithm of the Kalman filter is summarized in Table 2.1.

Table 2.1 Description of the Kalman filter.

1. *Initialization*
   At $k = 0$, set:
   $\hat{\mathbf{x}}[0] = \mathcal{E}\{\mathbf{x}[0]\}$,
   $\mathbf{P}[0] = \mathcal{E}\{(\mathbf{x}[0] - \hat{\mathbf{x}}[0])(\mathbf{x}[0] - \hat{\mathbf{x}}[0])^T\}$.
2. *Computation*
   For $k = 1, 2, \ldots$, perform the computations:
   • *Time Updates* (Prediction Step)
   - State estimation propagation: Equation (2.19).
   - Error covariance propagation: Equation (2.21).
   • *Measurement Updates* (Correction Step)
   - Kalman gain matrix: Equation (2.22).
   - State estimate update: Equation (2.17).
   - Error covariance update: Equation (2.18).

2.3.2 Nonlinear Gaussian Approximation: Extended Kalman Filter

For a linear Gaussian DSSM, the linear transformation of a Gaussian random variable remains Gaussian, and it is implicit due to the fact that all of the higher order (> 1) derivatives of $\mathbf{g}_k$ equate to zero. However, for a nonlinear Gaussian DSSM, we may extend the use of Kalman filtering through a linearization procedure. The resulting filter is called the extended Kalman filter (EKF). Let us consider a nonlinear system described by the following DSSM

$$\mathbf{x}[k+1] = \mathbf{f}_k(\mathbf{x}[k]) + \mathbf{d}[k] \tag{2.23}$$

$$\mathbf{y}[k] = \mathbf{g}_k(\mathbf{x}[k]) + \mathbf{v}[k] \tag{2.24}$$

where the functions $\mathbf{f}_k(\mathbf{x}[k])$ and $\mathbf{g}_k(\mathbf{x}[k])$ denote a nonlinear transition matrix and a nonlinear measurement matrix that may be time-varying, respectively.

The basic idea of the EKF is to linearize the model of equations (2.23) and (2.24) around the current state estimate using a first-order truncation of the multidimensional Taylor-series expansion. The approximation proceeds in two stages [12].

Stage 1: Evaluate the Jacobian of the nonlinear functions

The two matrices are conducted as follow

$$\mathbf{F}[k+1,k] = \left.\frac{\partial \mathbf{f}_k(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}[k]} \tag{2.25}$$

$$\mathbf{G}[k] = \left.\frac{\partial \mathbf{g}_k(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}^-[k]}. \tag{2.26}$$

The entries of the matrices $\mathbf{F}[k+1,k]$ and $\mathbf{G}[k]$ are computable and all known, since $\hat{\mathbf{x}}[k]$ and $\hat{\mathbf{x}}^-[k]$ are available at time k.

Stage 2: Approximate the nonlinear functions

Once the two matrices are evaluated, they are then employed in a first-order Taylor approximation of the nonlinear functions $\mathbf{f}_k(\mathbf{x}[k])$ and $\mathbf{g}_k(\mathbf{x}[k])$ around $\hat{\mathbf{x}}[k]$ and $\hat{\mathbf{x}}^-[k]$, respectively. The resulting approximations of the two nonlinear functions are as follows

$$\mathbf{f}_k(\mathbf{x}[k]) \approx \mathbf{f}_k(\hat{\mathbf{x}}[k]) + \mathbf{F}[k+1,k](\mathbf{x}[k] - \hat{\mathbf{x}}[k]) \tag{2.27}$$

$$\mathbf{g}_k(\mathbf{x}[k]) \approx \mathbf{g}_k(\hat{\mathbf{x}}^-[k]) + \mathbf{G}[k](\mathbf{x}[k] - \hat{\mathbf{x}}^-[k]). \tag{2.28}$$

With the above approximations at hand, the nonlinear DSSM of (2.23) and (2.24) are approximated as follows

$$\mathbf{x}[k+1] \approx \mathbf{F}[k+1,k]\mathbf{x}[k] + \mathbf{d}[k] + \boldsymbol{\xi}[k] \tag{2.29}$$

$$\bar{\mathbf{y}}[k] \approx \mathbf{G}[k]\mathbf{x}[k] + \mathbf{v}[k] \tag{2.30}$$

where the two quantities introduced are

$$\bar{\mathbf{y}}[k] = \mathbf{y}[k] - (\mathbf{g}_k(\hat{\mathbf{x}}^-[k]) - \mathbf{G}[k]\hat{\mathbf{x}}^-[k]) \tag{2.31}$$

$$\boldsymbol{\xi}[k] = \mathbf{f}_k(\hat{\mathbf{x}}[k]) - \mathbf{F}[k+1,k]\hat{\mathbf{x}}[k]. \tag{2.32}$$

In the above equations, $\bar{\mathbf{y}}[k]$ can be regarded as an observation vector because the entries in the term $\bar{\mathbf{y}}[k]$ are all known at time k. Likewise, the entries in the term $\boldsymbol{\xi}[k]$ are also all known at time k. In the linear case, $\bar{\mathbf{y}}[k] = \mathbf{y}[k]$ and $\boldsymbol{\xi}[k] = \mathbf{0}$.

Applying the Kalman filter to these approximations gives the following EKF recursion:

$$\hat{\mathbf{x}}[k] = \hat{\mathbf{x}}^-[k] + \mathbf{K}[k](\mathbf{y}[k] - \mathbf{g}_k(\hat{\mathbf{x}}^-[k])) \tag{2.33}$$

$$\mathbf{P}[k] = (\mathbf{I} - \mathbf{K}[k]\mathbf{G}[k])\mathbf{P}^-[k]. \tag{2.34}$$

The optimal terms in this recursion are given by

$$\hat{\mathbf{x}}^-[k] = \mathbf{f}_k(\hat{\mathbf{x}}[k-1]) \tag{2.35}$$

$$\hat{\mathbf{y}}^-[k] = \mathbf{g}_k(\hat{\mathbf{x}}^-[k]) \tag{2.36}$$

$$\mathbf{P}^-[k] = \mathbf{F}[k,k-1]\mathbf{P}[k-1]\mathbf{F}^T[k,k-1] + \mathbf{Q}[k-1] \tag{2.37}$$

$$\mathbf{K}[k] = \mathbf{P}^-[k]\mathbf{G}^T[k](\mathbf{G}[k]\mathbf{P}^-[k]\mathbf{G}^T[k] + \mathbf{R}[k])^{-1}. \tag{2.38}$$

In Table 2.2, we present a summary of the EKF algorithm.

The linearization method employed in the EKF does not take into account the fact that the state $\mathbf{x}$ is a random variable with inherent uncertainty. Therefore, the approximations often introduce large errors in the posterior mean and covariance of the transformed Gaussian random variable, which may lead to suboptimal performance and sometimes divergence of the filter. Furthermore, the EKF approximates $p(\mathbf{x}[k]|\mathbf{y}[1:k])$ to be Gaussian. When the true density is non-Gaussian such as when it is bimodal or heavily skewed, a Gaussian approximation can not describe it well.

Table 2.2 Description of the extended Kalman filter.

1. *Initialization*
   At $k = 0$, set:
   $\hat{\mathbf{x}}[0] = \mathcal{E}\{\mathbf{x}[0]\}$,
   $\mathbf{P}[0] = \mathcal{E}\{(\mathbf{x}[0] - \hat{\mathbf{x}}[0])(\mathbf{x}[0] - \hat{\mathbf{x}}[0])^T\}$.
2. *Computation*
   For $k = 1, 2, \ldots$, perform the computations:
   • *Time Updates* (Prediction Step)
   - State estimation propagation: Equation (2.35).
   - Error covariance propagation: Equation (2.37).
   • *Measurement Updates* (Correction Step)
   - Kalman gain matrix: Equation (2.38).
   - State estimate update: Equation (2.33).
   - Error covariance update: Equation (2.34).

2.3.3 Derivative-Free Estimation: Unscented Kalman Filter

Two major flaws in the EKF recursion applied to general nonlinear DSSMs need to be addressed. First, the probabilistic spread of the state and noise random variables, as captured by the covariance, plays an important role in the validity of the EKF's first-order approximation. However, it is completely disregarded during the linearization process. Next, due to the first-order accuracy of the propagated means and covariances, the posterior mean estimate is clearly biased and the posterior covariance estimate is highly inaccurate and inconsistent. Some related algorithms [13],[20] have recently been proposed. These algorithms address the issues mentioned above by making use of novel deterministic sampling approaches that circumvent the need to compute analytical derivatives, such as the Jacobians of the DSSMs needed by the EKF. It turns out that these algorithms, collectively called *derivative-free* or *sigma-point* Kalman filters, are related through the implicit use of statistical linear regression [21] to compute the optimal terms in the Kalman update of (2.17).

Weighted statistical linear regression provides a solution to the problem of linearizing a nonlinear function of a random variable, which takes into account the actual uncertainty (i.e. probabilistic spread) of the random variable. This leads to linearizing the nonlinear function through a linear re-

gression between *multiple* points drawn from the prior distribution of the random variable, instead of linearizing the function through a truncated Taylor-series expansion at a *single* point which is usually taken from the mean value. Since this statistical approximation technique takes into account the statistical properties of the prior random variable, the resulting expected linearization error tends to be smaller than that of a first-order Taylor-series linearization.

The unscented Kalman filter (UKF) [13],[14] is a sigma-point Kalman filter that derives the location of the sigma-points as well as their corresponding weights, and is a new MMSE state estimator for a nonlinear DSSM. Basic to formulating the UKF is the unscented transformation [13]. The UKF formalism chooses $2L+1$ sigma (or regression) points χ_i in state space with their associated weights $W_i (i = 1, \ldots, L)$, where L is the dimension of the state $\mathbf{x}$,

$$\chi_i = \begin{cases} \hat{\mathbf{x}}[k], & i = 0 \\ \hat{\mathbf{x}}[k] + (\sqrt{(L+\lambda)\mathbf{P}[k]})_i, & i = 1, \ldots, L \\ \hat{\mathbf{x}}[k] - (\sqrt{(L+\lambda)\mathbf{P}[k]})_{i-L}, & i = L+1, \ldots, 2L \end{cases} \tag{2.39}$$

$$W_i = \begin{cases} \frac{\lambda}{L+\lambda}, & i = 0 \\ \frac{1}{2(L+\lambda)}, & i = 1, 2, \ldots, 2L \end{cases} \tag{2.40}$$

where $(\sqrt{(L+\lambda)\mathbf{P}[k]})_i$ is the ith column of the matrix square root, e.g. lower-triangular Cholesky factorization, and λ is a degree of freedom in the choice of the sigma points χ_i. The sample set is chosen to have the same mean and covariance as the distribution of $\mathbf{x}[k]$

$$\hat{\mathbf{x}}[k] = \sum_{i=0}^{2L} W_i \chi_i[k] \tag{2.41}$$

$$\mathbf{P}[k] = \sum_{i=0}^{2L} W_i (\chi_i[k] - \hat{\mathbf{x}}[k])(\chi_i[k] - \hat{\mathbf{x}}[k])^T. \tag{2.42}$$

A set of the sigma points is represented as

$$\boldsymbol{\chi}[k] = [\hat{\mathbf{x}}[k] \quad \hat{\mathbf{x}}[k] + (\sqrt{(L+\lambda)\mathbf{P}[k]}) \quad \hat{\mathbf{x}}[k] - (\sqrt{(L+\lambda)\mathbf{P}[k]})]. \tag{2.43}$$

Keeping in mind the nonlinear DSSM of (2.23) and (2.24), we may write the time-update equations, based on the sigma points and accounting for the effects of system dynamics, as follows

[15]:

$$\boldsymbol{\chi}[k|k-1] = \mathbf{f}_k(\boldsymbol{\chi}[k-1]) \tag{2.44}$$

$$\hat{\mathbf{x}}^-[k] = \sum_{i=0}^{2L} W_i \chi_i[k|k-1] \tag{2.45}$$

$$\mathbf{P}^-[k] = \sum_{i=0}^{2L} W_i (\chi_i[k|k-1] - \hat{\mathbf{x}}^-[k])(\chi_i[k|k-1] - \hat{\mathbf{x}}^-[k])^T + \mathbf{Q}[k] \tag{2.46}$$

$$\boldsymbol{\mathcal{Y}}[k|k-1] = \mathbf{g}_k(\boldsymbol{\chi}[k|k-1]) \tag{2.47}$$

$$\hat{\mathbf{y}}^-[k] = \sum_{i=0}^{2L} W_i \mathcal{Y}_i[k|k-1]. \tag{2.48}$$

The measurement-update equations, accounting for the effects of observations, are as follows [15]:

$$\mathbf{P}_{yy}[k] = \sum_{i=0}^{2L} W_i (\mathcal{Y}_i[k|k-1] - \hat{\mathbf{y}}^-[k])(\mathcal{Y}_i[k|k-1] - \hat{\mathbf{y}}^-[k])^T + \mathbf{R}[k] \tag{2.49}$$

$$\mathbf{P}_{xy}[k] = \sum_{i=0}^{2L} W_i (\chi_i[k|k-1] - \hat{\mathbf{x}}^-[k])(\mathcal{Y}_i[k|k-1] - \hat{\mathbf{y}}^-[k])^T \tag{2.50}$$

$$\mathbf{K}[k] = \mathbf{P}_{xy}[k]\mathbf{P}_{yy}^{-1}[k] \tag{2.51}$$

$$\hat{\mathbf{x}}[k] = \hat{\mathbf{x}}^-[k] + \mathbf{K}[k](\mathbf{y}[k] - \hat{\mathbf{y}}^-[k]) \tag{2.52}$$

$$\mathbf{P}[k] = \mathbf{P}^-[k] - \mathbf{K}[k]\mathbf{P}_{yy}[k]\mathbf{K}^T[k] \tag{2.53}$$

where the last three equations follow from the Kalman filter theory. The UKF is a straightforward extension of the unscented transformation, as summarized in Table 2.3.

Restating a point mentioned earlier, the UKF is a derivative-free estimator, in that, by using multiple forward propagations, the need for explicit computation of Jacobians is avoided. Clearly, it is an advantage over the EKF. In general, the computational complexity of the UKF is $\mathcal{O}(L^3)$, but under special conditions (e.g. parameter estimation, including the training of neural networks), the complexity can be reduced to be $\mathcal{O}(L^2)$ [15],[22].

Table 2.3 Description of the unscented Kalman filter.

1. *Initialization*
   At $k = 0$, set:
   $\hat{\mathbf{x}}[0] = \mathcal{E}\{\mathbf{x}[0]\}$,
   $\mathbf{P}[0] = \mathcal{E}\{(\mathbf{x}[0] - \hat{\mathbf{x}}[0])(\mathbf{x}[0] - \hat{\mathbf{x}}[0])^T\}$.
2. *Computation*
   For $k = 1, 2, \ldots$, perform the computations:
   - *Sigma Points*: Equations (2.39) to (2.43).
   - *Time Updates* (Prediction Step): Equations (2.44) to (2.48).
   - *Measurement Updates* (Correction Step): Equations (2.49) to (2.53).

2.3.4 Sequential Monte Carlo Methods

Sequential Monte Carlo (SMC) methods are simulation-based methods which provide a convenient and attractive approach to computing the posterior distributions. SMC methods are flexible, easy to implement, parallelizible and applicable in very general settings. Since the proposal of the so-called bootstrap filter [23],[24] has been developed, the SMC methods have received great attention. Several closely related algorithms have been found in different research fields, under the names of bootstrap filter, condensation, particle filters, Monte Carlo filters, interacting particle approximations and survival of the fittest [16].

Monte Carlo Sampling

Monte Carlo (MC) methods are used to approximate the intractable integrals appearing in (2.10) and (2.11), when one has a large number of samples drawn from the required posterior distributions. The idea is to simulate N independent and identically distributed (i.i.d.) random samples, also named *particles*, $\{\mathbf{x}^i[0:k]\}_{i=1}^N$ according to $p(\mathbf{x}[0:k]|\mathbf{y}[1:k])$. An empirical estimate of the distribution is

$$\hat{p}(\mathbf{x}[0:k]|\mathbf{y}[1:k]) = \frac{1}{N}\sum_{i=1}^{N}\delta\left(\mathbf{x}[0:k] - \mathbf{x}^i[0:k]\right) \tag{2.54}$$

where $\delta(\mathbf{x}[0:k] - \mathbf{x}^i[0:k])$ denotes the Dirac delta function. The expectation of any function $f(\cdot)$ of $\mathbf{x}[0:k]$ can be obtained straightforwardly by the corresponding integrals as follows:

$$\begin{aligned}\hat{E}(f) = \mathcal{E}_{\hat{p}(\mathbf{x}[0:k]|\mathbf{y}[1:k])}\{f(\mathbf{x}[0:k])\} &= \int f(\mathbf{x}[0:k])\hat{p}_N(\mathbf{x}[0:k]|\mathbf{y}[1:k])d\mathbf{x}[0:k] \\ &= \frac{1}{N}\sum_{i=1}^{N} f\left(\mathbf{x}^i[0:k]\right). \end{aligned} \tag{2.55}$$

This estimate is unbiased with variance proportional to $1/N$ for the finite variance of $f(\mathbf{x}[0:k])$. From the strong law of large numbers,

$$\hat{E}(f) \underset{N\to+\infty}{\longrightarrow} E(f) \tag{2.56}$$

where $\longrightarrow$ denotes almost sure convergence. The advantage of this *perfect* MC sampling is clear. From the set of random samples $\{\mathbf{x}^i[0:k]\}_{i=1}^N$, one easily estimates any quantity $E(f)$ and the rate of convergence of this estimate is independent of the dimension of the integrand. Unfortunately, this is usually not the case with the posterior density $p(\mathbf{x}[0:k]|\mathbf{y}[1:k])$ being multivariable, non-standard and typically only known up to a normalizing constant. In applied statistics, Markov chain Monte Carlo (MCMC) methods are a popular approach to sampling from such complex probability distributions. MCMC methods, however, are iterative methods unappropriate to recursive estimation problems. Therefore, alternative approaches have been developed.

Importance Sampling

An alternative solution is to use the *importance sampling*. With introducing another arbitrary probability distribution $q(\mathbf{x}[0:k]|\mathbf{y}[1:k])$, the so-called *importance sampling distribution*[2], the estimate of the function $f(\mathbf{x}[0:k])$ can be represented as

$$E(f) = \int f(\mathbf{x}[0:k])\frac{p(\mathbf{x}[0:k]|\mathbf{y}[1:k])}{q(\mathbf{x}[0:k]|\mathbf{y}[1:k])}q(\mathbf{x}[0:k]|\mathbf{y}[1:k])d\mathbf{x}[0:k] \tag{2.57}$$

[2]It is often referred to as the proposal distribution or the importance function.

defining the *importance weight* as:

$$w(\mathbf{x}[0:k]) = \frac{p(\mathbf{x}[0:k]|\mathbf{y}[1:k])}{q(\mathbf{x}[0:k]|\mathbf{y}[1:k])}. \tag{2.58}$$

and applying the Bayes' theorem to (2.57), we get the following

$$\begin{aligned} E(f) &= \int f(\mathbf{x}[0:k]) \frac{p(\mathbf{y}[1:k]|\mathbf{x}[0:k])p(\mathbf{x}[0:k])}{p(\mathbf{y}[1:k])q(\mathbf{x}[0:k]|\mathbf{y}[1:k])} q(\mathbf{x}[0:k]|\mathbf{y}[1:k])d\mathbf{x}[0:k] \\ &= \frac{1}{p(\mathbf{y}[1:k])} \int f(\mathbf{x}[0:k])w(k)q(\mathbf{x}[0:k]|\mathbf{y}[1:k])d\mathbf{x}[0:k] \\ &= \frac{\int f(\mathbf{x}[0:k])w(k)q(\mathbf{x}[0:k]|\mathbf{y}[1:k])d\mathbf{x}[0:k]}{\int p(\mathbf{y}[1:k]|\mathbf{x}[0:k])p(\mathbf{x}[0:k]) \frac{q(\mathbf{x}[0:k]|\mathbf{y}[1:k])}{q(\mathbf{x}[0:k]|\mathbf{y}[1:k])} d\mathbf{x}[0:k]} \\ &= \frac{\int f(\mathbf{x}[0:k])w(k)q(\mathbf{x}[0:k]|\mathbf{y}[1:k])d\mathbf{x}[0:k]}{\int w(k)q(\mathbf{x}[0:k]|\mathbf{y}[1:k])d\mathbf{x}[0:k]} \end{aligned} \tag{2.59}$$

where $w(k)$ is called the unnormalized importance weights

$$w(k) = \frac{p(\mathbf{y}[1:k]|\mathbf{x}[0:k])p(\mathbf{x}[0:k])}{q(\mathbf{x}[0:k]|\mathbf{y}[1:k])}. \tag{2.60}$$

By simulating N i.i.d. samples $\{\mathbf{x}^i[0:k]\}_{i=1}^N$ according to $q(\mathbf{x}[0:k]|\mathbf{y}[1:k])$, the estimate of $E(f)$ can be approximated by

$$\hat{E}(f) = \frac{\frac{1}{N}\sum_{i=1}^N f(\mathbf{x}^i[0:k])w^i[k]}{\frac{1}{N}\sum_{i=1}^N w^i[k]} = \sum_{i=1}^N f(\mathbf{x}^i[0:k])\tilde{w}^i[k] \tag{2.61}$$

where $\mathbf{x}^i[0:k]$ denotes samples (or particles) drawn from $q(\mathbf{x}[0:k]|\mathbf{y}[1:k])$ and $\tilde{w}^i[k]$ is the *normalized importance weights*

$$\tilde{w}^i[k] = \frac{w^i[k]}{\sum_{j=1}^N w^j[k]}, \quad i = 1, 2, \ldots, N. \tag{2.62}$$

For N finite, the estimate $\hat{E}(f)$ is biased because it involves a ratio of two estimates. However, under weak assumptions, the strong law of large numbers in (2.56) applies. Under the additional assumptions, a central limit theorem with a convergence rate still independent of the dimension and the integrand holds [25]. It means clearly that the integration method can be interpreted as

a simulation-based method for sampling, where the estimate of the posterior distribution $p(\mathbf{x}[0:k]|\mathbf{y}[1:k])$ is approximated by

$$\hat{p}(\mathbf{x}[0:k]|\mathbf{y}[1:k]) = \sum_{i=1}^{N} \tilde{w}^i[k]\delta\left(\mathbf{x}[0:k] - \mathbf{x}^i[0:k]\right). \tag{2.63}$$

The perfect MC sampling corresponds to $\tilde{w}^i[k] = 1/N$ and $q(\mathbf{x}[0:k]|\mathbf{y}[1:k]) = p(\mathbf{x}[0:k]|\mathbf{y}[1:k])$.

Sequential Importance Sampling

Importance sampling is a general MC integration method, but it is a batch method inadequate for recursive estimation. To obtain the estimate of $p(\mathbf{x}[0:k]|\mathbf{y}[1:k])$ sequentially, one needs to propagate $\hat{p}(\mathbf{x}[0:k]|\mathbf{y}[1:k])$ in time, without modifying the past simulated states $\{\mathbf{x}^i[0:k]\}_{i=1}^N$. This means that, admitting $q(\mathbf{x}[0:k-1]|\mathbf{y}[1:k-1])$ as the marginal distribution at time $k-1$, the importance function can be expanded as

$$\begin{aligned} q(\mathbf{x}[0:k]|\mathbf{y}[1:k]) &= q(\mathbf{x}[0:k-1]|\mathbf{y}[1:k-1])q(\mathbf{x}[k]|\mathbf{x}[0:k-1],\mathbf{y}[1:k]) \\ &= q(\mathbf{x}[0])\prod_{i=1}^{k} q(\mathbf{x}[i]|\mathbf{x}[0:i-1],\mathbf{y}[1:i]). \end{aligned} \tag{2.64}$$

We can see that a recursive expression for the importance weights can be derived by using the above equations. Indeed, we get the modified weights as follows

$$w^i[k] \propto w^i[k-1]\frac{p(\mathbf{y}[k]|\mathbf{x}^i[k])\,p(\mathbf{x}^i[k]|\mathbf{x}^i[k-1])}{q\left(\mathbf{x}^i[k]|\mathbf{x}^i[0:k-1],\mathbf{y}[1:k]\right)}, \quad i = 1,2,\ldots,N \tag{2.65}$$

which underlies the basic principle of the sequential importance sampling (SIS) particle filter. The posterior filtering density $p(\mathbf{x}[k]|\mathbf{y}[1:k])$ can be approximated by

$$\hat{p}(\mathbf{x}[k]|\mathbf{y}[1:k]) = \sum_{i=1}^{N} \tilde{w}^i[k]\delta\left(\mathbf{x}[k] - \mathbf{x}^i[k]\right). \tag{2.66}$$

This point-mass estimate can approximate any general distribution arbitrarily well, limited only by the number of particles used and how well the above mentioned importance sampling conditions

are met. This SIS algorithm provides a foundation of particle filtering based on MC simulation.

The computational complexity of the SIS algorithm is $\mathcal{O}(N)$, with its greatest advantage being its parallel implementation. The memory requirement is $\mathcal{O}((k+1)N)$, as it is necessary to keep all the simulated trajectories. However, when we only need to estimate the filtering distribution $p(\mathbf{x}[k]|\mathbf{y}[1:k]$ and when the chosen importance distribution is not dependent on the past samples $\{\mathbf{x}^i[0:k-2]\}_{i=1}^N$, the memory requirement becomes $\mathcal{O}(N)$.

Particle Filtering

The SIS particle filter discussed so far has a serious limitation: the variance of the importance weights increases stochastically as k increases. Typically, after a few iterations, only one particle has a non-zero importance weight, while the remaining weights tend to zero [26]. A large number of samples are therefore effectively removed from the sample set since their importance weights become numerically insignificant. This phenomenon is often referred to as *weight degeneracy* or *sample impoverishment*. It is not possible to avoid this degeneracy phenomenon. To monitor the degeneracy, a suggested measure, *effective sample size*, was introduced [25],[26]. To reduce degeneracy problem's undesirable effect, the brutal force approach is to use a very large N. However, this is often impractical. Other methods, such as a good choice of importance density [27] and the use of resampling, have been proposed to remedy the problem.

A resampling (or selection) stage may be used to eliminate samples with low importance weights and multiply samples with high importance weights [28]. Various authors have described efficient algorithms for accomplishing this task in $\mathcal{O}(N)$ operations [27],[29]. However, because resampling replaces the particles with high importance weights with many replicates, it also introduces certain correlations within them, in particular when there are only a few dominant weights. The problem is known as the *loss of diversity*. To improve this problem, Markov chain Monte Carlo (MCMC) techniques are helpful [30]. The implementation of a generic particle filter is summarized in Table 2.4. More details for particle filtering algorithms can be referred to [16],[31] and [32].

Table 2.4 A generic particle filtering algorithm.

1. *Initialization*
   At $k = 0$
   - For $i = 1, \ldots, N$,
     - Draw samples (or particles) $\mathbf{x}^i[0] \sim p(\mathbf{x}[0])$.
     - Set $w^i[0] = 1$ and $k = 1$.
2. *Importance Sampling Step*
   For $k = 1, 2, \ldots$
   - For $i = 1, \ldots, N$,
     - Sample $\mathbf{x}^i[k] \sim q(\mathbf{x}[k]|\mathbf{x}^i[0 : k-1], \mathbf{y}[1 : k])$.
     - Set $\mathbf{x}^i[0 : k] = (\mathbf{x}^i[0 : k-1], \mathbf{x}^i[k])$.
   - For $i = 1, \ldots, N$,
     - Evaluate the importance weights $w^i[k]$ according to (2.65).
   - For $i = 1, \ldots, N$,
     - Normalize the importance weights $\tilde{w}^i[k]$ according to (2.62).
3. *Resampling (Selection) Step*
   - Multiply/suppress samples $\mathbf{x}^i[k]$ with high/low importance weights $\tilde{w}^i[k]$, respectively, to obtain N random samples approximately distributed according to $p(\mathbf{x}[k]|\mathbf{y}[1 : k])$.
   - For $i = 1, \ldots, N$,
     - Set $w^i[k] = \tilde{w}^i[k] = 1/N$.
   - (Optional) Do a single MCMC move-step to add further 'variety' to the particle set without changing their distribution.
4. *Output*
   - The output of the algorithm is a set of samples that can be used to approximate the posterior distribution as in (2.66).
   - From these samples, any estimate of the system state can be calculated, such as the MMSE estimate,
     $\hat{\mathbf{x}}[k] \approx \sum_{i=1}^{N} \tilde{w}^i[k]\mathbf{x}^i[k]$.
   - Set $k = k + 1$ and go to Step 2.

Chapter 3

Channel Equalization, Estimation, And Non-Gaussian Noise

In this chapter, we review the literature and provide an overview of channel models, equalization, and estimation as well as non-Gaussian noise in communication systems. In Section 3.1 we present the relevant channel models in terms of a finite impulse response (FIR) filter, including time-invariant and time-varying models. Multiple-input multiple-output (MIMO) channel models are also briefly discussed. In Section 3.2 we first classify channel equalization methods and an overview for typical equalization schemes are given, such as linear, nonlinear, and neural network equalizers. Training algorithms used in equalization are also discussed. In the last subsection turbo equalization, which is receiving great attention nowadays, is covered. Channel estimation techniques are described in Section 3.3. The main topics covered are training-based channel estimation and semiblind channel estimation based on both snapshot approaches and adaptive filtering-based approaches. Iterative channel estimation methods combined with iterative (turbo) receivers are included. Channel estimation in MIMO wireless systems is reviewed. In particular, MIMO channel tracking using Bayesian filtering is introduced. In Section 3.4, we deal with two non-Gaussian noise models: Gaussian mixture models and α-stable models, which are commonly used as ambient noise models in communication systems.

3.1 Communication Channels

Practical channel models are used to characterize the wireline and wireless communications. Let us consider a baseband description of a wireless communication system. The symbol sequence $s[i]$ is filtered by the transmit filter $g_{\mathrm{TX}}(t)$, distorted by the physical channel $g_{\mathrm{CH}}(t;\tau)$, corrupted by additive noise $n(t)$, and finally filtered by the receive filter $g_{\mathrm{RX}}(t)$. The received baseband signal $r(t)$ can then be written as

$$r(t) = \sum_{i=-\infty}^{\infty} g(t; t - iT_s)s[i] + n(t) \tag{3.1}$$

where T_s is the symbol period, and $n(t) = g_{\mathrm{RX}}(t) * z(t)$ where $z(t)$ represents the additive white Gaussian noise (AWGN) signal, and $g(t;\tau)$ is the composite channel given by [33]

$$g(t;\tau) := \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g_{\mathrm{RX}}(x)g_{\mathrm{TX}}(\tau - \theta - x)g_{\mathrm{CH}}(t - x;\theta)dxd\theta. \tag{3.2}$$

Symbol rate sampling, i.e., sampling at rate $1/T_s$, is an option, but when the channel bandwidth is larger than $1/(2T_s)$, the rate $1/T_s$ is lower than the Nyquist rate. This causes aliasing, which deteriorates the performance. Fractional sampling, i.e., rate T_f/T_s with $T_f > 1$, can solve this problem [34]. However, the performance will not increase much when increasing T_f beyond $T_f = 2$, because the channel bandwidth is never significantly larger than $1/T_s$.

If the continuous-time signal $r(t)$ is sampled once every T_s section, i.e., baud rate sampling, we obtain the discrete-time sequence

$$r[k] := r(t)|_{t=kT_s} = \sum_{i=-\infty}^{\infty} s[i]h[k; k-i] + n[k] = \sum_{i=-\infty}^{\infty} h[k;i]s[k-i] + n[k] \tag{3.3}$$

where $h[k;i]$ is the effective channel response at time k to a unit input at time $k - i$. For a causal system, $h[k;i] = 0$ for $i < 0(\forall k)$.

3.1.1 Time-Invariant Filter Channel Model

The channel model in (3.2) has a complex structure and contains a large number of parameters, which cause a major problem when trying to estimate the channel. Therefore, we look at a limited time window $t \in (0, KT_s)$, which corresponds to $k \in \{0, 1, \ldots, K-1\}$. A channel is said to be time-invariant if the channel time variation over KT_s is negligible. Assuming that each composite channel satisfies $g(t;\tau) = 0$ for $\tau \notin [0, LT_s]$, each time-invariant channel can be modeled by

$$h[k;\nu] = \sum_{i=0}^{L-1} \delta[\nu - i]h_i \tag{3.4}$$

which is the most common channel model, called the finite impulse response (FIR) filter channel. The FIR filter channel, which is a practical channel model, is well structured and contains a small number of parameters. As shown in Fig. 3.1, when sampled at the baud rate, the channel model can be represented as

$$r[k] = \sum_{i=0}^{L-1} h_i s[k-i] + n[k] = \sum_{i=0}^{L-1} h[i]s[k-i] + n[k] \tag{3.5}$$

where $h_i = h[i] = h[0;i]$ is the time-invariant channel response with length L, and $\{s[k]\}$ is the sequence of transmitted symbols.

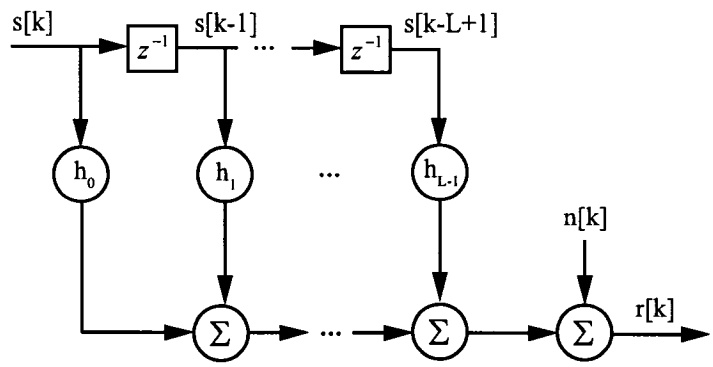


Fig. 3.1 FIR filter channel model with ISI.

3.1.2 Time-Varying Channels

In order to describe the dynamics of time-varying channels, we may introduce the statistical characteristics of the channel. The autocorrelation properties of each independent path are governed by [35]

$$R[\tau] = \mathcal{E}\{h[k]h^*[k-\tau]\} = J_0(2\pi f_D \tau) \tag{3.6}$$

where J_0 is the zero-order Bessel function of the first kind, τ is the time difference, and f_D denotes the Doppler frequency representing the relative motion between the transmitter and the receiver. It shows how fast the channel relies on the carrier frequency (f_c) and mobile speed (v), illustrated by $f_D = (vf_c)/c$ where c is the speed of light. The dynamics of a wide-sense stationary (WSS) channel are modeled as an autoregressive (AR) model of order p. A third-order AR process can accurately model the channel dynamics [36]. However, information-theoretic results have demonstrated that a first-order Markovian approximation turns out to be sufficient to model the Rayleigh narrowband time-varying channel [37].

A channel is said to be time-varying if the channel time variation over KT_s is not negligible. Assuming that each composite channel satisfies $g(t;\tau) = 0$ for $\tau \notin [0, LT_s]$, the time-varying FIR input-output relationship can be written as

$$r[k] = \sum_{i=0}^{L-1} h_i[k]s[k-i] + n[k]. \tag{3.7}$$

Suppose that $h[k;i] = h[k]\delta[i-0]$ where $\delta[i-0]$ is the Kronecker delta located at 0, i.e., $\delta[i-0] = 1$ for $i = 0$ and $\delta[i-0] = 0$ for $i \neq 0$. Then we have a time-selective and frequency-nonselective channel model whose output is given by

$$r[k] = h[k]s[k] + n[k]. \tag{3.8}$$

A time-nonselective and frequency-nonselective channel is modeled as

$$r[k] = hs[k] + n[k] \tag{3.9}$$

where h is a random variable or a constant. Time-varying channel coefficients can be modeled by

means of a second-order Markov model [38] and a basis extension model (BEM) [39].

3.1.3 MIMO Channel Models

We briefly describe sampled signal models of multiple-input and multiple-output (MIMO) links. A comprehensive overview of MIMO channel models can be referred to [40] and references therein.

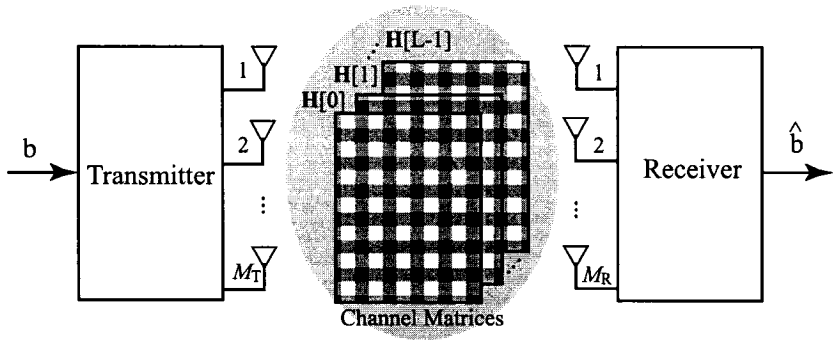


Fig. 3.2 MIMO wireless link.

Frequency-Flat Channel

Consider a MIMO setup with M_T transmit and M_R receive antennas. A simplified MIMO wireless link may be depicted as in Fig. 3.2. We assume that the channel is modeled as a flat-fading Rayleigh random process and the fading is to be uncorrelated across antennas. The received signal vector $\mathbf{r} \in \mathbb{C}^{M_R \times 1}$ on the receiver at symbol time k can be

$$\mathbf{r}[k] = \sqrt{\frac{E_s}{M_T}}\mathbf{H}\mathbf{s}[k] + \mathbf{n}[k] \tag{3.10}$$

where E_s is the symbol energy, $\mathbf{H} \in \mathbb{C}^{M_R \times M_T}$ denotes the channel matrix, $\mathbf{s} \in \mathbb{C}^{M_T \times 1}$ is the symbol vector simultaneously transmitted by the M_T transmit antennas, and $\mathbf{n} \in \mathbb{C}^{M_T \times 1}$ denotes the spatio-temporally white zero-mean circularly symmetric complex Gaussian (ZMCSCG) noise vector with variance N_0 in each dimension.

Frequency-Selective Channel

Let the $M_R \times M_T$ matrix $\mathbf{H}[l]$ $(l = 0, 1, \ldots, L-1)$ denote the channel, where L is the maximum channel length of all the $M_R M_T$ single-path links as in Fig. 3.2. The received vector $\mathbf{r}$ of dimension $M_R \times 1$ may be expressed as

$$\begin{aligned}\mathbf{r}[k] &= \sqrt{\frac{E_s}{M_T}} \begin{bmatrix} \mathbf{h}_{1,1} & \cdots & \mathbf{h}_{1,M_T} \\ \vdots & \ddots & \vdots \\ \mathbf{h}_{M_R,1} & \cdots & \mathbf{h}_{M_R,M_T} \end{bmatrix} \begin{bmatrix} \mathbf{s}_1[k] \\ \vdots \\ \mathbf{s}_{M_T}[k] \end{bmatrix} + \mathbf{n}[k] \\ &= \sqrt{\frac{E_s}{M_T}}\, \mathcal{H} \begin{bmatrix} \mathbf{s}_1[k] \\ \vdots \\ \mathbf{s}_{M_T}[k] \end{bmatrix} + \mathbf{n}[k] \end{aligned} \tag{3.11}$$

where $\mathbf{h}_{i,j} = [h_{i,j}[L-1] \cdots h_{i,j}[0]]$, $\mathbf{s}_j = [s_j[k-L+1] \cdots s_j[k]]^H$. T consecutive received vector samples can be stated as follows:

$$[\mathbf{r}[k] \cdots \mathbf{r}[k+T-1]] = \sqrt{\frac{E_s}{M_T}}\, \mathcal{H} \begin{bmatrix} \mathcal{S}_1 \\ \vdots \\ \mathcal{S}_{M_T} \end{bmatrix} + \mathcal{V} \tag{3.12}$$

where $\mathcal{S}_j (j = 1, \ldots, M_T)$ and $\mathcal{V}$ are defined as the following, respectively

$$\mathcal{S}_j = \begin{bmatrix} s_j[k-L+1] & s_j[k-L+2] & \cdots & s_j[k-L+T] \\ \vdots & & \ddots & \vdots \\ s_j[k-1] & s_j[k] & \cdots & s_j[k+T-2] \\ s_j[k] & s_j[k+1] & \cdots & s_j[k+T-1] \end{bmatrix},$$

$$\mathcal{V} = \begin{bmatrix} n_1[k] & n_1[k+1] & \cdots & n_1[k+T-1] \\ n_2[k] & n_2[k+1] & \cdots & n_2[k+T-1] \\ \vdots & & \ddots & \vdots \\ n_{M_R}[k] & n_{M_R}[k+1] & \cdots & n_{M_R}[k+T-1] \end{bmatrix}.$$

3.2 Channel Equalization

Thanks to the distortive characteristic of the propagation environment, transmitted data symbols spread out in time and interfere with each other. This is called an intersymbol interference (ISI) phenomenon. The degree of ISI relies on the data rate: the higher the data rate, the more ISI is introduced. Moreover, mobility in wireless communications introduces channel time variation, which is harmful. Channel equalization, mitigating these fading effects, is a main challenge in current and future communication systems.

The term *equalization* is used to describe the deconvolution process in communication systems [41]. The idea behind equalizers is to automatically reverse the effects of the channel, thereby recovering the original signal [42]. To achieve reliable communications that satisfy the system requirements such as the desired data rates, transmitter power, and bandwidth constraints, the communication system engineer designs modulation/demodulation and encoding/decoding techniques. In fact, the system designer may avoid the need for channel equalization by selecting $T_s \gg T_m$ where T_s is the time duration of the transmitted signaling waveforms and T_m is the channel multipath spread. If the system designer selects the symbol time duration such that $T_s < T_m$, ISI involved in the received signal should be mitigated by the use of an equalizer. Equalizer design must balance ISI mitigation with noise enhancement, because both the signal and the noise pass through the equalizer, which can enhance the noise power.

Fig. 3.3 illustrates a simple analog equalizer. Let $s(t)$ be a signal that is passed through a channel with frequency response $H(f)$. At the receiver front end, white Gaussian noise $n(t)$ is added to the signal. Therefore, the signal input to the receiver is

$$R(f) = S(f)H(f) + N(f) \tag{3.13}$$

where $N(f)$ has power spectral density N_0. If the bandwidth of $s(t)$ is B, then the noise power within the signal bandwidth of interest is N_0B. The ISI introduced by the channel is removed by an analog equalizer in the receiver defined by

$$H_{\text{eq}}(f) = \frac{1}{H(f)}. \tag{3.14}$$

Note that the equalizer inverts the channel frequency response. The equalized signal $Y(f)$ after passing through the equalizer becomes

$$Y(f) = [S(f)H(f) + N(f)]H_{eq}(f) = S(f) + N'(f) \tag{3.15}$$

where $N'(f)$ is colored Gaussian noise with power spectral density $N_0/|H(f)|^2$. Thus, all ISI has been removed from the transmitted signal $S(f)$. However, the equalizer in (3.14) will greatly enhance the noise power at some frequencies, if at some frequencies $H(f)$ has a spectral null ($H(f) = 0$) or $H(f)$ is greatly attenuated. In this situation the equalized system will perform poorly due to its greatly reduced signal-to-noise ratio (SNR), albeit the ISI effects are removed. Therefore, the goal of equalization must be balanced so that in the process of removing ISI, the SNR of the post-equalization signal is maximized [43]. In general, linear equalizers greatly enhance the noise power because they invert the channel frequency response. Nonlinear equalizers suffer less from noise enhancement than linear equalizers, because they do not invert the channel frequency response. But, nonlinear equalizers typically entail a higher complexity.

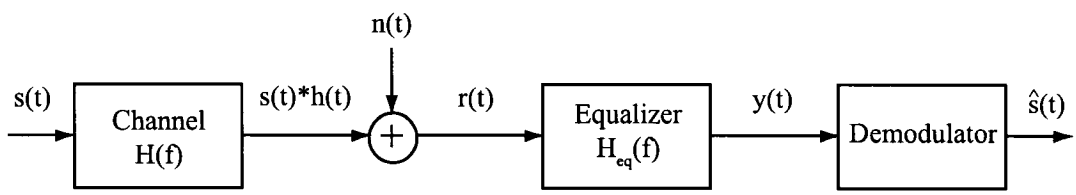


Fig. 3.3 Channel equalization.

3.2.1 Equalizer Classification

Channel equalization has been extensively studied in the literature. A comprehensive overview of equalization techniques can be found in [44],[45],[46],[47],[48] and references therein. Classification of equalization techniques in types, structures and algorithms is illustrated in Fig. 3.4. Equalization techniques can be classified into supervised (or training-based) and unsupervised (or blind) techniques. In training-based equalization, a training (or pilot) sequence is available periodically during the transmission of information. Training-based equalizers use the training sequence

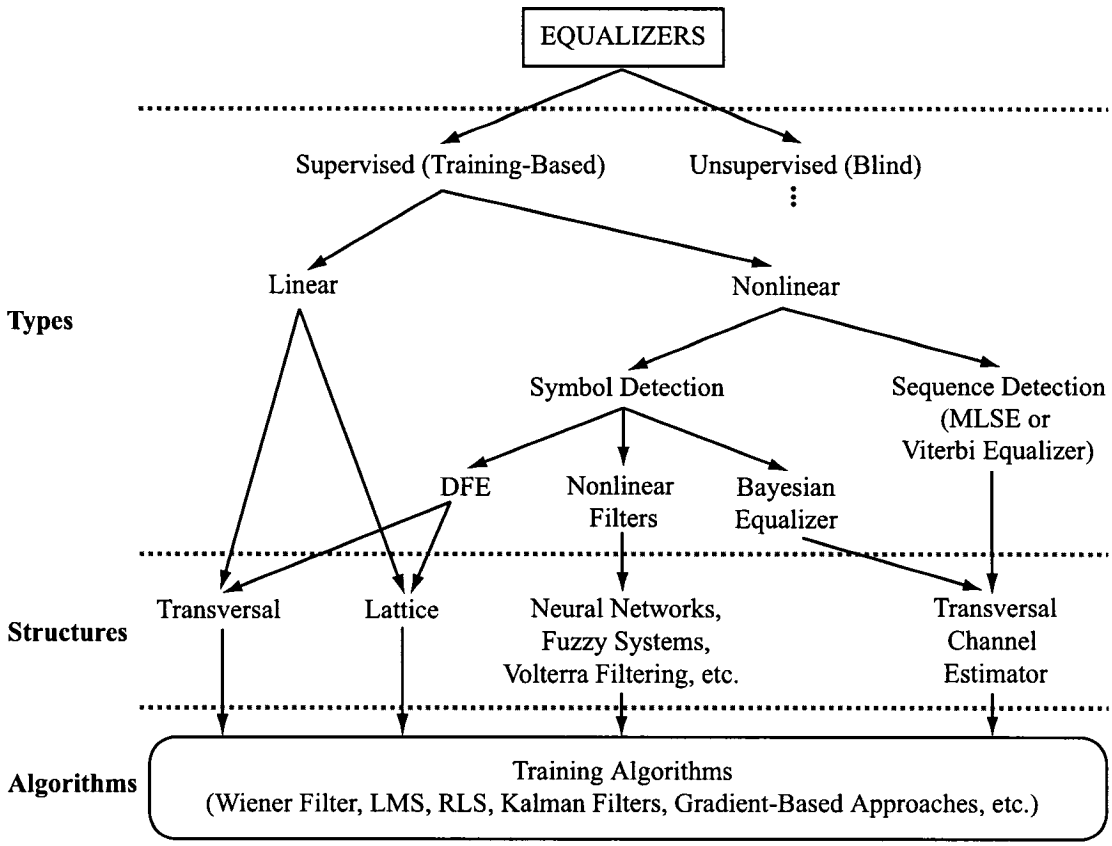


Fig. 3.4 Equalizer classification: types, structures and algorithms.

to estimate channel states or to optimize their weights during the training stage. However, in some communication systems, the transmission of training sequences is either impractical or too costly. In this situation blind equalization techniques are suitable. Even though blind equalization goes beyond the scope of this thesis, the indepth descriptions of blind equalization techniques can be found in [47],[48],[49] and [50].

Training-based techniques fall roughly into two categories: linear and nonlinear. Associated with each type of equalizer is one or more structures for implementing the equalizer. There is a class of algorithms for each structure, that may be employed to adaptively adjust the equalizer parameters according to some specified performance criterion. Linear equalizers are generally the simplest to implement and to understand conceptually, and they are commonly used in applications where the channel distortion is not too severe. However, linear equalizers do not perform well on channels with spectral nulls in their frequency response or with nonlinear distortion. Furthermore, linear equalization techniques typically suffer from more noise enhancement than nonlinear equalization techniques, and are therefore avoided for fading multipath channels. Instead, nonlinear equalization methods are used.

There are two types of nonlinear equalization techniques: symbol detection and sequence detection. Symbol detection methods remove ISI from each symbol and then detect each symbol individually, whereas sequence detection methods detect sequences of symbols where the effect of ISI is part of the estimation process. In Fig. 3.4, all linear equalizers as well as most nonlinear equalizers belong to the symbol detection category. There are several types of nonlinear equalization methods. These include decision-feedback equalization, nonlinear filter-based equalization, Bayesian equalization, and maximum-likelihood sequence estimation (MLSE). Based on perfect channel knowledge, MLSE is the optimal form of sequence detection for the AWGN channel, in the sense that it minimizes the probability of a sequence error [44]. MLSE, so-called Viterbi equalizer, is efficiently implemented by means of the Viterbi algorithm [44],[2] based on Euclidean distance metrics. However, the computational complexity of MLSE grows exponentially with the span of ISI.

On the other hand, as a finite memory equalizer, symbol detection methods are relatively simple and computationally less complex than MLSE, and use a fixed number of input samples to detect the transmitted symbol. The optimum finite memory equalizer is the Bayesian equalizer (BE). The

optimum decision function of the BE is given by the maximum *a posteriori* probability (MAP) criterion, which can be derived from Bayes' theory [51]. A finite memory BE can provide a performance comparable to the MLSE, but with a reduced computational complexity [52].[1] The BE (hereafter BE denotes the finite memory BE) provides the lower performance bound for symbol detection equalizers in terms of probability of error or bit error rate (BER) and can be implemented with linear or nonlinear systems.

Among symbol detection techniques, the decision-feedback equalizer (DFE) is the most common, since it is a computationally efficient suboptimum alternative and generally performs well. The computational complexity of the DFE is a linear function of the number of filter taps. However, on channels with low SNR, the DFE suffers from error propagation when bits are decoded in error, leading to poor performance. Both the DFE and linear equalizers may be implemented either in a transversal or a lattice structure [44],[12],[53],[54]. Nonlinear filter-based equalizers have been successfully developed, based on artificial neural networks [55],[56],[57],[58], fuzzy adaptive filters [59],[60] and Volterra filters [61],[62],[63]. Nonlinear filter-based equalizers have their own structures, dependent on inherent filter structures. They can also be realized in the form either with decision feedback or without decision feedback. These nonlinear equalizers treat equalization as a pattern classification process where the equalizer attempts to classify the input vector into a number of transmitted symbols.

The process of equalizer training and tracking is often referred to as adaptive equalization, since the equalizer adapts to the changing channel. Adaptive equalizers require algorithms for updating the filter tap coefficients and estimating channel state information during training and tracking. Both linear and symbol detection equalizers (except for BE) usually use training algorithms to optimize the equalizer weights, while the BE and MLSE use them for channel estimation.

3.2.2 Linear Equalizers

Linear equalizers can be realized in the two forms: transversal filter and lattice structure. In the following, we describe the linear transversal filter, followed by two linear equalizers: the zero-forcing (ZF) equalizer and the minimum mean square error (MMSE) equalizer.

[1] An infinite memory BE can provide a performance better than the MLSE, but it is highly complex.

Linear Transversal Filter

The simplest traditional equalizer is based on the linear transversal (tapped-delay line) filter. Assuming a linear equalizer implemented with an M tap transversal filter, the linear transversal equalizer (LTE) is

$$H_{\text{eq}}(z) = \sum_{i=0}^{M-1} w_i z^{-i} \tag{3.16}$$

where $H_{\text{eq}}(z)$ and w_i are the z transform and the tap weights of the equalizer, respectively. Causal linear equalizers have $w_i = 0$, $i < 0$. The LTE receives a tapped-delayed input vector $\mathbf{r}[k] = [r[k], r[k-1], \cdots, r[k-M+1]]^T$, composed of the symbols corrupted by noise, at the receiver, and provides the output $y[k]$ generated by the weighted summation of input vector $\mathbf{r}[k]$ with tap weight vector $\mathbf{w}$. The output of the LTE is computed per symbol and can be represented by

$$y[k] = \sum_{i=0}^{M-1} w_i r[k-i]. \tag{3.17}$$

The equalizer design must specify the tap weight vector $\mathbf{w}$ for a given channel frequency response, and the algorithm for updating $\mathbf{w}$ as the channel varies. The weight vector $\mathbf{w}$ is optimized for some performance criteria. In the next section, we describe two solutions widely used in optimizing the tap weights.

Zero-Forcing (ZF) Equalizer

The received samples $\{r[k]\}$, the inputs to the equalizer, can be represented based on the discretized combined system response $f(t) = h(t) * g^*(-t)$ as

$$R(z) = S(z)F(z) + N_g(z) \tag{3.18}$$

where $N_g(z)$ is the power spectrum of the white noise after passing through the matched filter $G^*(1/z^*)$ and

$$F(z) = H(z)G^*(1/z^*) = \sum_k f(kT_s)z^{-k} \tag{3.19}$$

where T_s is the modulation symbol time. The simplest way to remove ISI is to select the equalizer as follows

$$H_{\mathrm{ZF}}(z) = \frac{1}{F(z)} \tag{3.20}$$

where $H_{\mathrm{ZF}}(z)$ is a zero-forcing (ZF) filter. The ZF equalizer removes all ISI introduced in the combined response $f(t)$. However, the ZF equalizer suffers from the noise enhancement properties. If the channel $H(z)$ is sharply attenuated at any frequency within the bandwidth of interest, as is common on frequency-selective fading channels, the noise power will be significantly increased. It can be explained from the noise power spectrum $N(z)$, given by

$$N(z) = N_g(z)|H_{\mathrm{ZF}}(z)|^2 = \frac{N_0|G_m^*(1/z^*)|^2}{|F(z)|^2} = \frac{N_0}{|H(z)|^2}. \tag{3.21}$$

Minimum Mean Square Error (MMSE) Equalizer

Although the ZF equalizer removes ISI, it may not give the best error performance for a communication system because it does not take into account noise enhancement. This motivates an equalizer design to get better optimization between ISI mitigation and noise enhancement. One such equalizer is the minimum mean square error (MMSE) equalizer. The design goal of the MMSE equalizer is that the weight vector w is chosen to minimize the average mean square error (MSE) between the transmitted symbol $s[k]$ and the equalizer output $y[k]$, given in (3.17), i.e.

$$\mathcal{J} = \mathcal{E}\{|s[k] - y[k]|^2\}. \tag{3.22}$$

Finding the optimal filter coefficients $\{w_i\}$ becomes an optimal linear estimation problem. To enable known techniques for optimal linear estimation, we may consider the equalizer $H_{\mathrm{eq}}(z)$ as the concatenation of two filter components:

$$H_{\mathrm{eq}}(z) = \frac{1}{G_m^*(1/z^*)}\hat{H}_{\mathrm{eq}}(z) \tag{3.23}$$

where $\frac{1}{G_m^*(1/z^*)}$ is the noise whitening filter and $\hat{H}_{\mathrm{eq}}(z)$ is the ISI removal filter. Since the noise power spectrum at the receiver is $N_0|G_m^*(1/z^*)|^2$, the noise power spectrum at the output of noise

whitening filter is then

$$\frac{N_0|G_m^*(1/z^*)|^2}{|G_m^*(1/z^*)|^2} = N_0. \tag{3.24}$$

We assume that as shown in (3.16), the filter $\hat{H}_{\text{eq}}(z)$ is the LTE with M tap weight vector $\mathbf{w}$ and with input vector $\mathbf{v}[k]$, which is the output vector of the noise whitening filter. The output of the MMSE equalizer can be obtained as

$$y[k] = \mathbf{w}^T\mathbf{v} = \mathbf{v}^T\mathbf{w}. \tag{3.25}$$

We want to minimize the MSE by suitable choices of $\mathbf{w}$

$$\begin{aligned} \mathcal{J} &= \mathcal{E}\{|s[k] - y[k]|^2\} \\ &= \mathcal{E}\{\mathbf{w}^T\mathbf{v}\mathbf{v}^H\mathbf{w}^* - 2Re\{\mathbf{v}^H\mathbf{w}^*s[k]\} + |s[k]|^2\}. \end{aligned} \tag{3.26}$$

Assuming $\mathcal{E}\{|s[k]|^2\} = 1$, the MSE is given by

$$\mathcal{J} = \mathbf{w}^T\mathbf{R}_{\mathbf{v}}\mathbf{w}^* - 2Re\{\mathbf{v}_{\text{s}}\mathbf{w}^*\} + 1 \tag{3.27}$$

where $\mathbf{R}_{\mathbf{v}} = \mathcal{E}\{\mathbf{v}\mathbf{v}^H\}$ is the $M \times M$ Hermitian matrix and $\mathbf{v}_{\text{s}} = \mathcal{E}\{\mathbf{v}^H s[k]\}$ is a length M row vector. We obtain the optimal tap vector $\mathbf{w}$ by differentiating $\mathcal{J}$ with respect to each tap weight and setting the result to zero. Thus, the gradient $\nabla_{\mathbf{w}}\mathcal{J}$ is

$$\nabla_{\mathbf{w}}\mathcal{J} = \left(\frac{\partial\mathcal{J}}{\partial w_0}, \cdots, \frac{\partial\mathcal{J}}{\partial w_{M-1}}\right) = 2\mathbf{w}^T\mathbf{R}_{\mathbf{v}} - 2\mathbf{v}_{\text{s}} = 0. \tag{3.28}$$

From (3.28), the optimal tap weights are obtained as

$$\mathbf{w}_{\text{opt}} = (\mathbf{R}_{\mathbf{v}}{}^T)^{-1}\mathbf{v}_{\text{s}}^T. \tag{3.29}$$

Applying the $\mathbf{w}_{\text{opt}}$ to (3.27), we obtain the MMSE as

$$\mathcal{J}_{min} = 1 - \mathbf{v}_{\text{s}}\mathbf{R}_{\mathbf{v}}{}^{-1}\mathbf{v}_{\text{s}}^H. \tag{3.30}$$

For an infinite length equalizer, $\mathbf{w}^T\mathbf{R}_v = \mathbf{v}_s$ can be written as [43]

$$\sum_{i=-\infty}^{\infty} w_i(f[j-i] + N_0)\delta[j-i] = g_m^*[-j], \quad -\infty \leq j \leq \infty. \tag{3.31}$$

Taking z transforms and noting that $\hat{H}_{\text{eq}}(z)$ is the z transform of the tap weights w yields

$$\hat{H}_{eq}(z)(F(z) + N_0) = G^*(1/z^*). \tag{3.32}$$

When the equalizer is not restricted to be finite length, we get the MMSE equalizer as

$$H_{\text{MMSE}}(z) = H_{\text{eq}}(z) = \frac{\hat{H}_{\text{eq}}(z)}{G_m^*(1/z^*)} = \frac{1}{F(z) + N_0}. \tag{3.33}$$

Note that we find three interesting things from (3.33) [43]. First, the ideal infinite length MMSE equalizer cancels out the noise whitening filter. Second, the MMSE equalizer is identical to the ZF equalizer in the absence of the noise term N_0. Lastly, the design of this ideal MMSE equalizer clearly shows a balance between inverting the channel and noise enhancement. Further details in the MMSE equalizer can be found in [64] and [65].

3.2.3 Nonlinear Equalizers

Decision Feedback Equalizer (DFE)

A DFE consists of two components: a feedforward filter $W(z)$ and a feedback filter $V(z)$. The basic idea in the DFE is that once an information symbol has been detected, the ISI that it causes on future symbols may be estimated and subtracted out prior to symbol detection [66]. The DFE can be realized either as a transversal or as a lattice structure [67]. Fig. 3.5 displays a structure of the DFE. Employing the linear transversal filters, the DFE output can be expressed as

$$\begin{aligned} y[k] &= y_f[k] - y_b[k] \\ &= \sum_{i=0}^{M_1} w_i r[k-i] - \sum_{i=1}^{M_2} v_i \hat{s}[k-i] \end{aligned} \tag{3.34}$$

where $M_1 + 1$ and M_2 denotes the length of feedforward and feedback tap weights, respectively, and $\{\hat{s}[k-1], \cdots, \hat{s}[k-M_2]\}$ are previously detected symbols. From the structure itself, we can see that the DFE determines the ISI contribution from the detected symbols $\{\hat{s}[k]\}$ by passing them through the feedback filter that estimates the combined discrete equivalent baseband channel $F(z)$. Therefore, the feedback filter does not suffer from noise enhancement. Since the feedback filter sits in the feedback loop, it must be strictly causal, otherwise the system is unstable.

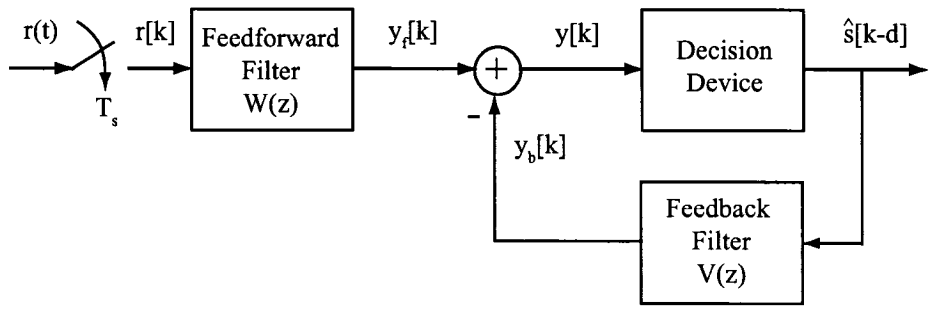


Fig. 3.5 Structure of the DFE.

ZF or MMSE criterion can be applied for selecting the filter coefficients for $W(z)$ and $V(z)$. For channels with deep spectral nulls, the DFEs generally perform much better than linear equalizers. However, the DFEs exhibit feedback errors if $s[k] \neq \hat{s}[k]$. This error, therefore, propagates to later bit decisions. The error propagation can be improved through channel coding, but it also seriously degrades performance on channels with low SNR. A systematic treatment of the DFE with coding can be found in [64] and [68]. Generally, the MMSE associated with the DFE is much lower than that of a linear equalizer, if the impact of feedback error is ignored.

The computational complexity of the DFE is a linear function of the number of taps of the feedforward and feedback filters. The DFE has been shown to be effective for equalizing the ISI in underwater acoustic communication channels [66]. It also provides a computationally efficient alternative to MLSE for use in the Global System for Mobile (GSM) cellular system [69]. We can find in [66] that the DFE has been used in digital communication systems for troposcatter channels operating in the SHF(3-30 GHz) frequency band and ionospheric channels in the HF (2-30 MHz) frequency band.

Bayesian Equalizer (BE)

An LTE is a classifier when used in conjunction with a decision device. The decision boundary of the LTE is a straight line rather than a hyperplane. The straight line divides the space into two regimes. The equalizer can decide incorrectly if the output of the channel is buried in noise. Furthermore, it is impossible to separate the two classes at all by a straight line if the channel is nonminimum phase. These shortcomings of the LTE lead directly to a Bayesian approach [70]. The Bayesian equalizer (BE) is known as the optimal solution in detection theory.

Assume that an equalizer is characterized by the feedforward order M and decision delay d. Under the assumption of binary transmission and AWGN with a variance σ_0^2, the decision boundary is defined as the solution to a conditional probability.

$$P_{s|\mathbf{r}}(\{s[k-d]=+1\}|y[k]) = P_{s|\mathbf{r}}(\{s[k-d]=-1\}|y[k]) \tag{3.35}$$

This equation defines the Bayesian decision function as follows:

$$f(\mathbf{r}[k]) = p_{\mathbf{r}|s}(y[k]|\{s[k-d]=+1\}) - p_{\mathbf{r}|s}(y[k]|\{s[k-d]=-1\}) \tag{3.36}$$

where $p_{\mathbf{r}|s}$ is the condition density function of $y[k]$ given $s[k-d]$. The optimal decision function of the BE [71],[72],[58],[73] is based on Bayes' theorem and is given by

$$\begin{aligned} y[k] &= f(\mathbf{r}[k]) \\ &= \sum_{i=1}^{N_s^+} (2\pi\sigma_0^2)^{-M/2} \exp\left(\frac{-||\mathbf{r}[k]-\mathbf{c}_i^+||^2}{2\sigma_o^2}\right) \\ &\quad - \sum_{j=1}^{N_s^-} (2\pi\sigma_0^2)^{-M/2} \exp\left(\frac{-||\mathbf{r}[k]-\mathbf{c}_j^-||^2}{2\sigma_o^2}\right) \end{aligned} \tag{3.37}$$

where $\mathbf{r}[k]$ denotes the equalizer input vector, σ_0^2 represents the channel noise variance, and $\mathbf{c}_i^+ \in \mathbb{R}^M$ and $\mathbf{c}_i^- \in \mathbb{R}^M$ are the positive and negative channel states, respectively. The terms N_s^+ and N_s^- are the number of positive and negative channel states with equal probability, respectively. Under the assumption of binary transmission, the sign of the decision function and scaling terms in (3.37)

can be ignored. Thus, the decision function can be simplified as

$$y[k] = \sum_{i=1}^{N_s} w_i \exp\left(\frac{-||\mathbf{r}[k] - \mathbf{c}_i||^2}{2\sigma_o^2}\right) \tag{3.38}$$

where N_s is the number of channel states with $N_s^+ = N_s^- = N_s/2$, and w_i are the weight parameters associated with each of the centers and are given by

$$w_i = \begin{cases} +1.0, & \mathbf{c}_i \in N_s^+ \\ -1.0, & \mathbf{c}_i \in N_s^-. \end{cases} \tag{3.39}$$

The estimate of the symbol from the memoryless decision device is

$$\hat{s}[k-d] = \begin{cases} +1.0, & y[k] \geq 0 \\ -1.0, & y[k] < 0. \end{cases} \tag{3.40}$$

The decision function of the BE is nonlinear and is entirely characterized with the known channel states and the noise. That is, performance of the BE relies on knowledge of the channel states. The channel state vector can be obtained from adaptive schemes [74],[75] such as adaptive filtering or clustering techniques.

Maximum Likelihood Sequence Estimation (MLSE)

MLSE avoids the problem of noise enhancement since it doesn't use an equalizer filter: instead it estimates the sequence of transmitted symbols. Given the channel response $h(t)$, the MLSE algorithm selects input sequence $\{s_k\}$ that maximizes the likelihood of the channel output $v(t)$, prior to matched filtering. Based on a Gram-Schmidt orthogonalization procedure, $v(t)$ on a time interval $[0, KT_s]$ can be expressed as

$$v(t) = \sum_{j=1}^{N} v_j \phi_j(t) \tag{3.41}$$

where $\{\phi_j(t)\}$ forms a complete set of orthogonal basis functions. The number N of functions in this set is a function of the channel memory, because $v(t)$ on $[0, KT_s]$ depends on $s_0, \ldots, s_K$.

With this expression we have

$$v_j = \sum_{k=-\infty}^{\infty} s_k h_{jk} + n_j = \sum_{k=0}^{K} s_k h_{jk} + n_j. \tag{3.42}$$

Then, the received signal sequence $\mathbf{v}^N = (v_1, \ldots, v_N)$ has a multivariate Gaussian distribution

$$p(\mathbf{v}^N|\mathbf{s}^K, h(t)) = \prod_{j=1}^{N} \left[\frac{1}{\pi N_0} \exp\left(-\frac{1}{N_0} \left| v_n - \sum_{k=0}^{K} s_k h_{jk} \right|^2 \right) \right]. \tag{3.43}$$

Given the received signals $\mathbf{v}^N$, the MLSE decodes this as the symbol sequence s^K that maximizes the likelihood function $p(\mathbf{v}^N|\mathbf{s}^K, h(t))$ or the log of this function. The MLSE outputs the sequence [43]

$$\begin{aligned} \hat{\mathbf{s}}^L &= \arg\max \left\{ \log p(\mathbf{v}^N|\mathbf{s}^L, h(t)) \right\} \\ &= \arg\max \left[2Re\left\{ \sum_k s_k^* r[k] \right\} - \sum_k \sum_m s_k s_m^* f[k-m] \right] \end{aligned} \tag{3.44}$$

where

$$r[k] = \sum_{j=1}^{N} v_j h_{jk}^* = \int_{-\infty}^{\infty} v(\tau) h^*(\tau - jT_s) d\tau \tag{3.45}$$

$$f[k-m] = \sum_{j=1}^{N} h_{jk} h_{jm}^* = \int_{-\infty}^{\infty} h(\tau - kT_s) h^*(\tau - mT_s) d\tau. \tag{3.46}$$

Note that the MLSE output depends only on the sampler output $\{r[k]\}$ after matched filtering and the channel parameter $f[k-m] = f(kT_s - mT_s)$ where $f(t) = h(t) * h^*(-t)$. The MLSE is optimal for detecting signals in the presence of ISI.

The Viterbi algorithm (VA) can be used in implementing the MLSE to reduce complexity [44],[2]. However, the computational complexity of the MLSE based on the VA still grows exponentially with the channel spread.

3.2.4 Neural Network Equalizers

Neural networks (NNs), inspired from biological neural systems, are well known as adaptive systems that adapt to an unknown environment through a learning process. NNs are a massively parallel, distributed processor, made up of simple processing units, called neurons. NNs can approximate continuous functions when their activation functions are sigmoidal [55],[56]. The universal approximation property of NNs suggests that neural network filters can approximate all filters defined by linear and nonlinear continuous functions. NNs can be viewed as an extension of classical linear adaptive filters to treat nonlinear modeling tasks [76]. NNs can, therefore, give solutions to complicated problems in communication systems due to the nonlinear parallel, distributed processing architecture, the capability of learning and generalization, and efficient hardware implementation.

Overviews of NNs to communication systems and adaptive receiver design have been reported in [57],[77]. Adaptive channel equalizers are used to reduce channel disturbances such as ISI, cochannel interference (CCI), noise, nonlinear distortions, fading, and time-varying characteristics in digital communication systems. Many types of neural network-based equalizers have been applied to communication channels. During the last decade, neural network filters have been successfully applied to adaptive equalization of communication channels because they can model arbitrary nonlinearities. Equalization schemes using neural network filters are divided into two streams: supervised and unsupervised (blind) equalization. The supervised equalizer undergoes learning in a supervised method that a desired response is supplied for the learning process. On the other hand, in the blind equalizer, the neural network filter uses known statistical properties of the received signal, and its learning process performs adaptation in an unsupervised manner. In this thesis, we treat only the supervised neural equalization approaches.

In neural equalization, the linear adaptive filter can be replaced with a neural network filter. A variety of neural network filter architectures have been proposed for channel equalization [58]. In the following subsections, we describe typical neural network models applied to adaptive equalization: multilayer perceptrons (MLPs), radial basis function (RBF) networks, recurrent neural networks (RNNs) and others.

Multilayer Perceptron (MLP) Equalizers

The most commonly used structure among all neural network-based equalizers is the MLP architecture. The popularity of MLP equalizers is due in part to their computational simplicity, finite parameterization, stability and smaller network size for an equalization problem as compared to other architectures [78].

An MLP consists of several hidden layers of neurons that are capable of performing nonlinear mapping between the input and the output layer. The hidden layers supply the capability to create complex nonlinear decision boundaries by using nonlinear activation functions like a sigmoid function. It has been shown that the MLP with only three layers is enough to create these boundaries [79]. Fig. 3.6 shows the architecture of an MLP equalizer with three layers. The basic element of the MLP equalizer is the neuron. Each neuron in the layer has primary local connections and is characterized by a set of weights applied to the previous layer to which it is connected and a bias b. The jth neuron in the pth layer accepts inputs $v_h^{(p-1)}(h = 1, 2, \ldots, N)$ from the $(p-1)$th layer and yields an output $v_j^{(p)}$, which is also a scalar, expressed in the following way:

$$v_j^{(p)} = f_j\left(\sum_{h=1}^{N} w_{hj} v_h^{(p-1)} + b_j^{(p)}\right) \tag{3.47}$$

where $f_j(\cdot)$ represents the nonlinear activation function. The output $v_j^{(p)}$ serves as input to the $(p+1)$th layer to which the neuron is connected. The weights $\{w_{hj}\}$ and biases $\{b_j\}$ are updated during training [70]. A gradient descent-based algorithm, the so-called back-propagation (BP) algorithm [80], is the most popular for the MLP training.

In MLP equalizers, the network output is passed through a decision device to determine the estimated symbol. The MLP has been incorporated into the DFE structure to enhance the performance of the DFE by Siu *et al.* [70] and Chen *et al.* [81], and it has been shown that the MLP-based DFE provides better performance than conventional DFEs. Afterwards, a number of MLP-based equalizers have been developed for various channels during the last decade. Some of them have been applied for CCI suppression [82], equalization of satellite channels [83], indoor radio channels [84] and nonlinear magnetic storage channels [85]. Very recently, the MLP-based DFE with lattice structure, which can overcome slow convergence rate of the MLP-DFE through whiten-

ing the input data, has been developed for equalization of both fixed and time-varying channels [78],[86].

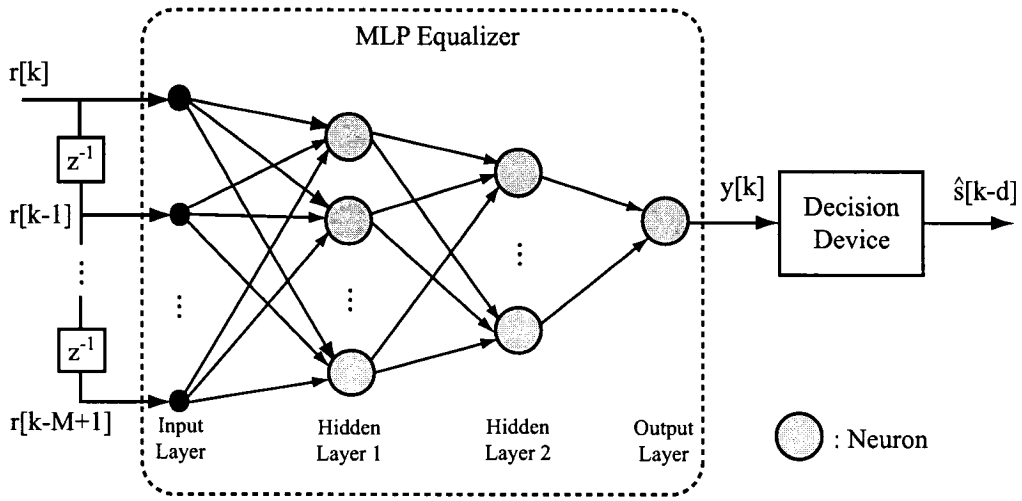


Fig. 3.6 Structure of an MLP equalizer.

Despite a performance improvement, MLP equalizers have raised controversial issues. These include [73]:

- There is very little understanding on the relationship between the MLP structures and the equalization problems.
- The high degree of nonlinearity of MLPs makes their theoretical analysis of performance difficult, and therefore adaptation parameters are generally chosen by trial and error.
- No relationship has been derived between the MLP equalizers and the optimal Bayesian equalizers.
- There is no weight initialization method guaranteeing proper weight convergence.
- Gradient descent-based learning algorithms for the MLP equalizers require long training sequence and time.

Some of these issues have been addressed in the literature. To speed up the training, Kalman filters [87] and least squares (LS)-type algorithms [88] have been developed at the cost of computational

complexity. The issues relating to the design of MLP structures for equalization applications have been addressed in [89].

Radial Basis Function (RBF) Networks

In the beginning, RBF networks were developed for data interpolation in multi-dimensional space [90]. Later, the RBF networks were reinterpreted as a least squares (LS) estimator by Broomhead and Lowe [91]. This approach, as the LS estimator, has led to its widespread application in signal processing problems such as time-series prediction, system modeling, interference rejection and channel equalization [58].

Consider an RBF equalizer with M inputs and N RBF functions. As shown in Fig. 3.7, the RBF equalizer can realize a mapping $f : \mathbb{R}^M \rightarrow \mathbb{R}$ by the following function

$$y[k] = \sum_{i=1}^{N} w_i \phi_i(||\mathbf{r}[k] - \mathbf{c}_i[k]||) \tag{3.48}$$

where $\mathbf{r}[k]$ is the input vector, w_i is weights, and $\phi_i(\cdot)$ is a radially symmetric, scalar basis function with centers $\mathbf{c}_i$. The RBF has several well-known basis function families. Typical examples includes Gaussian kernel, thin plate spline, multi quadratic and inverse multi-quadratic [58]. The relationship between the RBF equalizer and the BE introduced above was derived in [75]. When the Gaussian kernel is employed to configure the RBF equalizer, the relationship between (3.48) and (3.38) is clear. The centers of (3.48) are the noise-free channel states of (3.38). In an equalization scheme, since the decision function is passed through a decision device, the scalar value, for example, $\frac{1}{2\sigma_o^2}$ is not relevant as far as the decision functions are concerned and can be normalized to unity. This relationship can provide the parametric implementation of the BE with the RBF equalizer.

Training the RBF equalizer includes two processes: estimation of the centers or clustering and learning of the weights connected with the output layer. One of the most popular schemes to estimate the centers is a clustering algorithm like the k-means algorithm [51]. Other schemes for training the centers include the orthogonal least squares (OLS) [92] and back-propagation algorithm [93]. The output weights are trained in a supervised manner. The fastest manner in terms of learning rate is to use the relationship with the Bayesian equalizer and simply assign

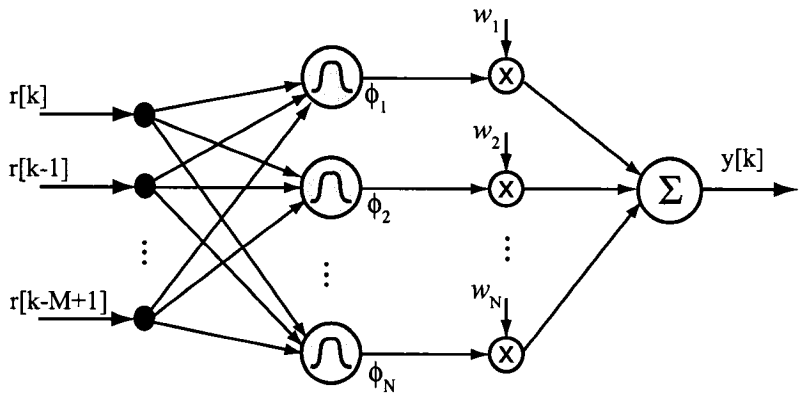


Fig. 3.7 An RBF equalizer.

values (-1 or $+1$) to the weights, as explained in (3.39). The weights can also be trained with an LMS, RLS, or other learning algorithms.

Numerous RBF equalizers have been developed in the literature to overcome ISI, CCI, noise, and nonlinear distortions of digital communication channels. Among them, Chen *et al.* developed an RBF equalizer for overcoming ISI and CCI, and introduced a complex-valued version of an RBF equalizer [74],[94],[95]. Cha *et al.* [96] also introduced complex RBF networks independently and employed a stochastic gradient learning algorithm for their complex RBF equalizer. In addition, a recurrent RBF equalizer was devised by Cid-Sueiro *et al.* [97] to reduce the complexity which grows exponentially with the memory of the channel, and it is an alternative to Viterbi algorithm for non-Gaussian noise. Recently, Deng *et al.* [98],[99] applied a new RBF network learning algorithm, the minimal resource-allocation network (MRAN) [100], to channel equalization problems. Complex-valued implementations of RBF equalizers were proposed by Chen *et al.* [94],[95].

The RBF equalizer is subject to computational complexity though it can provide optimal performance with small pilot sequences. The computational burden grows exponentially as the number of RBF units increases with equalizer order and the channel delay order. Some issues for computational complexity and design of RBF equalizers have been discussed in [58].

Recurrent Neural Network Equalizers

Feedforward NNs (FNNs), such as MLP or RBF networks, are mainly concerned with equalizer design thanks to their structural simplicity [58],[78]. However, recent research results show that recurrent neural networks (RNNs) [79] are superior to FNNs in modeling nonlinear systems and predicting time-series signals. Moreover, FNN-based equalizers require a large amount of connection and storage, while RNN-based equalizers can reduce the burdens of FNN counterparts in terms of connection and storage. The time-varying nature of fading channels may be interpreted as a dynamic system with uncertainties in its coefficients. FNN equalizers have an implicit difficulty in dealing with the time-varying nature because they are static nonlinear models.

In an RNN, the network structure handles time in an implicit manner through the use of *feedback* [57]. The dynamic behavior of the network can be described by a state-space model. A crucial property of a RNN described by the state-space model is that it can approximate a wide class of nonlinear dynamic systems. The network is able to extrapolate nonlinear dynamic data not trained on. Equalizer structures suitable for time-varying channels may benefit from these properties of RNNs. A structure of a decision-feedback RNN equalizer is depicted in Fig. 3.8 [101],[102]. The structure uses the simple recurrent network, so-called the *Elman's* network [103], which includes recurrent connections from the hidden units to a layer of context units consisting of unit delays. These context units store the outputs of the hidden units for one time step, and then feed them back to the input layer. Due to the nature of feedback around the hidden units, these units discover abstract representations of time [79]. The discrete state-space equations of the RNN equalizer follows the form

$$\mathbf{x}[k] = f(\mathbf{r}[k], \mathbf{u}[k], \mathbf{x}[k-1]; \mathbf{W}_h) \tag{3.49}$$

$$y[k] = f(\mathbf{x}[k]; \mathbf{w}_o) \tag{3.50}$$

where $\mathbf{r}[k]$, $\mathbf{u}[k]$ and $\mathbf{x}[k-1]$ represent the received signal vector, the decision feedback vector and the recurrent state vector, respectively. $\mathbf{W}_h$ and $\mathbf{w}_o$ are the weight matrix connected to the hidden layer and the weight vector linked to the output layer, respectively. The neurons operate with activation functions $f(\cdot)$. This RNN equalizer has been applied to equalization of time-varying linear and nonlinear channels and has achieved a good performance gain with fast convergence

rates [101],[102]. A similar RNN equalizer without decision feedback was proposed by Parisi *et al* [104].

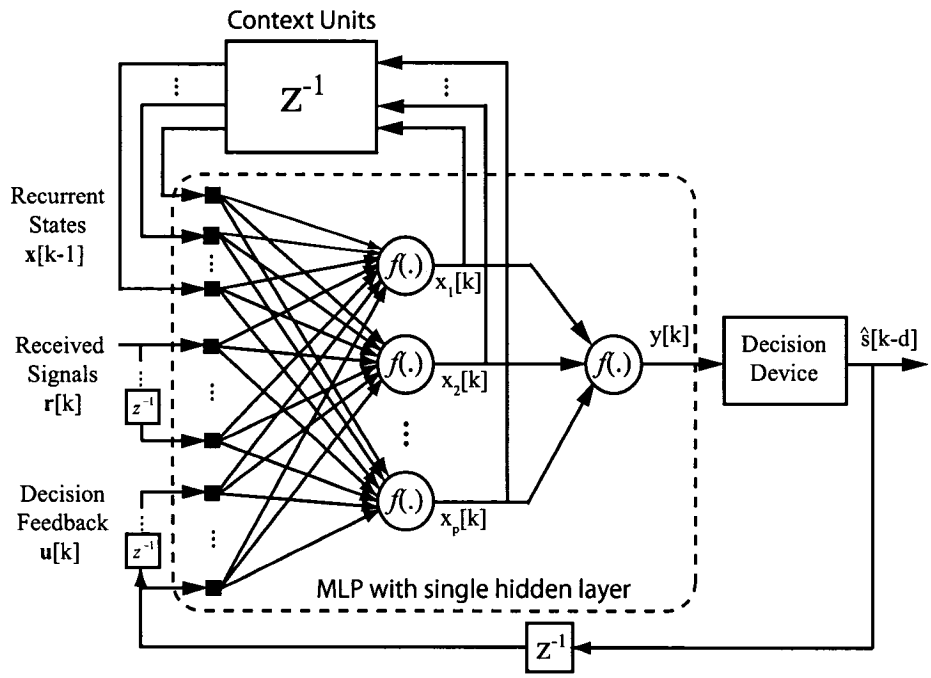


Fig. 3.8 Structure of a decision-feedback RNN equalizer.

Other popular RNN equalizers have been designed based on a fully connected recurrent network. Kechriotis *et al.* [105] showed that an RNN equalizer with a small number of neurons outperforms the LTE and the FNN equalizer for linear and nonlinear channels. A decision-feedback recurrent neural equalizer (DFRNE), a modified version of the RNN equalizer proposed in [105], was developed by Ong *et al.* [106] and modified by Hacioglu [107]. Ong *et al.* [106] also showed the DFRNE outperforms both the LTE and the FNN equalizers and the convergence rate of the DFRNE is faster and more robust than that of the RNN equalizer. Other RNN equalizer structures have been proposed. These include a complex-bilinear RNN [108] and single node and multiple node RNN equalizers [109].

3.2.5 Training Algorithms

Most of the equalizers described so far, including the LTE, DFE, BE and MLSE, are designed based on the perfect channel knowledge. In practice, the channel is in general unknown when the receiver is designed. Therefore, it must be estimated. Moreover, since in wireless communications the channels are time-varying, the channel should be periodically tracked. Adaptive equalization techniques optimize the equalizer coefficients using the training sequence and/or the detected data, without direct channel estimation [2].

In LTE and DFE, the optimization criterion commonly used for the equalizer coefficients is to minimize the MSE between the desired equalizer output and the actual equalizer output. The M equalizer coefficients $\mathbf{w}$ that minimize the MSE are obtained by the optimum Wiener filter solution, which may be expressed as [44],[12]

$$\mathbf{w}_{\text{opt}} = \mathbf{R}^{-1}\mathbf{p} \tag{3.51}$$

where $\mathbf{R}$ is the autocorrelation matrix of the received signal (i.e., equalizer input) vector $\mathbf{r}$ at given time instant, and $\mathbf{p}$ is the vector of cross correlations between the desired data symbol and the received signal vector. The convergence of the Wiener filter is very fast. It typically converges in around MT_s for the M equalizer coefficients. However, due to a matrix inversion, the computation requires $\mathcal{O}(M^2)$ to $\mathcal{O}(M^3)$ complexity.

If the computational complexity is a critical issue, alternatively, the minimization of the MSE can be achieved recursively by the stochastic gradient algorithm, called the least mean square (LMS) algorithm [12]. With the LMS algorithm, the coefficients are updated as

$$\hat{\mathbf{w}}[k+1] = \hat{\mathbf{w}}[k] + \mu e[k]\mathbf{r}^*[k] \tag{3.52}$$

where μ is the step-size parameter that governs the rate of adjustment, $e[k]$ is the error signal defined as the difference between the transmitted symbol $s[k]$ and at the equalizer output $y[k]$,

$$e[k] = s[k] - \hat{\mathbf{w}}^H[k]\mathbf{r}[k] = s[k] - y[k] \tag{3.53}$$

[2]Blind equalizers do not use the training sequence. Instead, they use the detected data only to learn the channel response and to train the equalizer coefficients.

and the $\mathbf{r}^*[k]$ represents the complex conjugate of the equalizer input vector $\mathbf{r}[k]$. The LMS algorithm offers greatly reduced complexity which is a linear function of the number of tap weights, i.e. $\mathcal{O}(M)$. The convergence rate of the LMS algorithm is typically slow because μ is small for stable and good performance of the algorithm.

A fast convergence rate can be achieved by the recursive least-squares (RLS) algorithm [12], which minimizes the sum of exponentially weighted errors. Let $\mathbf{P}[k]$ be the inverse correlation matrix. The RLS algorithm is computed by the following recursive equations:

$$\mathbf{k}[k] = \frac{\mathbf{P}[k-1]\mathbf{r}[k]}{\lambda + \mathbf{r}^H[k]\mathbf{P}[k-1]\mathbf{r}[k]} \tag{3.54}$$

$$\xi[k] = s[k] - \hat{\mathbf{w}}^H[k-1]\mathbf{r}[k] \tag{3.55}$$

$$\hat{\mathbf{w}}[k] = \hat{\mathbf{w}}[k-1] + \mathbf{k}[k]\xi^*[k] \tag{3.56}$$

$$\mathbf{P}[k] = \lambda^{-1}(\mathbf{P}[k-1] - \mathbf{k}[k]\mathbf{r}^H[k]\mathbf{P}[k-1]) \tag{3.57}$$

where λ is the sum of exponentially weighted squared errors. The filtering operation of the algorithm is described by (3.55) where the equalizer is excited to calculate the *a priori* estimation error $\xi[k]$, which is different from the *a posteriori* estimation error defined in (3.53). The adaptive operation of the algorithm is described by (3.56) where the weight vector is updated by increasing its old value by an amount equal to the product of the complex conjugate of the *a priori* estimation error $\xi[k]$ and the time-varying gain vector $\mathbf{k}[k]$.

The recursive update equation for the inverse correlation matrix $\mathbf{P}[k]$, given by (3.57), is numerically unstable. For this reason, other algorithms with better numerical stability have been derived based on a square-root factorization of $\mathbf{P}[k]$, defined as

$$\mathbf{P}[k] = \mathbf{P}^{1/2}[k]\mathbf{P}^{H/2}[k] \tag{3.58}$$

where the upper triangular matrix $\mathbf{P}^{H/2}[k]$ is the Hermitian transpose of $\mathbf{P}^{1/2}[k]$. Such algorithms are named *square-root* RLS (SR-RLS) algorithms [12],[110]. These algorithms have a computational complexity of order $\mathcal{O}(M^2)$. Other RLS algorithms are called fast-RLS algorithms [111],[112] which have been devised with a computational complexity on the order of $\mathcal{O}(M)$. A comprehensive description of adaptive RLS filtering algorithms can be found in [113].

Table 3.1 summarizes the computational complexity and the relative convergence rate of adaptive filtering algorithms [43],[66]. In general, the convergence rate of the LMS algorithm is particularly slow in channels with spectral nulls, whereas the convergence rate of the RLS algorithm is unaffected by the channel spectrum [44]. There are various tradeoffs in terms of complexity and convergence performance between the family of RLS algorithms and the two extremes of the Wiener filter and the LMS algorithm. It can be noted that the SR-RLS and fast Kalman filter may be unstable in their convergence and tracking, which is the price paid for their fast convergence with relatively low complexity [43].

Table 3.1 Training algorithms and their characteristics.

Algorithm	Multiplications	Complexity	Convergence
Wiener Filter	M^2 to M^3	Very high	Fast ($\approx MT_s$)
LMS	$2M+1$	Low	Slow ($> 10MT_s$)
RLS	$2.5M^2+4.5M$	High	Fast ($\approx MT_s$)
SR-RLS	$1.5M^2+6.5M$	High	Fast ($\approx MT_s$)
Fast Kalman	$20M+5$	Fairly low	Fast ($\approx MT_s$)

Instead of the MSE minimization, bit error rate (BER) minimization equalizers were proposed in [114] and [115] , which make the eye opening wider as the SNR approaches infinity. An approximate minimum BER (AMBER) equalizer proposed in [116] achieves a substantial reduction of BER. On the other hand, information theoretic learning-based adaptive equalizers using an entropy formulation were proposed. The entropy minimization technique for equalization was presented in [117] and the negentropy minimization equalizer can be found in [118].

The adaptive filtering algorithms discussed above so far have been limited to the standard LTE and DFE. Training algorithms of other adaptive equalizers, in particular recurrent neural equalizers, will be provided in Chapters 4 and 5.

3.2.6 Turbo Equalization

MLSE is the optimal form of equalization. Due to the exponentially-increasing complexity, using the MLSE equalizer is practically prohibitive in channels in which the ISI spans a large number

of symbols. Much work has been done to reduce the complexity of the MLSE. Most of these techniques are focused on either reducing the number of surviving sequences or shortening the length of the ISI span in the Viterbi algorithm.

Turbo codes and the turbo decoding principle, which achieve tremendous performance gains, can be used in equalizer design [119],[120]. The resulting turbo equalization is an iterative equalization and decoding technique that can achieve equally impressive performance gains for communication systems that send coded data over ISI channels. A block diagram building turbo equalization is given in Fig. 3.9. A turbo equalizer constitutes two components: the soft-input soft-output (SISO) equalizer and the SISO decoder. The turbo equalizer uses "soft" information, which can be converted into probabilities. Each SISO block computes the soft information, called the log likelihood ratio (LLR), associated with the transmitted symbol. The equalization and decoding process is performed by passing soft information between the equalizer and the decoder. The main idea of turbo equalization is to feed back the *a posteriori* probability (APP) values of individual bits obtained by one SISO block as *a priori* information to the other. The interleavers are included into the iterative loop to further disperse the direct feedback effect. In particular, the correlations between neighboring symbols are largely suppressed by the use of an interleaver.

In the SISO equalizer, an equalization algorithm yields the conditional LLRs $\mathcal{L}(a_k|\mathbf{r})$ of bit a_k given the observation $\mathbf{r}$. This LLRs split into the two contributions: the extrinsic LLRs $\mathcal{L}_e(a_k|\mathbf{r})$ and the intrinsic LLRs $\mathcal{L}(a_k)$, that is,

$$\mathcal{L}(a_k|\mathbf{r}) = \mathcal{L}_e(a_k|\mathbf{r}) + \mathcal{L}(a_k). \tag{3.59}$$

The prior LLRs $\mathcal{L}_e(c_k|\mathbf{r})$ produced by deinterleaving are fed to the SISO decoder. Similarly, the SISO decoder produces the *a posteriori* LLRs $\mathcal{L}(c_k|\mathbf{p})$, where $\mathbf{p} = [P(c_1|\mathbf{r})\ P(c_2|\mathbf{r})\ \ldots]^T$ is the set of probabilities. The SISO equalizer utilizes the extrinsic LLRs $\mathcal{L}_e(c_k|\mathbf{p})$ interleaved to $\mathcal{L}_e(a_k|\mathbf{p})$ as the prior LLRs. After some number of iterations, the turbo equalizer converges on its estimate of the transmitted symbol.

A MAP equalizer is usually used to compute the APP of the transmitted symbol given the past channel outputs [121],[122],[123]. Instead of the MAP equalizer, the linear MMSE equalizer can take advantage of the *a priori* LLRs $L_e(a_k|\mathbf{p})$ interpreted as probabilities on the transmitted

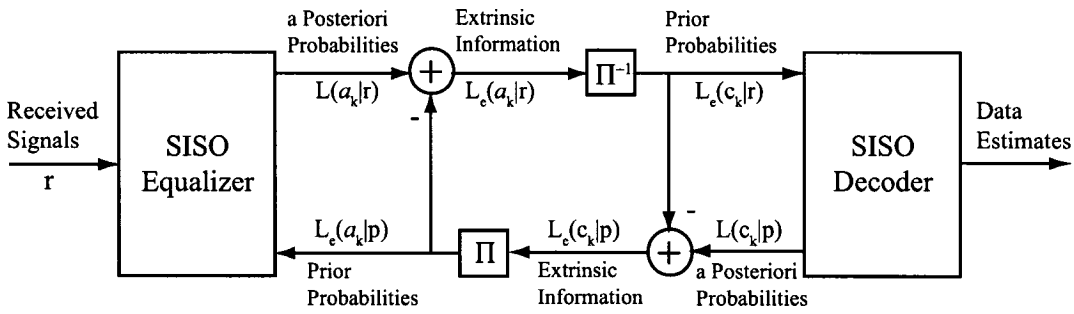


Fig. 3.9 Block diagram of turbo equalization.

symbols s_k by recomputing the statistics of s_k, i.e. the mean $\mathcal{E}[s_k]$ and covariance $\mathrm{C}[s_k, s_k]$, re-estimating $\hat{s}_k$ and recomputing the extrinsic information. This is equivalent to extracting only the extrinsic component of the information in the iterative scheme, thus it leads to low-complexity alternatives of the MAP equalizer [124],[125]. Several low-complexity alternatives for the SISO equalizer exist [126],[127],[128]. RBF network-assisted turbo equalizers with less complexity were also proposed in [129],[130].

For multiple-antenna wireless links, several MIMO turbo equalization techniques have been developed [131],[132],[133],[134],[135]. In particular, Abe and Matsumoto in [133] and [134] extended the MMSE equalizer to MIMO turbo equalization with iterative channel estimation.

3.3 Channel Estimation

The design of an optimal receiver needs the knowledge of the channel. The acquisition of channel state information (CSI), often unknown in practice, is important in communication systems in order to minimize the detection error. CSI is required at coherent receivers to perform key receiver signal processing functions such as equalization, interference cancellation and decoding. Furthermore, the knowledge of CSI at the transmitter can be used for adapting the modulation rate or for the power control.

The channel state is estimated by the receiver using a training and/or pilot sequence emitted by the transmitter. Referred to as *training-based channel estimation*, these techniques estimate

the channel from both the known training symbols regularly transmitted and their corresponding observations. Most of the mobile communication standards include training symbols for channel estimation. The "train-before-transmit" mode of operation for the training-based approaches is adequate when the channel is slowly changing as in the case of voiceband communications over telephone channels. A drawback of training-based channel estimation is that bandwidth efficiency decreases because of a non-negligible part of the data burst; for example, 20 % of the bits in a burst are used in GSM. Training-based estimation uses the received signals only containing known symbols, ignoring other information; for example, statistics of unknown symbols. Therefore some useful information gets lost. In certain communication systems, training symbols are not exploitable when synchronization between the transmitter and the receiver is not possible.

In *blind channel estimation* which is "train-while-transmit" mode of operation, no training symbols are used. The receiver estimates the channel from the signals received during normal data transmission. A benefit of the blind approach is the improvement of bandwidth utilization for time-varying channels. However, the most popular blind techniques based on second-order statistics (SOS) suffer from a lack of robustness. The SOS blind techniques leave an indeterminacy in the channel, a scale or constant phase, or a discrete phase factor [136]. This suggests that the SOS blind methods should not be used alone but with some form of additional information. Furthermore, because blind methods are based on the received signals only, no use is made of the knowledge of some input symbols and some information gets lost as in training-based estimation.

Semiblind channel estimation techniques can provide a solution to cope with the disadvantages of training-based and blind channel estimation. Semiblind methods combine both training-based and blind estimation techniques and are more powerful than the two methods separately. It is motivated by the fact that in data transmission some known symbols can always be incorporated to improve the quality of channel estimates.

In the following we only treat two approaches: training-based estimation and semiblind estimation techniques. Blind estimation approaches are beyond the scope of this thesis, yet the interested reader can be referred to [47].

3.3.1 Training-Based Estimation Techniques

The transmitter emits a training sequence or periodic pilot symbols, which are known to the receiver and can be used by the receiver for channel estimation. The pilot-assisted transmission is common in modern communication systems. Some of the standardized pilot design schemes are found in [137]. For example, in the GSM and EDGE (Enhanced Data rates for Global Evolution) systems [138], a known training sequence is located in the middle of the time slot. In the downlink slot structures of the Interim Standard (IS)-136 system [139],[140], a single training sequence is located in a preamble. However, in addition to the training sequence, known pilot symbols are placed regularly throughout the slot when the channel varies significantly.

Under the quasi-static assumption, the channel state remains constant over the length of a frame, changing independently between consecutive frames. We assume that the training symbols located in a preamble take the first P symbols of the whole symbol frame of length N. Assuming $K = P - (L - 1)$ received samples in total where L denotes the number of channel coefficients, the channel in (3.5) can be expressed as

$$\mathbf{r} = \mathbf{S}\mathbf{h} + \mathbf{n} \tag{3.60}$$

where

$$\begin{aligned} \mathbf{r} &= [r[0]\, r[1] \ldots r[K-1]]^T, \\ \mathbf{h} &= [h[0]\, h[1] \ldots h[L-1]]^T, \\ \mathbf{n} &= [n[0]\, n[1] \ldots n[K-1]]^T \end{aligned}$$

and $\mathbf{S}$ is a $K \times L$ matrix whose rows correspond to different shifts of the transmitted training sequence of symbols. It is common to model the channel coefficients as uncorrelated, zero-mean complex Gaussian random variables. This corresponds to Rayleigh fading, in that channel tap amplitudes are Rayleigh distributed. In the following, the channel model is given in (3.60) and noise is AWGN unless otherwise noted.

Snapshot Approaches

We give an overview of well-known snap-shot approaches for training-based channel estimation even though a wide variety of methods exploiting adaptive filtering techniques exist [44]. Comprehensive descriptions may be referred to [141] and [142].

Correlation-Based Estimation: A simple approach to channel estimation is a correlation method which correlates the received signal to a subsequence of the training sequence. The resulting channel estimate $\hat{\mathbf{h}}_{\mathrm{CO}}$ based on correlation is given by [143]

$$\hat{\mathbf{h}}_{\mathrm{CO}}[k] \cong R_{sr} \cong \frac{1}{P_s} \sum_{i=i_0}^{i_0+P_s-1} s^*[i]r[k+i]. \tag{3.61}$$

where P_s is the length of the subsequence used in correlation and i_0 is the index of the first symbol used in correlation. In [144] and [145], the correlation method was applied to the GSM cellular system, requiring a 26-symbol training sequence in the middle of the slot. The correlation method is computationally efficient, yet suffers from high noise sensitivity.

Least-Squares (LS) Estimation: If no assumptions are made about the statistics of the fading nor of the AWGN, the LS solution minimizing the squared error quantity yields the following estimator [142],[146]

$$\hat{\mathbf{h}}_{\mathrm{LS}} = \arg \min_{\mathbf{h}} \{\| \mathbf{r} - \mathbf{Sh} \|^2\} = \mathbf{S}^{\dagger}\mathbf{r} \tag{3.62}$$

where $\mathbf{S}^{\dagger} = (\mathbf{S}^H\mathbf{S})^{-1}\mathbf{S}^H$ is the Moore-Penrose pseudoinverse of the matrix $\mathbf{S}$. The computational complexity of finding the pseudoinverse of $\mathbf{S}$ is $\mathcal{O}(P^2M_T + PM_T^2 + \min(P, M_T)^3)$ [147]. The computational complexity of LS estimation increases exponentially as the number of training symbols increases. The LS solution is also the best linear unbiased estimate (BLUE) for the channel coefficients [148]. This solution can be further simplified to

$$\hat{\mathbf{h}} = \frac{1}{P}\mathbf{S}^H\mathbf{r} \tag{3.63}$$

provided that the periodic autocorrelation function of the training sequence is ideal with small delays because the correlation matrix $\mathbf{S}^H\mathbf{S}$ becomes diagonal.

Minimum Mean-Square Error (MMSE) Estimation: A widely used approach is linear MMSE channel estimation [149],[150],[151], which requires second-order channel statistics. We constrain the MMSE channel estimator to be a linear function of $\mathbf{r}$. By using the Wiener-Hopf equation [12] and an approximation $\mathcal{E}\{\mathbf{S}\mathbf{S}^H\} \cong \mathbf{S}\mathbf{S}^H$, the approximated linear MMSE solution is given by

$$\hat{\mathbf{h}}_{\text{MMSE}} = \arg\min_{\mathbf{h}} \mathcal{E}\{\| \mathbf{h} - \hat{\mathbf{h}} \|^2\} = (\mathbf{S}^H\mathbf{S} + R_n)^{-1}\mathbf{S}^H\mathbf{r} \tag{3.64}$$

where $R_n = \sigma_n^2\mathbf{I}$ is the autocorrelation matrix of the noise vector $\mathbf{n}$, taking account the *i.i.d.* assumption. It requires the knowledge of the noise variance. The LS estimator in (6.28) is therefore a special case of the MMSE estimator with no requirement of the noise power σ_n^2. More specifically, the MMSE approach approximates the LS approach at high SNR, whereas for low SNR it tends to the correlation approach.

Maximum Likelihood (ML) Estimation: The ML solution can be defined as

$$\hat{\mathbf{h}}_{\text{ML}} = \arg\max_{\mathbf{h}} p(\mathbf{r} \mid \mathbf{h}) = (\mathbf{S}^H R_n^{-1}\mathbf{S})^{-1}\mathbf{S}^H R_n^{-1}\mathbf{r} \tag{3.65}$$

where $p(\mathbf{r} \mid \mathbf{h})$ denotes the conditional probability density function (pdf) of $\mathbf{r}$ given $\mathbf{h}$. The ML turns out to be the best among all unbiased estimators in terms of having low MMSE and is the most efficient in the sense that it achieves the Cramér-Rao lower bound. This is also the classical linear LS estimator, which can be realized recursively. Assuming white noise, the ML solution above becomes equivalent to the LS solution given in (6.28) [146]. If the noise is not white, yet the noise covariance is known, then the ML method can improve channel estimation using knowledge of noise [149].

Maximum *a Posteriori* Probabilities (MAP) Estimation: The MAP solution can be defined as

$$\hat{\mathbf{h}}_{\text{MAP}} = \arg\max_{\mathbf{h}} p(\mathbf{h} \mid \mathbf{r}) = \arg\max_{\mathbf{h}} \frac{p(\mathbf{r} \mid \mathbf{h})p(\mathbf{h})}{p(\mathbf{r})} \tag{3.66}$$

where $p(\mathbf{r} \mid \mathbf{h})$ denotes the conditional pdf of $\mathbf{r}$ given $\mathbf{h}$. We observe that the MAP estimate is equivalent to maximizing $p(\mathbf{r} \mid \mathbf{h})p(\mathbf{h})$ where *a priori* statistical knowledge of the channel coefficient $p(\mathbf{h})$ is included. Assuming the noise and the channel coefficients are independent, the

MAP solution is given as [149]

$$\hat{\mathbf{h}}_{\text{MAP}} = (\mathbf{S}^H R_h \mathbf{S} + R_n)^{-1} R_h \mathbf{S}^H \mathbf{r} \tag{3.67}$$

where R_h is the covariance matrix of channel coefficients. When the noise is not white, the MAP estimator also needs knowledge of the noise covariance. Under the assumption that channel coefficients are i.i.d., complex Gaussian random variables, with zero mean and unit variance, the MAP and the MMSE estimator are equivalent [142].

Parametric Approaches: The composite channel response h can be expressed in the form of $\mathbf{h} = \mathbf{Ac}$, where A represents a known quantity, the pulse shape autocorrelation function, and c denotes an unknown quantity which is the radio channel response. According to the parametric system model, the snapshot approaches to channel estimation, described so far, can take advantage of this structure because we need to estimate only the unknown component c. For example, parametric approach-based LS channel estimation has been developed in [149],[152]. The LS estimate for c is given by

$$\hat{\mathbf{c}} = (\mathbf{A}^H \mathbf{S}^H \mathbf{A})^{-1} \mathbf{A}^H \mathbf{S}^H \mathbf{r}. \tag{3.68}$$

Because the known symbols include the known pulse shape autocorrelation A after transmit and receive filtering, the composite channel response estimate $\hat{\mathbf{h}}$ is represented as

$$\hat{\mathbf{h}} = \mathbf{A}\hat{\mathbf{c}}. \tag{3.69}$$

The parametric approach may be beneficial when the number of channel taps is less than that of composite channel taps L due to the reduction of estimation error variance.

Pilot-Assisted Interpolation Approaches for Time-Varying Channels

For the slot structure of IS-136 with pilot symbols, where the long slot duration requires a time-varying channel model, the pilot symbols regularly inserted can be effectively used to estimate the channel. *Pilot-assisted interpolation* approaches measure the channel at the pilot locations and use the measurements to estimate the channel states at the data locations. Linear interpolation technique is one of the simplest forms of pilot-assisted channel estimation. The channel estimate

with linear interpolation is a linear combination of the two neighborhood channel measurements [153],[154]. Linear interpolation may be viewed as a filter with symbol-spaced taps which include zeros at the unknown data locations. To obtain the channel estimates in the sense of MMSE, Wiener filtering can be used for interpolation [155],[156]. The coefficients of the Wiener filter depend on the availability of the Doppler spread. The filter coefficients can be chosen based on the worst case value of the Doppler spread when unavailable. Other types of filters such as Gaussian interpolation filters or Nyquist interpolation can be used too.

3.3.2 Semiblind Estimation Techniques

Semiblind estimation methods are aimed at combining both training sequence and blind information and at exploiting the positive aspects of both training-based and blind estimation techniques. The concept of semiblind channel estimation is depicted in Fig. 3.10 which explains it using a GSM burst. There are advantages of semiblind techniques over both training-based and blind counterparts [136]. First, semiblind techniques avoid the possible pitfalls of blind estimation because they incorporate the information of known symbols. Thus semiblind estimation appears more robust than blind estimation. Second, in addition to the known symbols, exploiting the blind information allows the estimation of longer channel impulse responses than with a certain training sequence. This feature is of interest for the application of mobile communications in mountainous area. Third, compared to a training-based approach, semiblind methods allow the use of shorter training sequences for a given channel length and desired estimation equality. In addition to robustness considerations, the performance of semiblind techniques is superior to that of both training-based and blind techniques separately. Lastly, semiblind techniques are promising in cases where training-based and blind methods fail.

Semiblind ML Methods

Although ML estimation is computationally expensive, it is popularly used for channel estimation because it is conceptually simple and usually has good performance when the sample size is sufficiently large. Semiblind estimation assumes the additional knowledge of the input symbols; particularly, part of input symbols is known. There are two semiblind ML estimators studied based

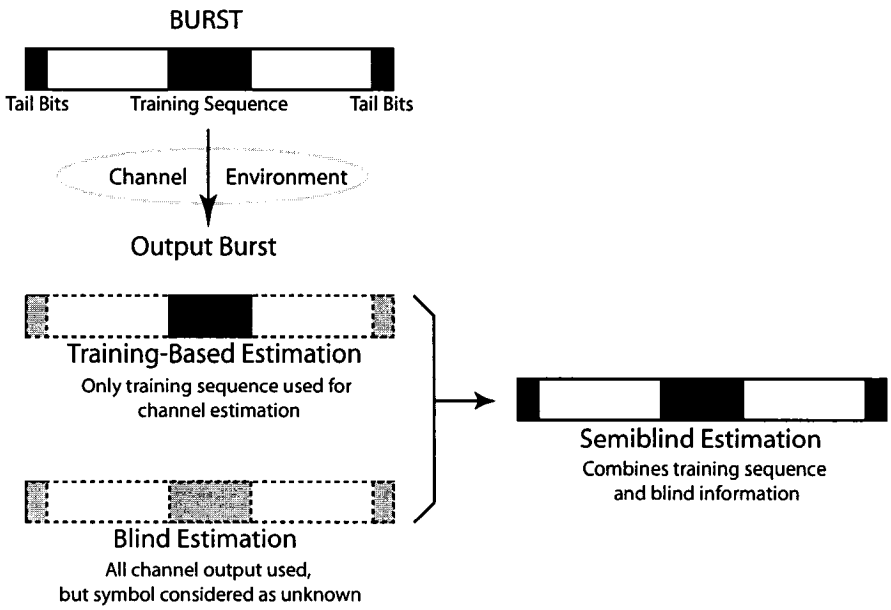


Fig. 3.10 Principle of semiblind estimation (Example of a GSM burst).

on different models corresponding to two different forms of *a priori* knowledge on unknown input symbols [136]. The first model is the deterministic model which does not exploit any knowledge and considers both the unknown input symbols and the channel as deterministic quantities. In the second model which is known as the stochastic model, the unknown input symbols are usually considered as a white Gaussian random variable with known parameters.

For semiblind estimation, the simplified system model of AWGN is given as [136]

$$\mathbf{R} = \mathcal{T}(h)\mathcal{S} + \mathbf{N} \tag{3.70}$$

where $\mathcal{T}(h)$ is a block Toeplitz matrix, $\mathcal{S}$ is the vector of input symbols written as $\mathcal{S} = \mathcal{P}\begin{bmatrix} S_K \\ S_U \end{bmatrix}$ where S_K are the known symbols and S_U are the unknown symbols, and $\mathbf{N}$ is the independent AWGN vector. In the deterministic model, the data have a Gaussian distribution with mean and covariance

$$m_R = \mathcal{T}_K(h)\mathcal{S}_K + \mathcal{T}_U(h)\mathcal{S}_U \tag{3.71}$$

$$C_{RR} = \sigma_n^2 I. \tag{3.72}$$

Deterministic techniques proceed to the estimation of h with $\mathcal{S}_U$ as nuisance parameters. The estimation is based on the received signal structure. In the stochastic (or Gaussian) model, the unknown input symbols are considered as i.i.d. Gaussian random variables of zero mean and variance σ_a^2 and the known symbols as deterministic, i.e., mean $\mathcal{S}_K$ and variance 0. Therefore, the first and second-order moments of the data are

$$m_R = \mathcal{T}_K(h)\mathcal{S}_K \tag{3.73}$$

$$C_{RR} = \sigma_a^2 \mathcal{T}_U(h)\mathcal{T}_U(h)^H + \sigma_n^2 I. \tag{3.74}$$

The Gaussian assumption is intended for ML estimation approaches for which knowledge of the complete distribution is required, allowing incorporation of the SOS of the data:

$$\mathbf{R} \sim \mathcal{N}(m_R, C_{RR}) \tag{3.75}$$

where the Gaussian hypothesis for the symbols leads to a Gaussian distribution for R. Unlike in the deterministic model, the input symbols in the stochastic model are no longer nuisance parameters for channel estimation. The parameters to be jointly estimated are the channel coefficients and the noise variance.

A semiblind ML estimation approach was proposed based on both the deterministic model [157] and the stochastic model [158]. The results showed that the semiblind ML approach based on the stochastic model gives better performance than the counterpart based on the deterministic model. Moreover, the semiblind approach appears more robust than blind estimation as it allows the estimation of any channel with only a few training symbols [136]. A treatment of the performance of ML estimation in the form of the Cramér-Rao bound (CRB) and a performance comparison with training-based and blind estimation for the two models can be found in [159]. As demonstrated in [159], the semiblind estimation methods offer significant performance improvement over either the training-based or the blind estimation methods.

There are many generalizations of semiblind channel estimation approaches to incorporate known symbols. In [160] authors extended the approach proposed in [161] by restricting the transition of the hidden Markov model. In [162] a method using the knowledge of the known symbols is proposed to avoid the local minima in the maximization of the likelihood function,

which is often hampered by the existence of local minima. A popular approach is to combine the cost function for the blind channel estimator and that for the training-based estimator [163],[164].

Adaptive Filtering-Based Channel Tracking

In a time-varying channel environment the channel must be tracked by a channel estimator. In the case of IS-136 without pilot symbols, the channel estimation is performed by using both the training sequence and the data sequence, which belongs to a semiblind technique. Assuming that the data symbols are known, adaptive filtering techniques [12],[165] are a fundamental tool for channel estimation and tracking. Explicit estimation of the channel statistics, required in MMSE, ML and MAP channel estimation, can be avoided by using adaptive filtering-based channel estimators [166]. Adaptive filtering-based estimation approaches are closely related to a model of how the channel changes in time. There are two types of models approximating the channel dynamics: the deterministic and the stochastic models. The deterministic models represent the time evolution of the channel coefficients as a deterministic parametric function. Popular deterministic models may be the complex sinusoidal model [167] and the polynomial model using a set of polynomial basis functions [168]. In the following we only focus on the stochastic models.

In the stochastic models, the channel coefficients are described as stochastic random processes. The state-space model is commonly used to describe the channel environment as a linear, discrete dynamical system [12]. The state-space model is represented by the following pairs of equations:

$$\mathbf{x}[k+1] = \mathbf{F}\mathbf{x}[k] + \mathbf{w}[k] \tag{3.76}$$

$$\mathbf{y}[k] = \mathbf{G}\mathbf{x}[k] + \mathbf{n}[k]. \tag{3.77}$$

In the process equation given in (3.76), the state vector denoted by $\mathbf{x}(k)$ corresponds to the channel coefficients $\mathbf{h}[k]$, and $\mathbf{F}$ and $\mathbf{w}[k]$ denote the state transition matrix and the process noise vector, respectively. In the measurement equation of (3.77), $\mathbf{y}[k]$ which corresponds to the received signal vector $\mathbf{r}[k]$ is observable in the presence of the measurement noise $\mathbf{n}[k]$, and $\mathbf{G}$ is the measurement matrix which includes the transmitted symbols. The process equation can be simplified employing

the random walk model [169] as follows

$$\mathbf{h}[k+1] = \mathbf{h}[k] + \mathbf{w}[k] \tag{3.78}$$

by denoting the state transition matrix $\mathbf{F}$ as identity.

The time-varying channel coefficients can be modeled as autoregressive (AR) or autoregressive moving average (ARMA) processes. For example, if the channel components are assumed to fade independently and following the same statistical model, the process equation employing a first-order AR (AR(1)) process can be written as

$$\mathbf{h}[k+1] = \alpha\mathbf{h}[k] + \mathbf{w}[k] \tag{3.79}$$

where $0 < \alpha \leq 1$. For a Rayleigh fading channel of maximum Doppler frequency f_D, $\alpha = J_0(2\pi f_D T_s)$ as in (3.6). It is disclosed that the AR(1) model offers sufficient accuracy to model the narrowband time-varying channel [37]. Higher order AR modeling [170] is also used, including adaptive estimation of the AR parameters. Second-order AR (AR(2)) channel model [171],[172] is appropriate when the channel changes rapidly.

Least Mean Square (LMS) Estimation: The LMS family of algorithms is the most widely used in digital communications due to its simplicity and stability properties. The received signal in (3.7) can be rewritten as follows:

$$r[k] = \mathbf{s}^H[k]\mathbf{h}[k] + n[k]. \tag{3.80}$$

Considering the AR(1) channel model given in (3.79), we represent general LMS algorithms of the form [12]

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \mu\,\mathbf{s}[k]\,f_e^*[k] \tag{3.81}$$

where $\hat{\mathbf{h}}[k]$ is the estimate of $\mathbf{h}[k]$ at iteration k, μ is the step size, and $f_e[k]$ is the generic scalar function of the output estimation error. Different choices for $f_e[k]$ result in different adaptive LMS

algorithms. For example, for the standard LMS algorithm the error function $f_e[k]$ is

$$f_e[k] = e[k] = r[k] - \hat{\mathbf{h}}^H[k]\,\mathbf{s}[k] = r[k] - \hat{r}[k] \tag{3.82}$$

and for the normalized LMS (NLMS) algorithm the error function is given as

$$f_e[k] = \frac{e[k]}{\|\mathbf{s}[k]\|^2}. \tag{3.83}$$

We may view the NLMS estimator as the LMS estimator with a time-varying step-size parameter [12].

For a time-invariant channel, the LMS-based channel estimator can be viewed as an iterative, stochastic gradient approach to finding the channel coefficients, minimizing the MSE between the received signals and the estimated received signals. The choice of the step size μ is a trade-off between convergence rate and how noisy the model is (misadjustment noise). For a time-varying channel the step size trades tracking ability for misadjustment noise.

The application of the LMS algorithm for channel tracking goes back to three decades [173] and thereafter the LMS algorithms have become a popular form of tracking. For a complete review refer to [12] and [174] and references therein. Recently, for the nonstationary channel environment, the tracking performance of the different algorithms of the LMS family has been studied in [175], where expressions for the MSE and selections of step sizes for optimal tracking performance are derived and the performance of different LMS algorithms is compared. The performance of the LMS algorithms on the estimation of time-varying channels is analytically evaluated in [176], using the estimation error correlation matrix, the mean-square weight error and the MSE as parameters. In [177], the use of the second-order LMS algorithm and its self-tuning implementation for wireless channel tracking in frequency domain have been exploited.

Recursive Least Squares (RLS) Estimation: A drawback of the LMS algorithm is the slow convergence properties. Thanks to the fast convergence rates of the RLS algorithm, it has been widely applied to channel tracking. RLS channel estimation is performed according to the following [12],[113]

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \mathbf{k}[k]e^*[k] \tag{3.84}$$

and

$$\mathbf{k}[k] = \frac{\mathbf{P}[k]\mathbf{s}[k]}{\lambda + \mathbf{s}^H[k]\mathbf{P}[k]\mathbf{s}[k]} \tag{3.85}$$

$$\mathbf{P}[k+1] = \lambda^{-1}\left(\mathbf{P}[k] - \mathbf{k}[k]\mathbf{s}^H[k]\mathbf{P}^H[k]\right) \tag{3.86}$$

where λ is the forgetting factor. Unlike the LMS algorithm, the RLS algorithm updates a second-order quantity, the matrix $\mathbf{P}[k]$, which is the inverse of an exponentially smoothed estimate of the symbol vector correlation matrix. The performance index of the RLS algorithm is defined as an exponentially windowed sum of the squared error

$$\mathcal{J}[k] = \sum_{i=1}^{k} \lambda^{k-i} |e[i]|^2 = \sum_{i=1}^{k} \lambda^{k-i} \left| r[i] - \hat{\mathbf{h}}^H[k]\,\mathbf{s}[i] \right|^2. \tag{3.87}$$

This is why this form of the RLS algorithm is referred as the exponential windowed RLS (EW-RLS) technique [178]. As the errors in modeling the past received signals are weighted less, the RLS estimator tracks the channel by trying to accurately model the most recent data. Like the step size of the LMS algorithm, the forgetting factor λ (< 1) contributes to the convergence rate and the misadjustment properties of the RLS algorithm.

The EW-RLS algorithm is commonly used for the tracking of wireless channels [179],[149]. In [179], an optimal windowed RLS algorithm was developed in the frequency domain. Alternately, a sliding window RLS algorithm was also used in [168] to track parameters for a polynomial model of the channel. A linear polynomial-based EW-RLS algorithm in conjunction with the variable forgetting factor was developed to improve the tracking capability of a fading channel. Applications of a variety of RLS filters to channel tracking can be found in [174].

Kalman Filter Estimation: Based on the state-space model given in (3.76) and (3.77), Kalman filtering can jointly solve the process and measurement equations for the unknown state in an optimum manner. The Kalman filter can be applied to the channel estimation problem, assuming known symbols and known noise covariance. The one-step prediction Kalman filter of the channel states is given by

$$\hat{\mathbf{h}}[k+1] = \mathbf{F}\hat{\mathbf{h}}[k] + \mathbf{K}[k]e^*[k]. \tag{3.88}$$

In the above equation, the Kalman gain matrix $\mathbf{K}$ and the error correlation matrix $\mathbf{P}$ are defined as, respectively

$$\mathbf{K}[k] = \frac{\mathbf{F}\mathbf{P}[k]\mathbf{G}[k]}{\mathbf{G}^H[k]\mathbf{P}[k]\mathbf{G}[k] + \mathbf{Q}_n[k]} \tag{3.89}$$

$$\mathbf{P}[k+1] = \mathbf{F}\left(\mathbf{P}[k] - \frac{\mathbf{P}[k]\mathbf{G}[k]\mathbf{G}^H[k]\mathbf{P}[k]}{\mathbf{G}^H[k]\mathbf{P}[k]\mathbf{G}[k] + \mathbf{Q}_n[k]}\right)\mathbf{F}^H + \mathbf{Q}_w[k] \tag{3.90}$$

where $\mathbf{Q}_w$ and $\mathbf{Q}_n$ denote the correlation matrices of the process noise and the measurement noise, respectively.

In [180] the Kalman filter was applied to track fading channels during the data mode while the fast start-up estimator was used to obtain a unbiased channel estimate during the training mode. For a fast time-varying ISI channel a Kalman channel estimator using the soft statistics represented by the *a posteriori* probabilities of the channel states was proposed in [181]. Chen et al. [182] proposed a mixture Kalman filter-based adaptive receiver based on a Bayesian formulation and the sequence Monte Carlo technique. Recently, applications of the Kalman filter to the channel estimation has been developed for orthogonal frequency division multiplexing (OFDM) systems [183],[184], MIMO wireless channels [185], and space-time coded systems [186]. Song et al. [187] proposed Kalman filter-based iterative channel estimation combined with turbo equalization.

The Kalman filter for channel estimation discussed so far is based on the state-space signal model. Other channel models can be used along with a Kalman filter framework for channel estimation. A first-order random walk model, which simplifies the Kalman filter recursions given in (3.88) through (3.90), was used to channel estimation in [188]. Along with Kalman channel tracking, AR(1) channel modeling can be found in [170] and ARMA channel modeling can also be found in [189] and [190]. On the other hand, a computationally efficient Kalman-LMS algorithm [171] was developed by approximation and applied to second-order models which are the integrated random walk model and the AR(2) model.

3.3.3 Iterative Channel Estimation

A key issue in channel estimation is to reduce a portion of the training sequence and pilots to achieve high data rates and bandwidth efficiency. A future trend in channel estimation is to use as

few training symbols as possible. An important class of algorithms for semiblind channel estimation is to exploit an iterative strategy. The turbo principle applied to turbo decoding [191] and turbo equalization [119] along with channel estimation leads to iterative (or turbo) channel estimation [192],[193] where the quality of the channel estimates is improved from iteration to iteration.

Fig. 3.11 shows the basic principle of a turbo receiver which performs iterative channel and symbol estimation. In the turbo receiver, iterative procedures for channel and symbol estimation are as follows. First, initial channel estimation is done with only training symbols, and the channel estimate and its variance is delivered to the turbo decoder. Next, the turbo decoder provides the channel estimator with the data symbol estimate, which is possibly a well detected tentative decision in the form of soft or hard information. These symbol estimates are used as additional training symbols for iterative channel estimation. Hence, the channel estimator makes use of the entire received data. In the last step, the channel estimator delivers the new channel estimate and its variance to the turbo detector. This procedure of channel estimation and symbol detection is performed iteratively until the turbo receiver converges.

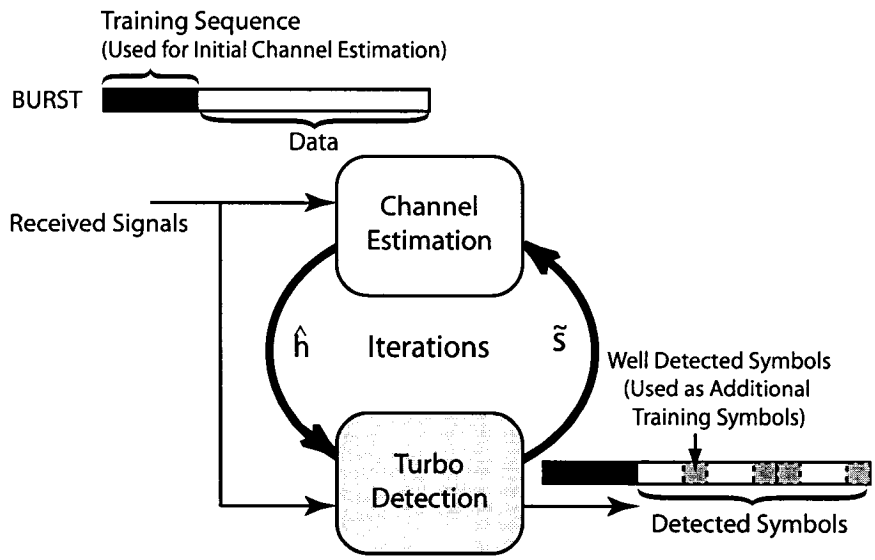


Fig. 3.11 Turbo channel-and-symbol estimation.

There are distinct advantages of iterative channel estimation [194].

- Initially less training symbols are required, i.e., more user data can be transmitted.

- After some iterations the channel estimate is capable of approaching the ML estimates which are optimal.
- The iterative strategy of channel estimation allows the receiver to use different algorithms, such as DFE, MLSE via Viterbi algorithm, or MAP detection via the BCJR (Bahl, Cocke, Jelinek and Raviv) algorithm [195].
- Iterative schemes easily exploit any *a priori* information that the receiver may have. This *a priori* information may arise because of pilot symbols, training symbols or error-control coding.

The feedback information from the turbo detector can be hard decisions or soft information on the data bits leading to hard or soft iterative channel estimation, respectively. Hard iterative channel estimation has been proposed in [192],[193],[196],[197], while soft iterative channel estimation has been proposed for block fading channels [192],[198],[199],[200] and for time-varying channels [187],[193],[201],[202]. In [181] channel estimation is proposed using soft symbols feedback from the detector (or equalizer) rather than the decoder. Kimura *et al.* [203] have studied the performance of two types of iterative channel estimation combined with turbo decoding; one is the outer-turbo channel estimation which carries out channel estimation before turbo decoding, and the other is the inner-turbo channel estimation [193] which incorporates iterative channel estimation into turbo decoding.

Adaptive filtering techniques discussed in Section 3.3.2 can be applicable to iterative channel estimation. The LMS filter was used in [202] and [204], the RLS filter was used in [201],[202], and [204], and the Kalman filter was employed in [187], [202] and [204].

3.3.4 Channel Estimation in MIMO Wireless Systems

All of the approaches to channel estimation discussed so far can apply to MIMO wireless systems, except that the signal model is suitably structured to reflect the MIMO training structure. In multiple-antenna transmit systems, additional training effort will be required because more parameters proportional to the number of transmit antennas have to be estimated [205],[206],[207]. There is a tradeoff between having more training symbols for better estimation and more channel

uses for higher rates. Optimal training design using information theory (i.e., Shannon capacity) was studied by Marzetta [205] and later by Hassibi and Hochwald [207] for the class of ergodic block fading MIMO channels. Training signals from the multiple transmitters can be mutually orthogonal in some dimension: time, frequency, or code. Orthogonal training signals offer the best estimation accuracy for a given transmit power under most circumstances [208]. In MIMO channels, knowledge of the channel at the transmitter can be leveraged to provide significant value in beamforming or pre-filtering [209]. In the following our review concentrates on only semiblind channel estimation approaches for MIMO wireless systems.

Iterative Snapshot Approaches

Iterative channel estimation techniques using snapshot approaches, such as MAP, LS and MMSE methods, were developed for MIMO wireless systems. Grant [210] proposed iterative channel estimation of a flat-fading MIMO channel in a synchronous CDMA system, where the MMSE channel estimate is updated in each iteration using hard decisions on the data symbols obtained from the decoder. A pilot embedded technique for soft iterative channel estimation was proposed in [211] for CDMA systems in a flat-fading channel with different iterative detection structure. To avoid the matrix inversion of the MAP channel estimator, an iterative method for decision-directed channel estimation in MIMO systems was proposed in [212].

On the other hand, in [213], a soft iterative channel estimator based on MAP and MMSE methods was proposed for multiuser MIMO systems in frequency-selective fading channels, where soft decision feedback of all data symbols from the decoder to the channel estimator was considered, while hard decision feedback was considered in [214]. In [215], Loncăr *at al.* proposed a comprehensive algorithm that performs channel estimation, multi-user detection and decoding in an iterative fashion. The multipath channel was estimated using pilot symbols and soft estimates of data symbols. Particularly, performance of channel estimators were compared, which use soft decisions estimated with both the extrinsic probabilities and the *a posteriori* probabilities of the coded symbols computed by the decoder. A novel linear MMSE channel estimator which exploits soft estimates of all the data symbols was derived in [216] for frequency-selective block-fading channels. In [217] a linear MMSE iterative channel estimation was derived for a frequency-selective time-varying MIMO channel, while a pilot symbol assisted modulation-based iterative channel

estimation over time and frequency MIMO fading channels is proposed in [218], applied in conjunction with turbo equalization. For transmit diversity OFDM transmission through frequency-selective channels, an iterative channel estimator based on the MAP algorithm was developed for both space-frequency OFDM and space-time OFDM systems in [219].

Iterative Adaptive Filtering Approaches

Combined with a MIMO turbo equalizer, an RLS-based iterative channel estimator was used in [214],[133] and [134]. Each user's initial channel impulse responses are obtained by training sequences. At the first iteration, further refined channel impulse responses are estimated by the RLS filter using both training sequences and only reliable hard decisions of data symbols computed by the decoder. To select the reliable estimates of data symbols, a thresholding method is applied. At the second iteration, the MIMO turbo equalizer uses the refined channel estimates. Due to the turbo principle, more additional hard decisions used for channel estimation are produced with more iterations.

In [135] and [220], RLS-based adaptive iterative estimation for MIMO frequency selective channels was studied, which is an extension of the research reported by Tüchler *et al.* [201] for single-input single-output systems. Unlike the works in [133] and [134], Sun *et al.* [135] uses soft decision feedback and estimates key parameters such as adaptive forgetting factor of the RLS filter, error variance of soft information, and excess delay spread.

Tracking for MIMO Channels

Kalman filtering is widely used for tracking time-varying MIMO channels. In [185], time-varying MIMO channel tracking was proposed using the Kalman filter aided by staggered decisions from a finite-length MMSE-DFE, which performs the task of equalization and separation of the sources. The results also revealed that the performance with the receivers employing Kalman tracking is significantly better at the expense of higher complexity than normal LMS and RLS adaptations when using either AR(1) or AR(2) approximations of the channel. For time-selective channels approximated by an AR(1) model, a Kalman channel tracker was employed in [186] to enable space-time (ST) decoding with diversity gains. In the decision-directed mode, the decoded symbols

by the ST decoder replace the information symbols for channel tracking, while in the training mode initial channel estimates are acquired by training symbols.

In [183], decision-directed Kalman channel tracking algorithms were proposed in both time and frequency domain for multi-input single-output (MISO)-OFDM systems. For MIMO-OFDM systems, Schafhuber *et al.* [221] exploited the extended Kalman filter (EKF) for simultaneous channel tracking and estimation of the state-transition matrix of a state-space signal model. Very recently, a joint data detection and channel estimation algorithm for MIMO-OFDM systems was proposed in [184] which combines the QR decomposition (QRD)-M algorithm and the Kalman filter. The QRD-M algorithm is used for data detection, while the Kalman filter is used for estimation of channel impulse responses in a manner similar to [185].

3.3.5 Bayesian MIMO Channel Tracking

Particle filters rooted in Bayesian estimation have captured the great attention of many researchers in diverse areas including signal processing, statistics and econometrics [222]. A major potential of particle filtering stems from its ability to cope with the difficulties encountered in nonlinear and/or non-Gaussian problems. Applications of particle filtering to communication problems are revealed, which include channel equalization and detection [41],[223], multiuser detection [224],[225], as well as channel estimation [226],[227],[228],[17]. We only review particle filtering techniques focusing on their application to channel tracking problems.

Keeping a MIMO channel in mind, we may consider the general discrete-time state-space model, composed of the state and observation (or measurement) equations, respectively [222],[17]

$$\mathbf{x}[k+1] = \mathbf{f}(\mathbf{x}[k], \mathbf{w}[k]) \tag{3.91}$$

$$\mathbf{y}[k] = \mathbf{g}(\mathbf{x}[k], \mathbf{v}[k]) \tag{3.92}$$

where $\mathbf{x}[k+1]$ is the state vector, i.e., the $(M_R M_T \times 1)$ complex-valued vector of the channel states stacked up in an orderly form; $\mathbf{y}[k]$ is the observed vector at the receiver; $\mathbf{f}$ and $\mathbf{g}$ are generic vector-valued functions, which are possibly time-variant; $\mathbf{w}[k]$ and $\mathbf{v}[k]$ denote the dynamic and measurement noise processes, respectively. The state in (3.91) characterizes the state transition probability $p(\mathbf{x}[k+1] \mid \mathbf{x}[k])$, whereas the measurement in (3.92) describes the likelihood $p(\mathbf{y}[k] \mid$

$\mathbf{x}[k]$).

In wireless communications, it is known that the measurement noise $\mathbf{v}$ in (3.92) and the dynamic noise $\mathbf{w}$ in (3.91) are contaminated by non-Gaussian noise, especially in an indoor environment. Empirical evidence can be found in [229] and [230] where the statistical property of measurement noise is non-Gaussian due to the impulsive nature of man-made electromagnetic noise as well as natural background noise. In a non-Gaussian environment, the Kalman filter-based solution for channel-state tracking may not be optimal anymore. Thus more sophisticated solutions using particle filtering techniques have been investigated recently. Fig. 3.12 depicts a general structure of particle filter-based channel tracking for a space-time (ST) coded MIMO system.

To our best knowledge, Chin *et. al.* [226], [227],[231] investigated the use of particle filtering for tracking a time-varying MIMO channel for the first time. In [226] the performance of a particle-based channel tracker was evaluated for an uncoded MIMO system where the noise present is both Gaussian and non-Gaussian noise. The recursive procedure used to track the channel is as follows. First, a rough estimate $\tilde{\mathbf{H}}[k]$ of the current channel state is obtained from the previous state $\hat{\mathbf{H}}[k-1]$ as

$$\tilde{\mathbf{H}}[k] = \alpha\hat{\mathbf{H}}[k-1] \tag{3.93}$$

with an assumption that the initial channel state is known. Second, based on the received signal vector $\mathbf{r}[k]$ and $\tilde{\mathbf{H}}[k]$, an initial estimate of the transmitted symbols $\tilde{\mathbf{s}}[k]$ is made. Next, using particle filtering and the initial estimate $\tilde{\mathbf{s}}[k]$ the rough channel estimate is refined producing $\hat{\mathbf{H}}[k]$. Finally, a refined symbol estimate $\hat{\mathbf{s}}[k]$ is generated using $\hat{\mathbf{H}}[k]$. A symbol block was composed of both data and pilot symbols, where a pilot symbol was inserted regularly between the data symbols. This structure can aid to prevent the channel tracker's divergence. Auxiliary particle filter [29] was employed in [226] in order to improve the efficiency of the particles. For both Gaussian and non-Gaussian environments, a significant performance gain was achieved by the use of the particle filter-aided channel tracker over the case without the tracker at higher SNRs. In particular, a block refinement technique using polynomial fitting was proposed in [231], which exploits the smoothly varying nature of the channel. By using the block refinement procedure, a better estimate of the channel state was obtained. For ST encoded systems, channel tracking results aided by particle filtering were reported in [227],[228],[17],[232]. It is worthy of noting

that in [17],[232] the authors have used a real-life data measured by a vehicle driving in downtown Manhattan [233].

Channel estimation using particle filtering has great potential and promise. Thanks to hardware architectures for high-speed particle filters very recently proposed in [234],[235], the computational burden of particle filtering can be reduced.

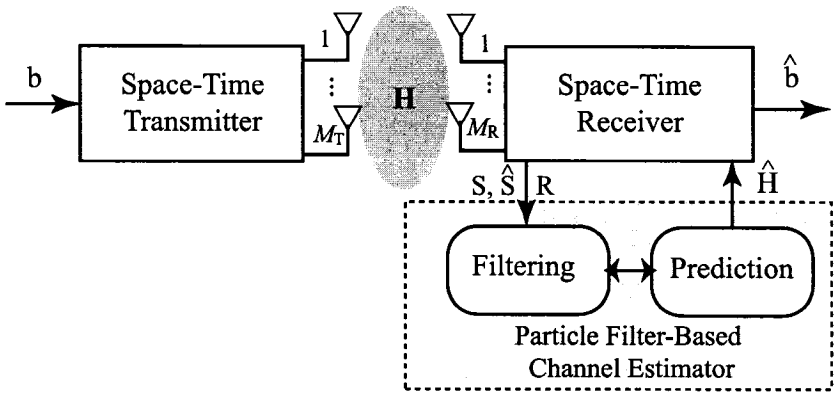


Fig. 3.12 Particle filter-based channel estimation in a MIMO link.

3.4 Non-Gaussian Noise Models

The ambient noise in communication channels has often been assumed to be Gaussian, being justified by the *central limit theorem*, because the Gaussian assumption often leads to mathematically tractable solutions for signal processing problems. However, the Gaussian noise model lacks realism for the ambient noise arising in many practical communication channels, including urban and indoor radio channels [229],[236] and underwater acoustic channels [237]. The ambient noise is known to be non-Gaussian through experimental measurements [229].

Generalized non-Gaussian models were developed by Middleton [3]: Class A, B, and C noise models. Class A noise (or interference) describes the type of electromagnetic interference (EMI) often encountered in telecommunication applications, and this ambient noise is largely due to *intelligent* [3] telecommunication operations. On the other hand, Class B noise usually represents natural

[3] In [3], *intelligent* noise or interference is distinguished from *nonintelligent* noise. Intelligent noise is defined as man-made and intended to convey a message or information. On the other hand, nonintelligent noise may be attributable to natural phenomena, such as atmospheric noise. Different types of noise are summarized concisely in

nonintelligent (i.e., non-information-bearing) noise and exhibits highly impulsive properties. Class C noise is the sum of Class A and B noise.

It is instructive that the Class A, B and C models can be distinguished based on EMI models, in terms of their bandwidth. Interference in Class A is coherent (which means that the noise bandwidth is less than the receiver bandwidth) in a narrowband receiver, causing a negligible number of transients (or ringings). Interference in Class B is impulsive (or incoherent), consisting of a large number of overlapping transients. Interference in Class C is a linear combination of both. These models have a direct physical interpretation and have been found to provide a good fit to a variety of noise and interference measurements.

In what follows, we discuss non-Gaussian noise models used extensively to model practical communication channels. Although asymmetric distributions [238] can model some practical models more precisely, we concentrate only on symmetric models for several reasons. The reasons include that: 1) Asymmetry may cause a significant increase in the computational complexity of signal processing; 2) *A priori* information about the process is required to estimate an asymmetric distribution; and 3) Symmetry represents plenty of important noise and interference processes found in wireless communications.

3.4.1 Gaussian Mixture Models

Simplified model used to approximate impulsive noise are Gaussian mixture models or contaminated Gaussian models. Gaussian mixture models have become popular in communications due to their mathematical tractability and their easy conceptual interpretation. These models give an approximation to Middleton's Class A model [239],[240], and have been used to model physical noise in radar, acoustic, and radio channels.

The Class A noise $n[k]$ combines the presence of an additive man-made noise component $i[k]$ with average variance σ_i^2 and an AWGN component $g[k]$ with variance σ_g^2 at time k, i.e.,

$$n[k] = i[k] + g[k]. \tag{3.94}$$

For the noise samples which are independently identically distributed (i.i.d.), the Class A pdf for

Table I in [3].

the real channel is given by [241]

$$p(n[k]) = \sum_{m=0}^{\infty} \frac{a_m}{\sqrt{2\pi}\sigma_m} \exp\left(-\frac{n^2[k]}{2\sigma_m^2}\right) = \sum_{m=0}^{\infty} a_m \mathcal{N}(0, \sigma_m^2) \tag{3.95}$$

with the abbreviation

$$a_m = e^{-A}\frac{A^m}{m!} \tag{3.96}$$

where the parameter A is the impulsive index. For the complex channel, the pdf is given by

$$p(n[k]) = \sum_{m=0}^{\infty} \frac{a_m}{2\pi\sigma_m^2} \exp\left(-\frac{|n[k]|^2}{2\sigma_m^2}\right). \tag{3.97}$$

Note that $p(n[k])$ is the weighted sum of an infinite number of Gaussian pdfs with increasing variance

$$\sigma_m^2 = \sigma^2 \frac{m/A + T}{1 + T} \tag{3.98}$$

where σ^2 defines the mean variance of the Class A model, and $T = \sigma_g^2/\sigma_i^2$ defines the relative variance of the two noise components, $i[k]$ and $g[k]$. As the impulsive index A becomes smaller, for example, $A = 0.1$, the Class A noise becomes impulsive, whereas the pdf becomes Gaussian for $A \rightarrow \infty$ [242]. Let the symbol $s[k]$ be transmitted over the memoryless additive white Class-A noise (AWCN) channel. The SNR of the AWCN channel is defined as

$$\text{SNR} = \frac{\mathcal{E}\{|s[k]^2|\}}{N_0} \tag{3.99}$$

where $N_0 = 2\sigma^2$ is the noise power spectral density.

Consider a commonly used two-term Gaussian mixture model, or ϵ-*contaminated* model, for the additive noise vector $\mathbf{n}$. The marginal pdf of the noise model can be represented as

$$p(n[k]) = (1 - \epsilon)\mathcal{N}(0, \sigma_1^2) + \epsilon\mathcal{N}(0, \sigma_2^2) \tag{3.100}$$

with $0 \leq \epsilon < 1$, $\sigma_1 > 0$, and $\sigma_2^2 = \kappa\sigma_1^2$ where $\kappa > 1$. The term $\mathcal{N}(0, \sigma_1^2)$ represents the nominal background noise, and the $\mathcal{N}(0, \sigma_2^2)$ term represents the impulsive component with ϵ controlling the probability that impulses occur. The shape of a distribution which affects the performance of

the system is varying by the parameters ϵ and κ with fixed total noise variance

$$\sigma^2 \triangleq (1-\epsilon)\sigma_1^2 + \epsilon\sigma_2^2. \tag{3.101}$$

Note that the strength of impulsiveness is associated with the ratio of the variances, $\frac{\sigma_1^2}{\sigma_2^2}$. An example of the pdf of a two-term Gaussian mixture model is given in Fig. 3.13, where $\sigma_1 = 1$, $\epsilon = 0.2$ and $\kappa = 2$.

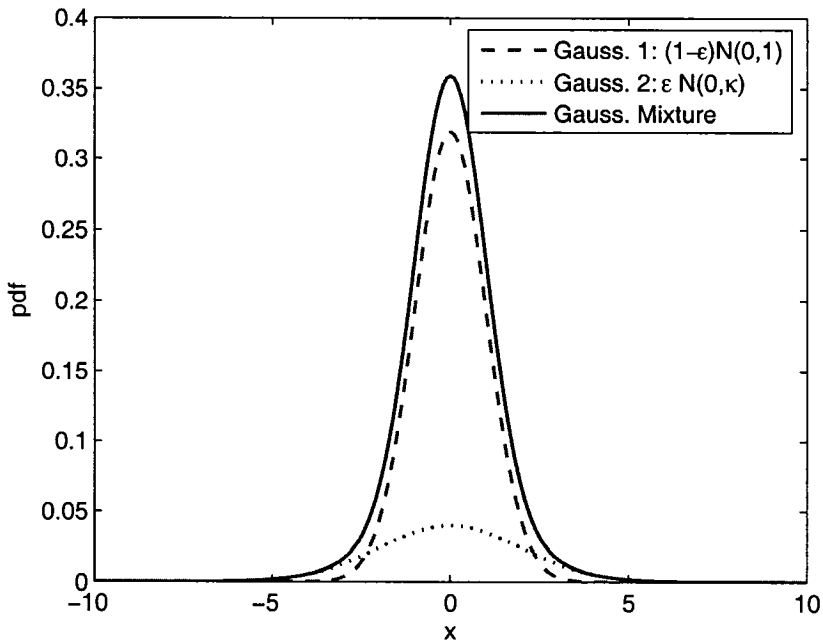


Fig. 3.13 Gaussian mixture density function ($\sigma_1 = 1, \epsilon = 0.2$ and $\kappa = 2$).

Plenty of research on communication receiver design using Gaussian mixture models has been done during the last decades. In 1970s, Madan *et al.* [243] considered the problem of impulsive noise suppression in analog communication systems. More recently, in multiple-access channels, a variety of techniques have been proposed to combat non-Gaussian ambient noise, especially Gaussian mixture noise. These include multiuser detection techniques [230],[244],[245],[246],[247], [248],[249], channel equalization [250],[251], and detection and estimation techniques [252],[253], [254]. Some adaptive receivers with certain nonlinearities were proposed in [255],[256] and [257].

Although the two-term Gaussian mixture model has become popular in communications, it oversimplifies the Class A model. Furthermore, the model in (3.100) has some drawbacks in modeling impulsive noise, which include [258]:

- The distribution is characterized by three parameters, ϵ, κ, and σ_1, in addition to the location. This may be an over-parameterized model.
- The model is complicated in terms of the computational aspects because the sum form of the density makes it difficult to handle general estimation problems.
- The use of the model in communications is physically unappealing because the impulsive nature of the noise processes observed in communications can not be explained by the model.

Another possible distribution is the Laplacian, which has heavier tails than the Gaussian. The Laplacian pdf is described by

$$p(n[k]) = \frac{1}{\sqrt{2}\sigma} e^{-\frac{\sqrt{2}|n[k]|}{\sigma}} \tag{3.102}$$

where the scale parameter $\sigma > 0$ is similar to the standard deviation of the Gaussian model. Some applications of Laplacian noise channels can be found. These include signal constellation [259], channel estimation and signal detection techniques [260],[261],[262],[263],[264], turbo decoders [265], and spectral efficiency [266],[267]. However, there is no strong evidence of appearance of heavy-tailed distributions in practical problems [258]. Furthermore, the "peaky" shape of the Laplacian pdf contradicts the widely accepted *Winsor's principle*, according to which all distributions are Gaussian in the middle.

The well-known generalized Gaussian model is a generalization of both Gaussian and Laplacian distributions. The generalized Gaussian distributions are described by exponential-type density functions as [268],[269]

$$p(n[k]) = \frac{\nu}{2\sigma\Gamma(\frac{1}{\nu})} e^{-\left|\frac{n[k]}{\sigma}\right|^{\nu}} \tag{3.103}$$

where σ and ν are the standard deviation of the distribution and a model parameter which controls the impulsiveness of the distribution, respectively, and $\Gamma(\cdot)$ is the Gamma function. For example, when $\nu = 2.0$ and 1.0, $p(n[k])$ becomes the Gaussian and Laplacian distributions, respectively. Conceptually, the lower the value of ν, the more impulsive the distribution is. The generalized

Gaussian distributions have been used as *ad hoc* models of electromagnetic radio noise [269]. However, the generalized Gaussian model has two limitations [258]. First, for $\nu < 1$ it inherits the peaky shape of the Laplacian distribution, resulting in the inability of the distribution to model practical impulsive phenomena. Second, its tails present decays of exponential order, in contrast with the commonly observed algebraic decay of heavy-tailed processes that appear in practice [270].

3.4.2 Alpha-Stable Models

Algebraic-tailed distributions possess heavier tails than both the Gaussian and the generalized Gaussian distributions, whose tails decay at an exponential rate. Therefore, algebraic-tailed models can serve as better representations of highly impulsive phenomena. There are two families of the algebraic-tailed models: generalized-t and α-stable models. These models play an important role in modeling noise and interference in communication systems. In this section we discuss α-stable models, which provide an approximation of the Class B model.

α-Stable Random Variables

An α-stable random variable is defined with its characteristic function, which is given by [270],[271]

$$P_\alpha(x) = \exp\left(-\gamma^\alpha |x|^\alpha (1 - j\beta \text{sgn}(x) \tan(\alpha\pi/2)) + j\delta x\right). \tag{3.104}$$

The characteristic function is determined by four parameters: α, γ, β and δ, where

- The characteristic exponent α: $\alpha \in (0, 2]$ is the index of stability which controls the heaviness of the tails of the stable density. A smaller value of α implies severe impulsiveness, while a value of α near 2 becomes a more Gaussian type of density.
- The scale parameter γ: $\gamma \in \mathbb{R}_+$ is also called the dispersion parameter, which plays a similar role to the variance of the distribution. For $\alpha = 2$, 2γ corresponds to the variance of the Gaussian distribution.
- The skewness parameter β: $\beta \in [-1, 1]$ is the skewness parameter which controls the symmetry of the distribution. The distribution is symmetric about the center δ when $\beta = 0$. For

$\alpha \neq 1$, left skewness and right skewness are represented by $\beta > 0$ and $\beta < 0$, respectively.

- The location parameter δ: $\delta \in \mathbb{R}$ is the location parameter which is corresponds to the mean of the distribution only when $\alpha > 1$ or $\beta = 0$. When $\alpha \leq 1$ and $\beta \neq 0$, the mean of the corresponding α-stable random variable is infinite.

For a random variable X distributed according to the rule described in (3.104), we use the notation $X \sim S_\alpha(\gamma, \beta, \delta)$. When $\beta = \delta = 0$, i.e., X is a symmetric α-stable (SαS) random variable, we denote $X \sim S\alpha S$. It has been found that the family of α-stable random variables provides useful models for a wide range of impulsive phenomenon [272]. α-stable processes possess desirable properties for modeling impulsive noise, including [273]:

- Choice of symmetry or asymmetry;
- Confirmation to the generalized central limit theorem;
- Having heavier tails than a Gaussian process;
- Heaviness of tails controlled by a single parameter.

The impulsive noise can be modeled as an α-stable random process [272],[258]. It is noted that the SαS model closely approximates the simple Middleton's Class B noise [3]. The SαS pdf $p_\alpha(t)$ is defined by its characteristic function, which is reduced from (3.104) when $\beta = 0$ and $\delta = 0$

$$P_\alpha(x) = e^{-\gamma|x|^\alpha} \tag{3.105}$$

in which case the characteristic function is real and even. The distribution corresponds to the zero-mean Gaussian distribution with variance 2γ when $\alpha = 2$, while it can be the Cauchy distribution when $\alpha = 1$. Except the limiting cases, $\alpha = 1$ and 2, there are no closed expressions for the

density function.[4] The commonly used pdf of the general SαS is given by [270],[258],[273]

$$p_\alpha(x) = \begin{cases} \frac{1}{\pi\gamma^{1/\alpha}} \sum_{k=1}^{\infty} \frac{(-1)^{k-1}}{k!} \Gamma(k\alpha+1) \sin\left(\frac{k\alpha\pi}{2}\right) \left(\frac{|x|}{\gamma^{1/\alpha}}\right)^{-k\alpha-1}, & 0<\alpha<1, x\neq 0 \\ \frac{\gamma}{\pi(x^2+\gamma^2)}, & \alpha = 1 \\ \frac{1}{\pi\alpha\gamma^{1/\alpha}} \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k)!} \Gamma\left(\frac{2k+1}{\alpha}\right) \left(\frac{x}{\gamma^{1/\alpha}}\right)^{2k}, & 1<\alpha<2 \\ \frac{1}{2\sqrt{\gamma\pi}} \exp(-\frac{x^2}{4\gamma}), & \alpha = 2. \end{cases} \tag{3.106}$$

A method generating stable random variables can be found in [274].

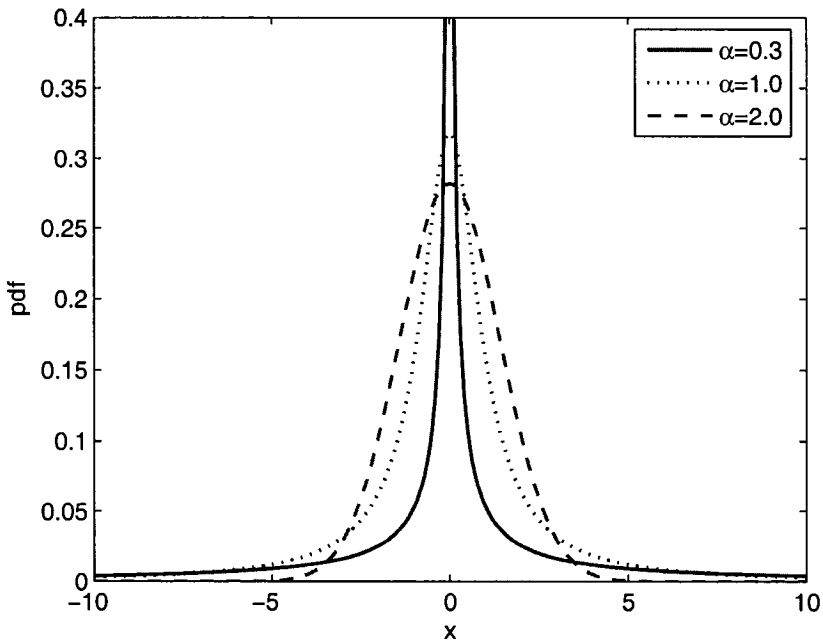


Fig. 3.14 SαS density function ($\gamma = 1$): the density peak of $\alpha = 0.3$ has been cut for better visualization.

Signal Processing of SαS Noise

From the viewpoint of signal processing, the statistics of the system plays a pivotal role to design an optimum receiver, which is usually based on the second-order statistics of the received signals.

[4] When $\alpha = 0.5$ and $\beta = 1$, the Lévy distribution is defined as $p_\alpha(x) = \left(\frac{\gamma}{2\pi}\right)^{1/2} (x-\delta)^{-3/2} \exp\left(-\frac{\gamma}{2(x-\delta)}\right)$ which is concentrated on (δ, ∞), i.e., $p_\alpha(x) = 0$ for $x \in (-\infty, \delta]$.

For example, in a digital communication system the additive white noise is often assumed to be Gaussian, because the finite variance of the Gaussian noise enables the spectral characterization of the system, which is based on the second-order moments. Thus, it is possible to reduce the passband model of the digital communication system to its baseband equivalent. Moreover, it is possible to reduce the digital communication system to a sample-rate, discrete-time equivalent system, with the finite variance assumption of the additive white noise.

When an α-stable random variable is adopted as the additive white noise, the above reduction of the system can not be valid due to its infinite variance. A primary difference between the Gaussian and the α-stable noise distributions is their tails. For $X \sim S_\alpha(\gamma, \beta, 0)$,

$$\lim_{\tau \to \infty} \tau^\alpha P(|X| > \tau) = \gamma C(\alpha) \tag{3.107}$$

where $C(\alpha)$ is a positive constant relying on α. The α-stable distributions have algebraic (i.e., inverse power) tails, while the Gaussian distribution has exponential tail [272]. The log-tail test [275],[273] is a practical method to examine the behavior of the α-stable tails. An important consequence of (3.107) is the nonexistence of the second-order moment of α-stable distributions, except the case $\alpha = 2$. The α-stable random variables exhibit finite absolute moments for orders less than α, i.e.,

$$\mathcal{E}\{|X|^p\} < \infty, \text{ for } 0 \leq p < \alpha. \tag{3.108}$$

In other words, when $p \geq \alpha$ the absolute moments become infinite, which means statistical analysis is not suitable. When $\alpha < 2$, the α-stable distributions present infinite variance, i.e. the second-order moment does not exist, making the use of variance as a measure of dispersion meaningless. Therefore, the standard second- or higher-order statistics can not be applied to the α-stable distributions. Furthermore, signal processing tools such as spectral analysis and least squares techniques, which are grounded on the assumption of finite variance, will be considerably weakened. In this scenario, fractional lower-order statistics (FLOS) [270],[272] can be applied to characterizes the behavior of non-Gaussian α-stable distributions.

In reality, α-stable random variables can not be physically realizable with infinite power because the ideal white noise has constant power spectrum throughout the spectrum. Their properties can be studied after passing through a band-limited system. In practice, all communication systems

operate over a finite dynamic range, as they have finite bandwidth. A technique can be used that provides the communication systems with the finite dynamic range in additive white α-stable noise environments. The nonlinear function, which is an ideal saturation device, is used at the front end of the receiver to limit the received signal within a specific dynamic range. The nonlinear clipping function is given as [273],[276],[277]

$$q(x,G) = \begin{cases} x, & |x| \leq G \\ G \cdot \text{sgn}(x), & \text{otherwise} \end{cases} \tag{3.109}$$

with G being the saturation value of the clipper. The resulting truncated signal possesses finite variance, yet its pdf is identical to that of the unrestricted signal within the range $(-G, G)$. Therefore, the power contained in the tails of the received signal distribution is concentrated at the saturation points in the pdf as Dirac pulses. The relative amplitude of the Dirac pulses on two points, $-G$ and G, depends only on the shape of the distribution within the dynamic range [273]. Any truncated distribution, which is identical to the α-stable distribution within the dynamic range of the receiver, could be acceptable to describe the pdf of the received signals. Some applications of the clipping function can be seen in [273], [277], and [278].

The power of a second-order process has been widely used in signal processing as a standard measure of signal strength. In the case of α-stable distributions, the second-order power is always infinite and thus it is meaningless information on signal strength. To enable signal processing for the class of α-stable processes, an alternative strength measure is necessary. Letting X be a logarithmic-order random variable, the geometric power of X is defined as [258]

$$S_0(X) = e^{\mathcal{E}\{\log|X|\}}. \tag{3.110}$$

The α-stable processes consist of one of the most important infinite-variance families in the logarithmic order class. The geometric power of the SαS random variables defined in (3.105) is given in a closed-form expression [258]

$$S_0 = \frac{(\gamma C_g)^{1/\alpha}}{C_g} \tag{3.111}$$

where $C_g = e^{C_e} \approx 1.78$ is the exponential of the Euler constant $C_e \approx 0.577$.

When the variance of additive white noise is infinite, the standard (second-order) SNR is zero,

and it is totally useless as an indicator of channel conditions. The geometric power allows us to define a universal indicator for constant amplitude signaling. The geometric SNR (GSNR) is defined as [258]

$$\text{GSNR} = 2C_g \left(\frac{A}{S_0}\right)^2 \tag{3.112}$$

where A is the amplitude of a modulated signal in an additive noise channel with noise geometric power S_0. The normalizing constant $2C_g$ guarantees that for the AWGN case ($\alpha = 2$) , the definition of the GSNR coincides with that of the standard SNR with noise variance 2γ. In the design of wireless communication receivers, the GSNR is widely employed with SαS noise environments [277],[278],[279],[280],[281],[282].

Recently, considerable research has been conducted with non-Gaussian SαS noise, which represents additive white noise in practical wireless channels. Bayesian equalization and channel estimation techniques [273],[276] and blind equalization [283] were proposed. Xue and Zhe [284] treated the problem of wireless channel tracking in SαS noise. In multiple access communications, multiuser detection techniques were developed in [278],[277],[281] and [285]. A detector for code acquisition systems of direct-sequence spread-spectrum (DS/SS) communications was proposed in [282]. Rupi *et al.* [286] proposed a robust spatial filtering technique for an inter-signal problem in wireless communications.

Robust filtering algorithms for SαS noise environments were proposed during the last decade. Least mean p-norm (LMP) algorithms were developed in [272],[287],[288], and [289]. Bodenschatz and Nikias [290] proposed median orthogonality as a linear filter criterion and derived a generalized Wiener-Hopf solution. Some linear and nonlinear filtering approaches, including adaptive filtering [291],[292], a nonlinear normalized LMS approach [293] and myriad filter algorithms [294],[295], were proposed. Signal processing applications for SαS signals have also been found. These include an expectation-maximization (EM) algorithm for estimating the source locations, the signal waveforms, and the noise distribution parameters [296], a time-frequency-based detection technique [297], and blind separation of impulsive SαS sources [298].

Part II

Neural Equalization in ISI Channels

Chapter 4

Adaptive Nonlinear Equalization Over Gaussian Channels: EKF Approaches

4.1 Introduction

In digital communications, data symbols are sent through linearly dispersive mediums such as telephone, cable, and wireless. Linear channel distortion in communication systems is caused by two main sources: limited bandwidth and multipath propagation. The linear distortion results in intersymbol interference (ISI) at the receiver, which leads to high error rates if left uncompensated. Hence, an equalization scheme is included in the receiver to compensate for the channel distortion. There are two major approaches to channel equalization [299]. The first approach is to estimate the channel impulse response and use an optimum method such as a maximum *a posteriori* probability or a maximum likelihood sequence estimate to recover the symbols transmitted. The second one is to directly equalize the channel to estimate the symbols, without performing the channel estimation. In this chapter we consider the second approach.

An equalizer in the receiver does a filtering task to mitigate ISI. Many types of neural networks have been successfully applied to communications channel equalization [77]. Among neural network-based equalizers, recurrent neural networks (RNNs) have shown better performance than feedforward neural networks (FNNs), since RNNs approximate infinite impulse response (IIR) filters while FNNs approximate finite impulse response (FIR) filters. The most popular recurrent neural equalizers (RNE) may be a fully connected RNN with real-time recurrent learning (RTRL)

algorithm [105],[300],[106]. Complex versions of the fully connected RNN training employing the RTRL are found in [301],[302],[303],[304] and [305]. Although the RTRL algorithm is popular due to its reasonable complexity in implementation, it is based on a gradient method using *first-order* derivative information and may exhibit inferior capability in terms of convergence speed relative to *second-order* derivative information-based learning techniques such as quasi-Newton, Levenberg-Marquardt, and conjugate gradient techniques.

The extended Kalman filter (EKF) forms the basis of a second-order neural network training method which is a practical and effective alternative to the aforementioned second-order methods. The essence of the recursive EKF procedure is that an approximate covariance matrix that encodes second-order information about the training problem is maintained and evolved during training. Since Singhal and Wu [306] introduced the EKF training algorithm for static FNNs, the EKF has served as the basis for the enhancement of computationally effective neural network training methods that enable the application of FNNs and RNNs to problems in signal processing, pattern classification, and control.

For training the fully connected RNN, Coelho [307] proposed an RTRL-based EKF algorithm to deal with complex-valued signal in channel equalization without decision feedback. However, the RNE employing the fully connected RNN has a problem of instability due to the nature of its nonlinear IIR filter structure, i.e. the leftover of past errors may make it unstable [106],[302]. To overcome the drawback of the RNE without decision feedback, structures of the RNE with decision feedback, so called decision-feedback RNE (DFRNE), have been proposed for both real and complex-valued equalization [106],[302],[303]. The DFRNEs can eliminate the remaining past errors and make the system more stable and robust than the RNE without decision feedback can.

In this chapter, two versions of the EKF algorithm, the global EKF (GEKF) and the decoupled EKF (DEKF) [308],[309], are investigated for training the DFRNE using the fully connected RNN for both real and complex-valued signal processing of channel equalization. Performance of the DFRNE trained with the EKF is evaluated and compared with that of the conventional DFRNE trained with the RTRL algorithm in terms of convergence rate, bit error rate (BER), and tracking capability, for selected nonlinear real and complex-valued communication channels.

4.2 System Model

A general model of a digital communications system with a decision feedback equalizer (DFE) is displayed in Fig. 4.1. It includes both linear and nonlinear distortions. A sequence, $\{s[k]\}$, extracted from a source of information is transmitted, and the transmitted symbols are then corrupted by channel distortion and buried in additive white Gaussian noise (AWGN). The channel with nonlinear distortion is modelled as

$$\begin{aligned} r[k] &= g(\hat{r}[k]) + v[k] \\ &= g\left(\sum_{i=0}^{N-1} h_i s[k-i]\right) + v[k] \end{aligned} \tag{4.1}$$

where $g(\cdot)$ is a nonlinear distortion, h_i is the linear finite impulse response of the channel with length N, $s[k]$ is the sequence of transmitted symbols, and $v[k]$ is the AWGN with zero mean and variance σ_0^2. The DFE is characterized by the three integers, m, n and d, known as the feedforward order, feedback order, and decision delay, respectively. The inputs to the DFE therefore consist of the feedforward inputs $\mathbf{r}[k] = [r[k], r[k-1], \cdots, r[k-m+1]]^T$ and feedback inputs $\mathbf{u}[k] = [u[k-1], \cdots, u[k-n]]^T$. The output of the DFE is $y[k]$ and it is passed through a decision device to determine the estimated symbol $\hat{s}[k-d]$. It is sufficient to use feedback order n [72],[310],

$$n = N + m - d - 2 \tag{4.2}$$

since the transmitted symbols contributing to decision of the equalizer at time k are given by $\mathbf{s}[k] = [s[k], s[k-1], \ldots, s[k-m-N+2]]^T$ for the feedforward order $m = d + 1$.

Throughout this chapter we use the DFRNE using the fully connected RNN as the DFE. An architecture and learning algorithms of the DFRNE will be detailed in the following section.

4.3 Learning Techniques for DFRNE

In this section, dynamic characteristics of the fully connected RNN used as the DFRNE are given in the form of a state-space model of the system. Three learning algorithms of the fully connected RNN, that is RTRL, GEKF and DEKF, are derived in the matrix form based on the state-space

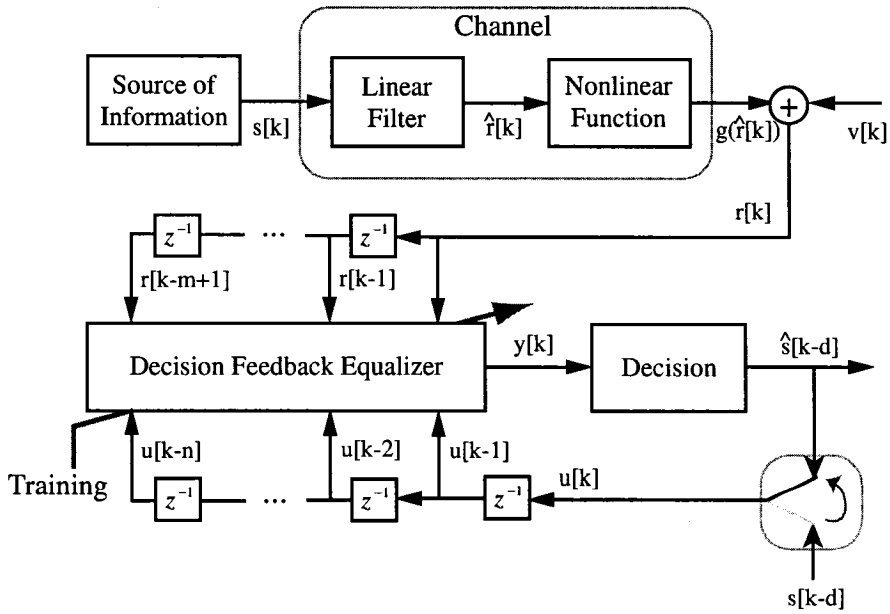


Fig. 4.1 A communications system with decision feedback equalizer. In training mode $s[k-d]$ is provided as training symbols, while $\hat{s}[k-d]$ is fed back to the DFE in decision-directed mode.

model. We describe hereafter complex-valued formulation of algorithms, albeit real-valued algorithms are used in simulations.

4.3.1 State-Space Formulation of Fully Connected RNN

For complex-valued RNN (CRNN), the input and output signals, the weights as well as the activation functions are all *complex* numbers. To emphasize the complex nature of the network, we use the complex notation to express components of the network as follows:

$$\zeta[k] = \zeta^I[k] + j\zeta^Q[k] \tag{4.3}$$

where the superscripts I and Q denote *in-phase* and *quadrature* components (i.e., real and imaginary parts), respectively, $j = \sqrt{-1}$, and ζ can be represented by any component among the inputs $\mathbf{u}$, states $\mathbf{x}$, outputs $\mathbf{y}$, weights $\mathbf{w}$, and activation functions $\varphi(\cdot)$.

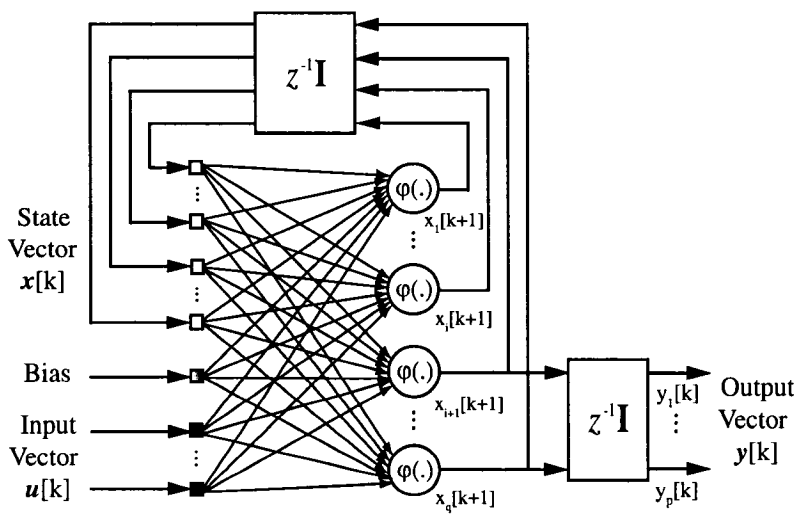


Fig. 4.2 A layout of fully connected recurrent neural network.

Let us consider the fully connected RNN. It consists of q neurons with l external inputs, as shown in Fig. 4.2. Let the q-by-1 vector $\mathbf{x}[k]$ denote the state of the network in the form of a nonlinear discrete-time system, the $(l + 1)$-by-1 vector $\mathbf{u}[k]$ denote the input (including bias)

applied the network[1], and the p-by-1 vector $\mathbf{y}[k]$ denote the output of the network. The dynamic behavior of the network, assumed to be *noise free*, is described by [79]

$$\begin{aligned} \mathbf{x}[k+1] &= \varphi(\mathbf{W}_x[k]\mathbf{x}[k] + \mathbf{W}_u[k]\mathbf{u}[k]) \\ &= \varphi(\mathbf{W}[k]\mathbf{z}[k]) \qquad (4.4) \\ \mathbf{y}[k] &= \mathbf{C}\,\mathbf{x}[k+1] \qquad (4.5) \end{aligned}$$

where $\mathbf{W}_x[k]$ is a q-by-q matrix, $\mathbf{W}_u[k]$ is a q-by-$(l+1)$ matrix, $\mathbf{C}$ is a p-by-q matrix; and $\varphi : \mathbb{R}^q \rightarrow \mathbb{R}^q$ is a diagonal map. The two separate weight matrices can be merged into a whole weight matrix $\mathbf{W}[k]$ with q-by-L dimension where $L = (q + l + 1)$, that is,

$$\mathbf{W}[k] = [\mathbf{W}_x[k]\ \mathbf{W}_u[k]] \qquad (4.6)$$

and the L-by-1 vector $\mathbf{z}[k]$ can be defined as

$$\mathbf{z}[k] = \begin{bmatrix} \mathbf{x}[k] \\ \mathbf{u}[k] \end{bmatrix} \qquad (4.7)$$

where $\mathbf{x}[k]$ is the q-by-1 state vector and $\mathbf{u}[k]$ is the $(l+1)$-by-1 input vector. The first element of $\mathbf{u}[k]$ is unity, which is the bias input, and in a corresponding way, the first column of $\mathbf{W}_u[k]$ is the bias terms applied to the neurons. The dimensionality of the state space, namely q, is the *order* of the system. Therefore the state-space model of Fig. 4.2 is an l-input, q-output recurrent model of order q. Equation (4.4) is the process equation of the model and (4.5) is the measurement equation. The process equation in the state-space description of the network is rewritten in the following form:

$$\mathbf{x}[k+1] = \begin{bmatrix} \varphi(\mathbf{w}_1^H[k]\mathbf{z}[k]) \\ \varphi(\mathbf{w}_2^H[k]\mathbf{z}[k]) \\ \vdots \\ \varphi(\mathbf{w}_q^H[k]\mathbf{z}[k]) \end{bmatrix} \qquad (4.8)$$

where $\varphi(\cdot)$ is an activation function, and the L-by-1 weight vector $\mathbf{w}_i[k]$, which is connected to

[1]When the RNN is used as the DFRNE, the input vector $\mathbf{u}[k]$ includes the received signals from the channel and the decision feedback inputs, as well as the bias input.

the ith neuron in the recurrent network, corresponds to the ith column of the Hermitian transposed weight matrix $\mathbf{W}^H[k]$.

To simplify the presentation of the following learning algorithms, we define new matrices as follows:

- The derivative matrix of the state vector $\mathbf{x}[k]$ with respect to the weight vector $\mathbf{w}_i$ [2]

$$\begin{aligned} \mathbf{\Lambda}_i^{AB}[k] &= \frac{\partial \mathbf{x}^A[k]}{\partial \mathbf{w}_i^B} \\ &\cong \begin{bmatrix} \frac{\partial x_1^A[k]}{\partial w_{i,1}^B} & \frac{\partial x_1^A[k]}{\partial w_{i,2}^B} & \cdots & \frac{\partial x_1^A[k]}{\partial w_{i,L}^B} \\ \frac{\partial x_2^A[k]}{\partial w_{i,1}^B} & \frac{\partial x_2^A[k]}{\partial w_{i,2}^B} & \cdots & \frac{\partial x_2^A[k]}{\partial w_{i,L}^B} \\ \vdots & & \ddots & \vdots \\ \frac{\partial x_q^A[k]}{\partial w_{i,1}^B} & \frac{\partial x_q^A[k]}{\partial w_{i,2}^B} & \cdots & \frac{\partial x_q^A[k]}{\partial w_{i,L}^B} \end{bmatrix} \end{aligned} \tag{4.9}$$

  where $A, B \in \{I, Q\}$ and $\mathbf{\Lambda}_i^{AB}[k]$ has a q-by-L dimension. According to the definition of the derivative matrix given in (4.9), the entire derivative matrix can be represented by

$$\begin{bmatrix} \mathbf{\Lambda}_i^{II}[k] & \mathbf{\Lambda}_i^{IQ}[k] \\ \mathbf{\Lambda}_i^{QI}[k] & \mathbf{\Lambda}_i^{QQ}[k] \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathbf{x}^I[k]}{\partial \mathbf{w}_i^I} & \frac{\partial \mathbf{x}^I[k]}{\partial \mathbf{w}_i^Q} \\ \frac{\partial \mathbf{x}^Q[k]}{\partial \mathbf{w}_i^I} & \frac{\partial \mathbf{x}^Q[k]}{\partial \mathbf{w}_i^Q} \end{bmatrix} \tag{4.10}$$

  where $i = 1, 2, \cdots, q$.

- $\mathbf{Z}[k]$ is a q-by-L matrix whose rows are all zero, except for the ith row that is equal to the transpose of vector $\mathbf{z}[k]$:

$$\mathbf{Z}_i^A[k] = \begin{bmatrix} \mathbf{0}^T \\ (\mathbf{z}^A)^T[k] \\ \mathbf{0}^T \end{bmatrix} \leftarrow i\text{th row}, \quad \forall\ i \tag{4.11}$$

  where $A \in \{I, Q\}$.

[2] The time index of the weight vector $\mathbf{w}$ is $[k-1]$.

- $\mathbf{\Phi}[k]$ is a q-by-q diagonal matrix:

$$\mathbf{\Phi}^A[k+1] = \text{diag}\big[\varphi'((\mathbf{w}_1^A)^T[k]\,\mathbf{z}^A[k]), \varphi'((\mathbf{w}_2^A)^T[k]\,\mathbf{z}^A[k]), \cdots, \varphi'((\mathbf{w}_q^A)^T[k]\,\mathbf{z}^A[k])\big] \tag{4.12}$$

where $A \in \{I, Q\}$ and $\varphi'(\cdot)$ denotes the derivative of $\varphi(\cdot)$.

Letting $\mathbf{a}[k+1] = \mathbf{W}[k]\mathbf{z}[k]$, we get the following components from the cross-coupled signal flow of complex signals

$$\mathbf{a}^I[k+1] = \mathbf{W}^I[k]\mathbf{z}^I[k] - \mathbf{W}^Q[k]\mathbf{z}^Q[k] \tag{4.13}$$

$$\mathbf{a}^Q[k+1] = \mathbf{W}^I[k]\mathbf{z}^Q[k] + \mathbf{W}^Q[k]\mathbf{z}^I[k]. \tag{4.14}$$

With these definitions given above, we derive the derivative submatrices, after differentiating (4.8) and using the *chain rule* of calculus. For example, $\mathbf{\Lambda}_i^{II}[k]$ is defined as

$$\begin{aligned}\mathbf{\Lambda}_i^{II}[k+1] &= \frac{\partial \mathbf{x}^I[k+1]}{\partial \mathbf{w}_i^I[k]} \\ &= \frac{\partial \boldsymbol{\varphi}(\mathbf{a}^I[k+1])}{\partial \mathbf{w}_i^I[k]} \\ &= \frac{\partial \boldsymbol{\varphi}(\mathbf{a}^I[k+1])}{\partial \mathbf{a}^I[k+1]} \frac{\partial \mathbf{a}^I[k+1]}{\partial \mathbf{w}_i^I[k]}\end{aligned} \tag{4.15}$$

where the two derivatives are represented as

$$\begin{aligned}\frac{\partial \boldsymbol{\varphi}(\mathbf{a}^I[k+1])}{\partial \mathbf{a}^I[k+1]} &= \text{diag}\left[\varphi'(a_1^I[k+1])\ \varphi'(a_1^I[k+1]) \cdots \varphi'(a_q^I[k+1])\right] \\ &= \mathbf{\Phi}^I[k+1]\end{aligned} \tag{4.16}$$

$$\frac{\partial \mathbf{a}^I[k+1]}{\partial \mathbf{w}_i^I[k]} = \mathbf{W}_x^I[k]\mathbf{\Lambda}_i^{II}[k] - \mathbf{W}_x^Q[k]\mathbf{\Lambda}_i^{QI}[k] + \mathbf{Z}_i^I[k]. \tag{4.17}$$

Therefore, the submatrix $\mathbf{\Lambda}_i^{II}[k]$ may be represented as follows:

$$\mathbf{\Lambda}_i^{II}[k+1] = \mathbf{\Phi}^I[k+1]\big[\mathbf{W}_x^I[k]\mathbf{\Lambda}_i^{II}[k] - \mathbf{W}_x^Q[k]\mathbf{\Lambda}_i^{QI}[k] + \mathbf{Z}_i^I[k]\big]. \tag{4.18}$$

Other submatrices, $\mathbf{\Lambda}_i^{IQ}$, $\mathbf{\Lambda}_i^{QI}$, and $\mathbf{\Lambda}_i^{QQ}$, are obtained from a repetition of the above procedures.

Finally, we obtain the following recursive equation

$$\begin{bmatrix} \mathbf{\Lambda}_i^{II}[k+1] & \mathbf{\Lambda}_i^{IQ}[k+1] \\ \mathbf{\Lambda}_i^{QI}[k+1] & \mathbf{\Lambda}_i^{QQ}[k+1] \end{bmatrix} = \begin{bmatrix} \mathbf{\Phi}^I[k+1] & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi}^Q[k+1] \end{bmatrix} \times \left(\begin{bmatrix} \mathbf{W}_x^I[k] & -\mathbf{W}_x^Q[k] \\ \mathbf{W}_x^Q[k] & \mathbf{W}_x^I[k] \end{bmatrix} \times \begin{bmatrix} \mathbf{\Lambda}_i^{II}[k] & \mathbf{\Lambda}_i^{IQ}[k] \\ \mathbf{\Lambda}_i^{QI}[k] & \mathbf{\Lambda}_i^{QQ}[k] \end{bmatrix} + \begin{bmatrix} \mathbf{Z}_i^I[k] & -\mathbf{Z}_i^Q[k] \\ \mathbf{Z}_i^Q[k] & \mathbf{Z}_i^I[k] \end{bmatrix} \right). \quad (4.19)$$

4.3.2 RTRL Algorithm

To proceed with the description of the complex-valued RTRL (CRTRL) process, let us define a cost function $\mathcal{J}[k]$, which is the instantaneous sum of squared errors,

$$\begin{aligned} \mathcal{J}[k] &= \frac{1}{2}\mathbf{e}^H[k]\,\mathbf{e}[k] \\ &= \frac{1}{2}\left[(\mathbf{e}^I)^T[k]\mathbf{e}^I[k] + (\mathbf{e}^Q)^T[k]\mathbf{e}^Q[k]\right] \end{aligned} \quad (4.20)$$

where the p-by-1 error vector $\mathbf{e}[k]$ is defined by using the measurement equation given in (4.5):

$$\begin{aligned} \mathbf{e}[k] &= \tilde{\mathbf{y}}[k] - \mathbf{y}[k] \\ &= \tilde{\mathbf{y}}[k] - \mathbf{C}\mathbf{x}[k+1] \\ &= \mathbf{e}^I[k] + j\mathbf{e}^Q[k] \end{aligned} \quad (4.21)$$

where $\tilde{\mathbf{y}}[k]$ denotes the desired output vector. The objective of the learning process is to minimize the cost function $\mathcal{J}[k]$. Now, we go to the stage of the weights updating. The updating equation of the weights is written by

$$\begin{aligned} \mathbf{w}_i[k+1] &= \mathbf{w}_i[k] + \Delta\mathbf{w}_i[k] \\ &= \mathbf{w}_i[k] - \eta\nabla_{\mathbf{w}_i}\mathcal{J}[k] \end{aligned} \quad (4.22)$$

where η is the *learning rate* and the gradient of the cost function $\mathcal{J}[k]$ with respect to the weight vector $\mathbf{w}_i[k]$, $\nabla_{\mathbf{w}_i}\mathcal{J}[k]$, is defined as

$$\nabla_{\mathbf{w}_i}\mathcal{J}[k] = \frac{\partial\mathcal{J}[k]}{\partial\mathbf{w}_i^I[k]} + j\frac{\partial\mathcal{J}[k]}{\partial\mathbf{w}_i^Q[k]}. \quad (4.23)$$

The real and imaginary parts of (4.23) can be obtained as follows, respectively:

$$\frac{\partial \mathcal{J}[k]}{\partial \mathbf{w}_i^I[k]} = -\big((\mathbf{e}^I)^T[k]\mathbf{C}\mathbf{\Lambda}_i^{II}[k] + (\mathbf{e}^Q)^T[k]\mathbf{C}\mathbf{\Lambda}_i^{QI}[k]\big) \tag{4.24}$$

$$\frac{\partial \mathcal{J}[k]}{\partial \mathbf{w}_i^Q[k]} = -\big((\mathbf{e}^I)^T[k]\mathbf{C}\mathbf{\Lambda}_i^{IQ}[k] + (\mathbf{e}^Q)^T[k]\mathbf{C}\mathbf{\Lambda}_i^{QQ}[k]\big). \tag{4.25}$$

The adjustment of the weight vector of neuron i in the network is determined by the following form

$$\begin{aligned}\Delta\mathbf{w}_i[k] &= \Delta\mathbf{w}_i^I[k] + j\Delta\mathbf{w}_i^Q[k] \\ &= \eta\left[\begin{bmatrix}(\mathbf{e}^I)^T[k]\mathbf{C} & (\mathbf{e}^Q)^T[k]\mathbf{C}\end{bmatrix}\begin{bmatrix}\mathbf{\Lambda}_i^{II}[k] & \mathbf{\Lambda}_i^{IQ}[k] \\ \mathbf{\Lambda}_i^{QI}[k] & \mathbf{\Lambda}_i^{QQ}[k]\end{bmatrix}\right]\begin{bmatrix}1 \\ j\end{bmatrix}.\end{aligned} \tag{4.26}$$

To start the learning process, the learning rate η is chosen to guarantee stable convergence, and the *initial conditions* of the derivative submatrices are set to

$$\mathbf{\Lambda}_i^{AB}(0) = \mathbf{0}, \quad \forall\, i \text{ and } A, B \in \{I, Q\}. \tag{4.27}$$

The implication of the initial conditions means that initially the recurrent network resides in a constant state [79].

For the real-valued RTRL algorithm, the matrix $\mathbf{\Lambda}_i$ and the adjustment of the weight vector of the ith neuron $\Delta\mathbf{w}_i[k]$ are given by, respectively

$$\mathbf{\Lambda}_i[k+1] = \mathbf{\Phi}[k+1]\big[\mathbf{W}_x[k]\mathbf{\Lambda}_i[k] + \mathbf{Z}_i[k]\big] \tag{4.28}$$

$$\Delta\mathbf{w}_i[k] = \eta\mathbf{C}\mathbf{\Lambda}_i[k]\mathbf{e}[k]. \tag{4.29}$$

4.3.3 GEKF Algorithm

To enable the EKF for training the complex-valued RNN, the network's behavior can be recast as the following nonlinear discrete-time system:

$$\mathbf{w}[k+1] = \mathbf{w}[k] + \boldsymbol{\omega}[k] \tag{4.30}$$

$$\mathbf{y}[k] = \mathbf{C}\,\boldsymbol{\varphi}\,(\mathbf{w}[k], \mathbf{z}[k]) + \boldsymbol{\nu}[k] \tag{4.31}$$

where all vectors and matrix are complex numbers, and the weight vector $\mathbf{w}[k]$ is defined by

$$\mathbf{w}[k] = \begin{bmatrix} \mathbf{w}_1[k] \\ \mathbf{w}_2[k] \\ \vdots \\ \mathbf{w}_q[k] \end{bmatrix} \tag{4.32}$$

where $\mathbf{w}_i[k](i = 1, 2, \cdots, q)$ is the i^{th} column of the Hermitian transposed (or complex conjugated and transposed) weight matrix $\mathbf{W}^H[k]$. Equation (4.30), known as the process equation, specifies that the state of the system is given by the network's weight parameter values $\mathbf{w}[k]$ and is characterized as a stationary process corrupted by process noise $\boldsymbol{\omega}[k]$. The measurement equation, given in (4.31), represents the network's output vector $\mathbf{y}[k]$ as the nonlinear function $\boldsymbol{\varphi}(\cdot)$ of the weight vector $\mathbf{w}[k]$ and the vector $\mathbf{z}[k]$ which includes both the input vector $\mathbf{u}[k]$ and the recurrent node activations $\mathbf{x}[k]$. This equation includes a random measurement noise vector $\boldsymbol{\nu}[k]$. The process noise $\boldsymbol{\omega}[k]$ is typically characterized as zero-mean, white noise with covariance given by $\mathcal{E}[\boldsymbol{\omega}_i\boldsymbol{\omega}_j^H] = \delta_{ij}\mathbf{Q}[k]$[3]. Similarly, the measurement noise $\boldsymbol{\nu}[k]$ is also characterized as zero-mean, white noise with covariance given by $\mathcal{E}[\boldsymbol{\nu}_i\boldsymbol{\nu}_j^H] = \delta_{ij}\mathbf{R}[k]$.

To prepare the application of the GEKF to the state-space model, the nonlinear function $\boldsymbol{\varphi}(\cdot)$ given in (4.31), must be linearized and the linearized model has the q-by-L derivative (or *Jacobian*) matrix $\boldsymbol{\Lambda}[k]$ for the q neurons and the L weights of the system. By Cauchy-Riemann equations [12], the complex Jacobian $\boldsymbol{\Lambda}[k]$ is defined as the partial derivatives of the q neurons with respect to the L complex weights as given by

$$\begin{aligned} \boldsymbol{\Lambda}[k+1] &= \frac{\partial \boldsymbol{\varphi}(\mathbf{a}[k+1])}{\partial \mathbf{w}[k]} \\ &= \nabla_{\mathbf{w}}\mathbf{x}[k+1] \\ &= \frac{\partial \mathbf{x}[k+1]}{\partial \mathbf{w}^I[k]} + j\frac{\partial \mathbf{x}[k+1]}{\partial \mathbf{w}^Q[k]} \\ &= \left(\boldsymbol{\Lambda}^{II}[k+1] + \boldsymbol{\Lambda}^{QQ}[k+1]\right) + j\left(\boldsymbol{\Lambda}^{QI}[k+1] - \boldsymbol{\Lambda}^{IQ}[k+1]\right). \end{aligned} \tag{4.33}$$

In general, the GEKF solution to the parameter estimation problem is given by the following

[3] δ denotes the Kronecker delta.

recursion:

$$\mathbf{\Gamma}[k] = [\mathbf{\Lambda}[k]\mathbf{P}[k]\mathbf{\Lambda}^H[k] + \mathbf{R}[k]]^{-1} \tag{4.34}$$

$$\mathbf{K}[k] = \mathbf{P}[k]\mathbf{\Lambda}^H[k]\mathbf{\Gamma}[k] \tag{4.35}$$

$$\hat{\mathbf{w}}[k+1] = \hat{\mathbf{w}}[k] + \mathbf{K}[k]\boldsymbol{\xi}[k] \tag{4.36}$$

$$\mathbf{P}[k+1] = \mathbf{P}[k] - \mathbf{K}[k]\mathbf{\Lambda}[k]\mathbf{P}[k] + \mathbf{Q}[k]. \tag{4.37}$$

The parameter vectors and signal vectors in equations (4.34) to (4.37) are described as follows:

$\mathbf{\Gamma}[k]$: q-by-q global scaling matrix;

$\mathbf{K}[k]$: L-by-q Kalman gain matrix;

$\boldsymbol{\xi}[k]$: q-by-1 vector which must be based on the error vector given in (4.21);

$\hat{\mathbf{w}}[k]$: estimate of the L-by-1 weight vector $\mathbf{w}[k]$;

$\mathbf{R}[k]$: q-by-q measurement covariance matrix;

$\mathbf{Q}[k]$: L-by-L process noise covariance matrix;

$\mathbf{P}[k]$: L-by-L approximate error covariance matrix.

The estimate, $\hat{\mathbf{w}}[k]$, is a function of the Kalman gain matrix $\mathbf{K}[k]$. In the GEKF algorithm, the measurement and process noise covariance matrices, $\mathbf{R}[k]$ and $\mathbf{Q}[k]$, are specified for all training instances, and the approximate error covariance matrix $\mathbf{P}[k]$ is initialized at the beginning of training as follows:

$$\mathbf{R}[0] = \mu^{-1}\,(\mathbf{I} + j\mathbf{I}) \tag{4.38}$$

$$\mathbf{Q}[0] = \rho\,(\mathbf{I} + j\mathbf{I}) \tag{4.39}$$

$$\mathbf{P}[0] = \epsilon^{-1}\,(\mathbf{I} + j\mathbf{I}) \tag{4.40}$$

where the parameters μ, ρ and ϵ are to be set.

For the real-valued GEKF, the in-phase component of the Jacobian $\mathbf{\Lambda}[k]$ in (4.33) is evaluated as follows

$$\mathbf{\Lambda}[k] = \mathbf{\Lambda}^{II}[k]. \tag{4.41}$$

Furthermore, all matrices and vectors in equations (4.34) to (4.37) are real values.

4.3.4 DEKF Algorithm

The computational requirements of the GEKF are dominated by the need to store and update the approximate error covariance matrix $\mathbf{P}[k]$ at each time step. For the network with q neurons and L weights, the computational complexity of the GEKF is $\mathcal{O}(qL^2)$ and its storage requirements are $\mathcal{O}(L^2)$. In contrast, the computational complexity and storage requirements of the DEKF become $\mathcal{O}(q^2L + q\sum_{i=1}^{q} L_i^2)$ and $\mathcal{O}(\sum_{i=1}^{q} L_i^2)$, where L_i is the number of weights in group i, respectively. It is noted that the computational complexity and storage requirements of the DEKF can be significantly less than those of the GEKF. We show an example of the effect of decoupling on the structure of the approximate covariance matrix in Fig. 4.3. For the DEKF, as shown in Fig. 4.3(b), only the shaded portions, *i.e.* block-diagonal submatrices, are updated and maintained, while the whole matrix depicted in Fig. 4.3(a) is updated and saved for the GEKF.

From (4.33), the complex sub-Jacobian $\mathbf{\Lambda}_i[k]$ for the DEKF is given by

$$\mathbf{\Lambda}_i[k+1] = \left(\mathbf{\Lambda}_i^{II}[k+1] + \mathbf{\Lambda}_i^{QQ}[k+1]\right) + j\left(\mathbf{\Lambda}_i^{QI}[k+1] - \mathbf{\Lambda}_i^{IQ}[k+1]\right). \tag{4.42}$$

Therefore, the complex DEKF algorithm with neuron decoupling is given by

$$\mathbf{\Gamma}[k] = \left[\sum_{i=1}^{q} \mathbf{\Lambda}_i[k]\mathbf{P}_i[k]\mathbf{\Lambda}_i^H[k] + \mathbf{R}[k]\right]^{-1} \tag{4.43}$$

$$\mathbf{K}_i[k] = \mathbf{P}_i[k]\mathbf{\Lambda}_i^H[k]\mathbf{\Gamma}[k] \tag{4.44}$$

$$\hat{\mathbf{w}}_i[k+1] = \hat{\mathbf{w}}_i[k] + \mathbf{K}_i[k]\boldsymbol{\xi}[k] \tag{4.45}$$

$$\mathbf{P}_i[k+1] = \mathbf{P}_i[k] - \mathbf{K}_i[k]\mathbf{\Lambda}_i[k]\mathbf{P}_i[k] + \mathbf{Q}_i[k] \tag{4.46}$$

where $\mathbf{K}_i[k]$ represents the L_i-by-q Kalman gain matrix, in which L_i denotes the number of weights connected to group i. $\hat{\mathbf{w}}_i[k]$ is the estimate of the L_i-by-1 weight vector $\mathbf{w}[k]$. In the DEKF

algorithm, the process noise covariance matrices of each group, $\mathbf{Q}_i[k]$, are specified for all training instances, and the approximate error covariance matrices of each group, $\mathbf{P}_i[k]$, are initialized at the beginning of training as follows:

$$\mathbf{Q}_i[0] = \rho\,(\mathbf{I} + j\mathbf{I}) \tag{4.47}$$

$$\mathbf{P}_i[0] = \epsilon^{-1}\,(\mathbf{I} + j\mathbf{I})\,. \tag{4.48}$$

where $\mathbf{Q}_i$ and $\mathbf{P}_i$ have L_i-by-L_i dimension. Initialization of the measurement noise covariance matrix, $\mathbf{R}[0]$, is given in (4.38). As the real GEKF algorithm, the real DEKF algorithm is evaluated using real-valued matrices and vectors.

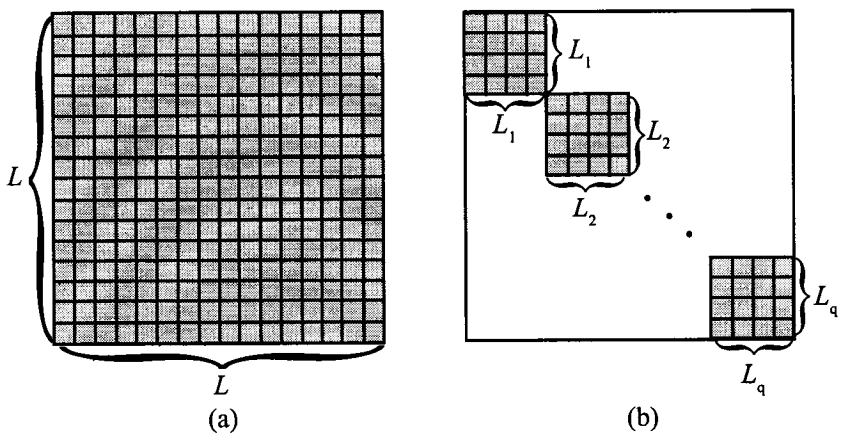


Fig. 4.3 Covariance matrix **P**: (a) GEKF and (b) DEKF.

4.3.5 Activation Functions

There are two classes of complex activation functions: Separate (or split) activation functions and fully complex activation functions [311]. As binary (for instance, BPSK and 2-PAM), 4-QAM and QPSK modulation schemes are applied, using hyperbolic tangent functions which are saturated in ± 1 is appropriate. However, for M-level modulation schemes (where $M > 4$) other specified activation functions are needed according to the level M. In this chapter, only separate activation

functions are considered, and when 4-QAM modulation is employed in simulations it is defined as

$$\begin{aligned}\varphi(a) &= \varphi(a^I) + j\varphi(a^Q) \\ &= \tanh(ba^I) + j\tanh(ba^Q)\end{aligned} \tag{4.49}$$

where the constant b is unity. When in simulations a real-valued channel is considered and BPSK modulation is applied, a real-valued activation function is simply given by $\varphi(a) = \tanh(ba)$.

4.4 Performance Evaluation

Through channel equalization experiments, the performance of the DFRNE trained with two versions of the EKF algorithm is evaluated and compared with that of the conventional DFRNE trained with the RTRL algorithm. In this section, two different properties of the DFRNEs are evaluated. One is the rate of convergence and the other is the tracking capability; convergence is a transient phenomenon, in contrast with tracking, which is a steady-state phenomenon [12].

In the following simulations, we set the feedforward order $m = 3$ and the feedback order $n = 2$. The number of neurons is set to $q = 2$ for the DFRNEs, in which the number of output neurons p is always one for the purpose of equalization. Henceforth, the error $\mathbf{e}[k]$ given in (4.21) can be scalar and in turn the vector $\boldsymbol{\xi}[k]$, used in (4.36) and (4.45), can be equal to $\mathbf{e}[k]$. The weight vector is initialized to small random real (or complex) values with $|\mathbf{w}(0)| < 10^{-3}$ for real (or complex) channels. Information symbols from uniformly distributed BPSK or 4-QAM signals in presence of ISI and AWGN are used in the simulations.

4.4.1 Convergence Rate

Channel Model 1: Real-Valued Nonlinear Channel

A linear channel model with a nonminimum phase has the transfer function:

$$H_1(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2}$$

which is used for channel equalization in [105],[104] and [81]. This channel has a deep spectral null at high frequency which may be encountered in radio transmission. The nonlinear channel is

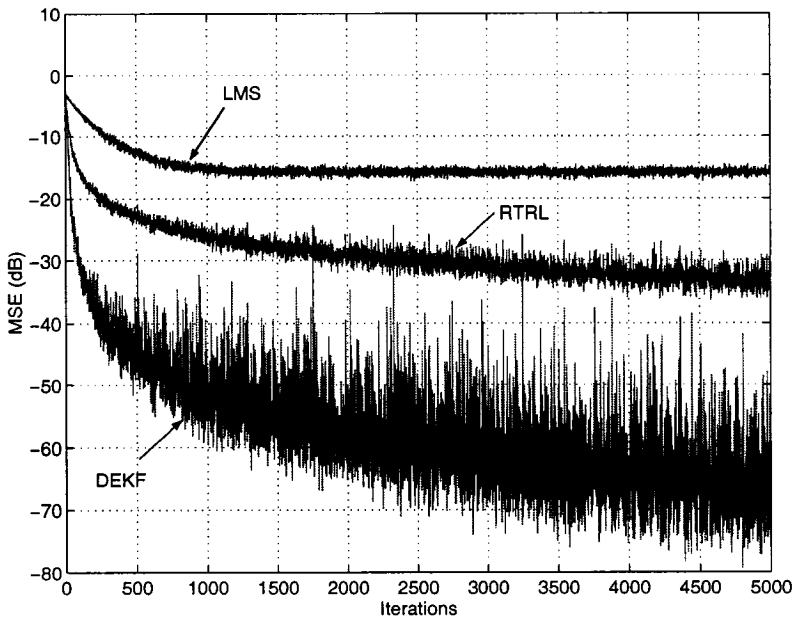
modeled as

$$r[k] = \tanh(\hat{r}[k]) + v[k]$$

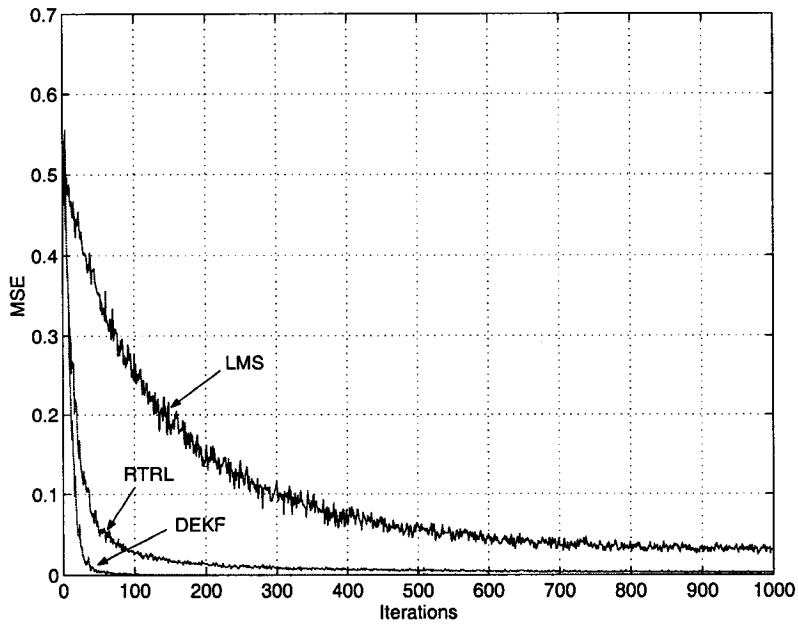
where a nonlinearity is applied to the output of the linear channel. This nonlinear distortion of the channel may take into account saturation effects due to transmission amplifiers. The learning rate of the RTRL is chosen empirically as $\eta = 0.1$ and this value ensures a stable convergence. The parameters for the GEKF and the DEKF are chosen empirically as $\mu = 0.1$, $\rho = 0.01$, and $\epsilon = 0.01$. The decision delay is $d = 2$.

The convergence properties of the DFEs used in the simulations are depicted in Fig. 4.4, with log and linear scales of MSE values. For the purpose of comparison, we show the result of an LMS-based DFE which has nine feedforward and seven feedback filter taps, albeit it is well known that the LMS-DFE is very slow to converge and produces poor performance for nonlinear channels. These results are ensemble-averaged over 200 independent runs. Each run has a different BPSK random sequence and random initial weights for all DFEs and is performed at a SNR of 14 dB. We observe that the MSE curves of the GEKF and the DEKF are indistinguishable. Thus we display only that of the DEKF. Both the GEKF and the DEKF outperform the RTRL in terms of convergence speed. For instance, the MSE value of the DEKF reaches around -50 dB after 10^3 training symbols, while MSE values of the LMS and the RTRL reach -15 and -27 dB, respectively. As shown in Fig. 4.4(b), the DEKF reaches steady state before 100 iterations. These results confirm that the two versions of the EKF algorithm provide an improvement with regard to both the convergence speed and the steady-state MSE. Fig. 4.5 shows the bit-error rate (BER) performance, averaged over 100 independent trials. In each trial, the first 100 symbols are used for training and the next 10^4 symbols are used for testing. The weight vectors of the DFRNEs are frozen after the training, and the transmission symbols are evaluated at the decision-directed mode. The GEKF and DEKF algorithms show exactly the same BER performance, and they attain about 1.3 dB of improvement over the RTRL at 10^{-4} of BER, while the LMS-based DFE does not recover the transmitted symbols. We have observed that the RTRL requires more than 200 training symbols to achieve the same BER performance of the GEKF and the DEKF.

Channel Model 2: Complex-Valued Nonlinear Channel



(a) y-axis: Log scale (10 log_{10}(MSE))



(b) y-axis: Linear scale

Fig. 4.4 Convergence properties for Channel Model 1 under SNR=14dB.

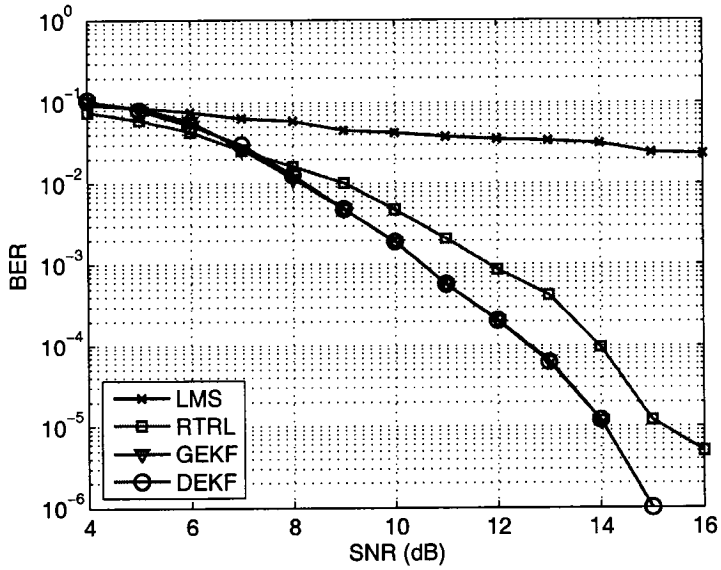


Fig. 4.5 BER performance for Channel Model 1 using 100 training symbols.

A nonminimum-phase complex-valued channel [304] is

$$H_2(z) = 0.2512 + (0.7071 + j0.7071)z^{-1}.$$

With a nonlinear distortion, the input signal of the DFRNE is given by [304]

$$r[k] = \hat{r}[k] + 0.02\hat{r}^2[k] + 0.005\hat{r}^3[k] + v[k].$$

In experiments, 4-QAM modulation is applied to this nonlinear complex channel. Complex algorithms, that is CRTRL, CGEKF and CDEKF, are used to train the complex DFRNE. The learning rate of the CRTRL is chosen empirically as $\eta = 0.1$. The parameters for the CGEKF and the CDEKF are chosen empirically as $\mu = 1.0$, $\rho = 200$, and $\epsilon = 0.1$. The decision delay is set to $d = 1$.

Each simulation result is generated by only one trial, i.e., one block transmission, in contrast to the ensemble-averaged results of Channel Model 1. Since, as noted before, convergence explains a transient response of the DFRNEs, pattern classification characteristics of the DFRNEs can be displayed through watching the outputs obtained from only one block of data. For a data block

which consists of 3×10^3 symbols, the generated convergence curves are depicted in Fig. 4.6 for an SNR of 14 dB. As shown in Fig. 4.6(b), the CGEKF and the CDEKF reach lower MSE values more rapidly than the CRTRL. As in Channel Model 1, the two versions of the complex EKF algorithm attain fast convergence speed and a lower MSE level. To show the pattern classification properties, eye diagrams of the received signals and the equalized outputs during training mode are used and displayed in Fig. 4.7(a) until 10^3 training symbols. After training, 2×10^3 symbols are tested during decision-directed mode as shown in Fig. 4.7(b). We note that the CGEKF and the CDEKF have much better pattern classification ability compared with the CRTRL.

4.4.2 Tracking Capability

Channel tracking performance of the DFRNEs operating in a nonstationary environment is tested for a particular time-varying channel. A nonlinear channel with time-varying coefficients is considered here.

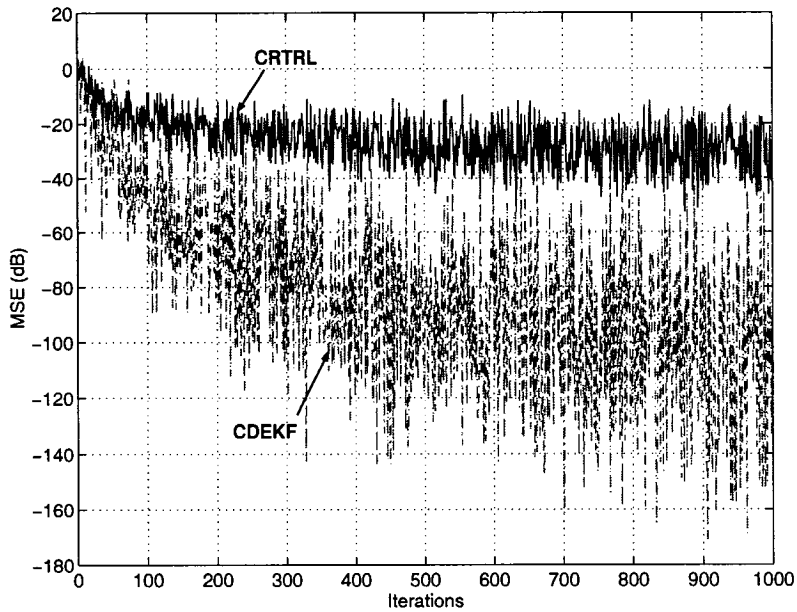
Channel Model 3: Nonlinear Time-Varying Channel

A time-varying discrete-time channel is described by

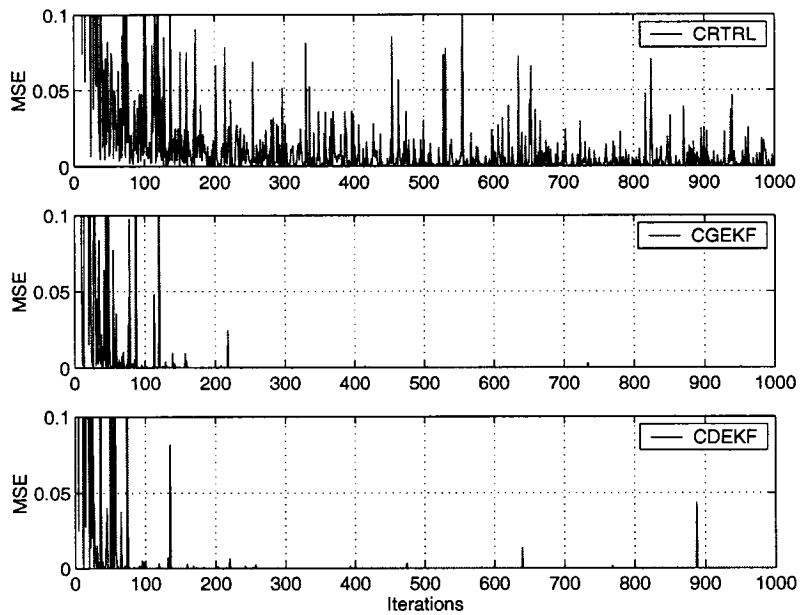
$$H_3(z) = 1.0 + a_1[k]z^{-1} + a_2[k]z^{-2}.$$

The nonlinear distortion employed in Channel Model 1 is applied to this channel. This channel model represents a nonlinear time-varying channel with $a_i[k](i = 1, 2)$ varying with time k. These time-varying coefficients are generated by convolving white Gaussian noise and a Butterworth filter response. The bandwidth of the Butterworth filter determines the relative bandwidth (fading rate) of the channel. A nominal 2 kHz channel with 2400 symbols/s sampling rate are assumed, and a second-order Butterworth filter having a 3 dB bandwidth of 0.5 Hz is used [38].

The parameters are set to the same values as those used in Channel Model 1. As expected, the GEKF provides faster channel tracking capabilities than the corresponding RTRL. Fig. 4.8(a) shows the time-varying coefficients $a_1[k]$ and $a_2[k]$ drawn for a fading rate of 0.5 Hz. For the DFRNEs to practice their tracking capability, they first pass from the transient phase to the steady-state phase of operation, and there should be provision for continuous update of the weights of the DFRNEs. The DFRNEs are in training phase until $k = 2000$ and then they are switched to tracking

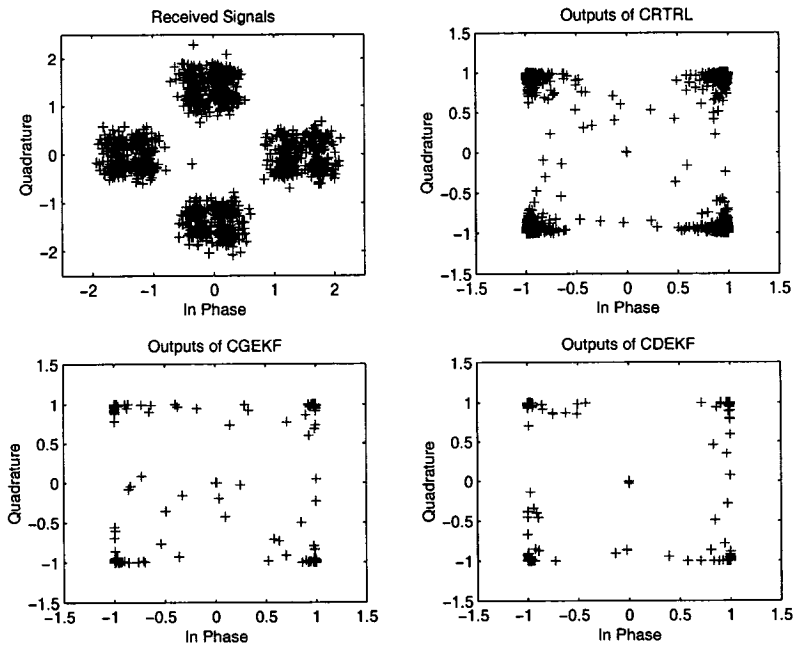


(a) y-axis: Log scale (10 log_{10}(MSE))

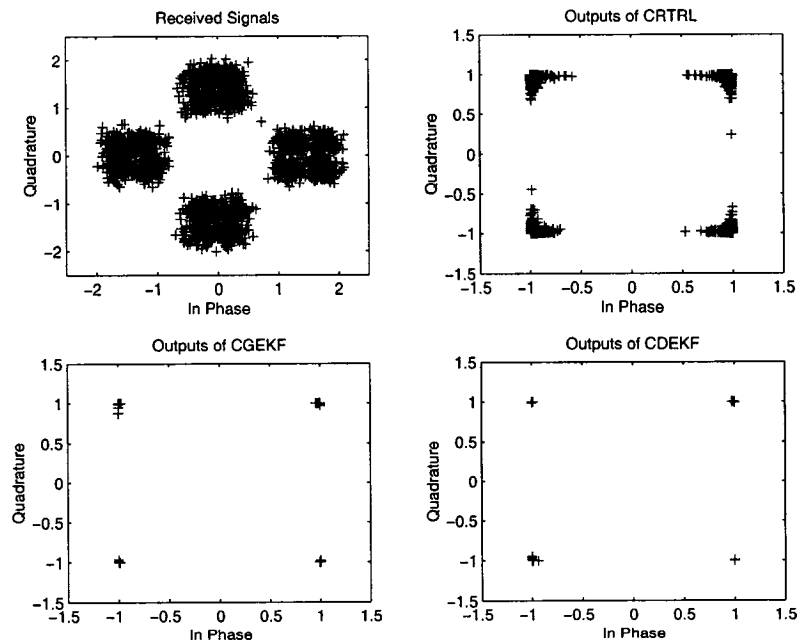


(b) y-axis: Linear scale

Fig. 4.6 Convergence properties for Channel Model 2 under SNR=14dB.



(a) Training mode



(b) Decision-directed mode

Fig. 4.7 Scatter plots for Channel Model 2 under SNR=14dB.

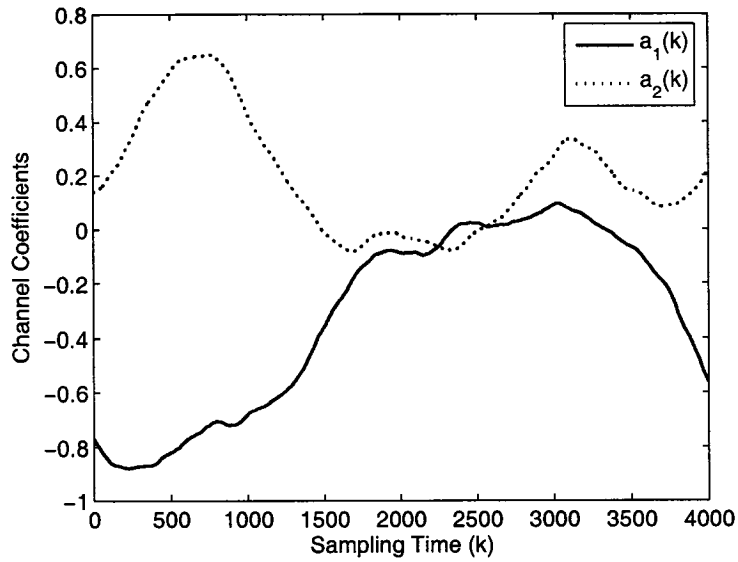
phase at $k = 2001$. Unlike simulations for Channel Model 1 and 2, the DFRNEs still update their weight vectors during testing (tracking) phase in order to track the fading characteristics of the channel. In Fig. 4.8(b), the channel tracking property is evaluated in both training phase and the decision-directed tracking phase at an SNR of 15 dB. This result verifies that the MSE value of the GEKF is much lower than that of the RTRL for both training and tracking phases.

Fig. 4.9 shows eye diagrams during decision-directed tracking mode for 2×10^3 symbols. The equalized outputs of the GEKF and DEKF have no spots near the decision boundary. In contrast, some of the RTRL's equalized outputs are located in the decision boundary, which creates wrong symbol detections. BER performance is illustrated in Fig. 4.10. It is averaged over 100 independent trials, where 100 training symbols and 10^3 testing symbols are supplied. BER performance of the GEKF and the DEKF is superior to the RTRL at high SNR area (SNR $>$ 14 dB) which is the typical operating range of digital transmission systems [312], while at the area of low SNR the RTRL shows better BER performance.

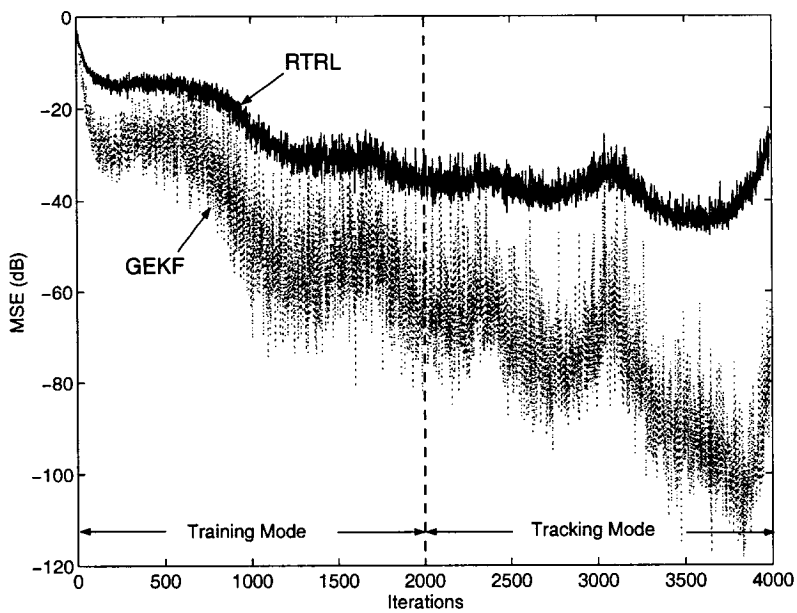
4.5 Concluding Remarks

For the AWGN channels, learning algorithms (the GEKF and the DEKF) based on the RTRL were derived for the complex-valued DFRNE, and their performance was tested through channel equalization experiments with nonlinear distortions. With regard to convergence rate, both the GEKF and the DEKF which are second-order derivatives information-based learning techniques outperform the RTRL, which uses first-order information in the learning process. This means that the fast convergence rate of the GEKF and the DEKF requires less training symbols and give better BER and pattern classification performance than the RTRL technique. In terms of channel tracking ability compared with the RTRL, the two forms of the EKF algorithm have shown rapid tracking properties for the channel with time-varying coefficients in both the training mode and the decision-directed tracking mode. For SNR higher than 14 dB the superiority of the GEKF and the DEKF algorithms compared with the RTRL was consistent in convergence rate, pattern classification, channel tracking as well as BER performance.

The EKF algorithms nevertheless have some drawbacks: the computational complexity and parameter selection. First, the computational cost of the EKF algorithms is relatively expensive.



(a) Time-varying coefficients at 0.5Hz.



(b) Convergence at SNR=15dB.

Fig. 4.8 Channel tracking capability for Channel Model 3.

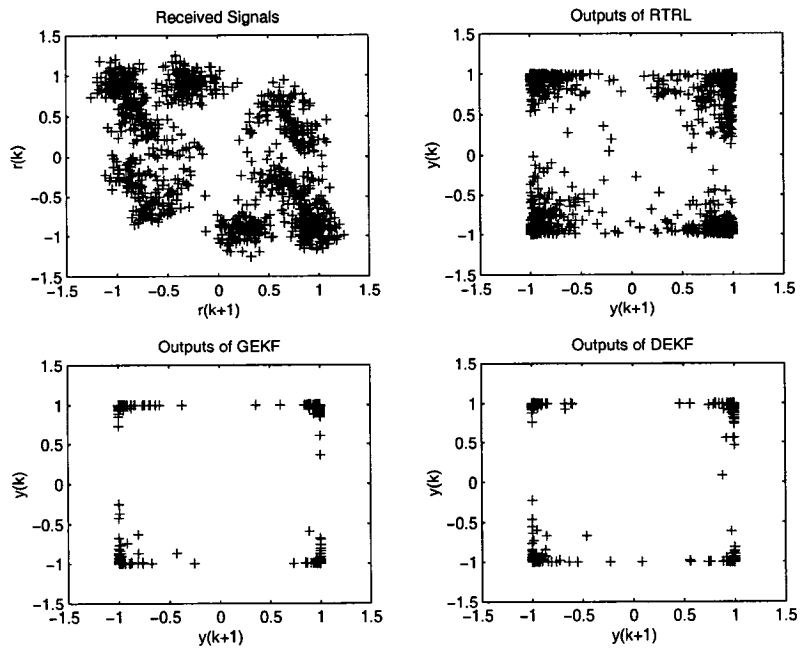


Fig. 4.9 Scatter plots for Channel Model 3 during tracking mode (SNR=15dB).

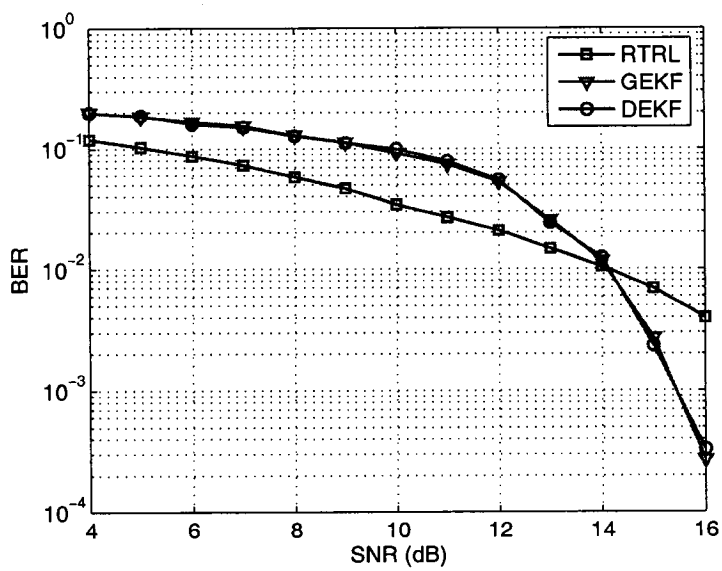


Fig. 4.10 BER performance for Channel Model 3.

Hence the DEKF is recommended for the DFRNE, since the GEKF and the DEKF have shown (almost) the same performance. Next, the EKF algorithms are more sensitive to initial parameter values than the RTRL. Careful parameter selection is therefore required in using the EKF algorithms.

The design and learning algorithms of adaptive nonlinear equalizers undertaken in this Chapter are only restricted to AWGN channel environments. For non-Gaussian noise channels, we will develop a recurrent neural equalization scheme using the unscented Kalman filter (UKF).

Chapter 5

Adaptive Nonlinear Equalization Over Non-Gaussian Channels: UKF Approach

5.1 Introduction

Information-bearing signals transmitted over practical communication channels such as acoustic underwater channels and indoor wireless channels are contaminated by noise. The channel noise is known to be impulsive, leading to non-Gaussian channels. The major factors limiting performance are non-Gaussian noise and time-varying multi-path propagation. Furthermore, multi-path propagation increases intersymbol interference (ISI) and causes frequency dependent fading. Since most communication receivers are designed for additive white Gaussian noise (AWGN), the application of such receivers in white non-Gaussian noise environments may not be suitable. Designing a receiver for additive non-Gaussian noise is without a doubt an intriguing topic.

In digital communication systems, an equalizer at the receiver is used to mitigate the effect of ISI. The Bayesian approach known as the maximum *a posteriori* probability (MAP) equalizer is the optimum solution [72],[58]. A Bayesian decision feedback equalizer (DFE) for a non-Gaussian α-stable environment was proposed in [276]. The performance of the optimum Bayesian DFE depends on two sets of parameters: the equalizer centers and their associated signs, and the parameters of non-Gaussian noise distributions. The equalizer centers can be calculated by estimates of the channel impulse response (CIR) with a linear adaptive algorithm [276]. To achieve an accurate estimate of the CIR, a long training sequence is needed, which causes lower data rates of trans-

mission. The parameters affecting the non-Gaussian distributions are the characteristic exponent and dispersion for a symmetric alpha-stable (SαS) noise. These must be determined. An efficient algorithm to estimate these parameters is based on fractional lower order statistics (FLOS) [313], but it suffers from a slow convergence rate. On the other hand, adaptive equalization schemes based on linear and nonlinear filtering techniques have been attractive alternatives. This is because estimation procedures for the CIR and the parameters associated with non-Gaussian distributions are not required.

It is known that the EKF provides an approximation to the optimal nonlinear estimation for Gaussian noise and it may have some limitations for non-Gaussian noise environments. The UKF, in contrast to the EKF, offers higher order approximations even for non-Gaussian distributions [14],[15]. Our assumption is that the recurrent neural equalizers combined with the UKF may lead to better performance over the EKF in non-Gaussian communication channels.

The performance evaluation of the recurrent neural equalizers with the UKF proposed in [314],[101] and [102] has been limited to only AWGN channels. In this Chapter, we evaluate the performance of the recurrent neural equalizer for non-Gaussian channels through Monte Carlo simulations. In the simulations, we use the SαS noise model to represent additive white non-Gaussian noise, which closely approximates the impulsive nature of noise in communication environments. To our best knowledge, it is the first application of neural equalization to non-Gaussian channels, albeit neural networks have been applied to signal detection [315],[316] and spread-spectrum receivers [280] in non-Gaussian noise environments.

We present a brief overview of the signal and impulsive noise model in Section 5.2. Section 5.3 starts describing a conventional DFE and its learning algorithm, called the normalized least mean p-norm which is suited for impulsive noise, and then gives a formulation of a DFE using a recurrent multilayer perceptron (RMLP). The UKF-based training algorithm for the RMLP equalizer is introduced and compared with distinction to the EKF in Section 5.4. In Section 5.5, we show simulation results and discuss performance comparison for stationary and nonstationary channels. Concluding remarks are given in Section 5.6.

5.2 System Model

5.2.1 Signal Model

Let $\{s[k]\}$ be the transmitted data sequence taken from a $\mathcal{M}$-ary alphabet $\mathcal{A}$ and consisting of independent and equiprobable binary symbols, i.e. $s_i[k] \in \{-1, +1\}$ at time k. The data symbols are transmitted through an ISI channel with a finite impulse response (FIR) system $\mathbf{h}$, which spans over N symbols. After appropriate filtering and sampling, the baseband received signal vector $\mathbf{r}[k]$ can be represented in matrix notation as

$$\mathbf{r}[k] = \mathbf{H}\mathbf{s}[k] + \mathbf{v}[k]. \tag{5.1}$$

In (5.1) $\mathbf{r}[k]$ is the received vector with the M most recent samples

$$\mathbf{r}[k] = [r[k]\, r[k-1] \cdots r[k-M+1]]^T,$$

$\mathbf{s}[k]$ is the transmitted symbol vector with the length $K = N + M - 1$

$$\mathbf{s}[k] = [s[k]\, s[k-1] \cdots s[k-K+1]]^T,$$

$\mathbf{H}$ is the $M \times K$ Toeplitz convolution matrix given by

$$\mathbf{H} = \begin{bmatrix} h_0 & h_1 & \cdots & h_{N-1} & 0 & 0 \\ 0 & h_0 & \cdots & h_{N-2} & h_{N-1} & 0 \\ \vdots & & & \ddots & & \vdots \\ 0 & 0 & \cdots & h_0 & \cdots & h_{N-1} \end{bmatrix},$$

and $\mathbf{v}[k]$ denotes the ambient noise vector

$$\mathbf{v}[k] = [v[k]\, v[k-1] \cdots v[k-M+1]]^T.$$

An equalizer must be used at the receiver to mitigate the effects of ISI and ambient noise from the received signal $r[k]$.

5.2.2 Non-Gaussian Noise Model

The impulsive noise can be modeled as an α-stable random process [272],[258], which closely approximates the simple Middleton's Class B noise [3],[277]. The symmetric α-stable (SαS) probability density function (pdf), $p_\alpha(t)$, is defined by its characteristic function

$$P(t) = e^{-\gamma|t|^\alpha} \tag{5.2}$$

where the parameters γ and α define a SαS distribution. The dispersion parameter γ ($\gamma > 0$) plays a similar role to the variance of the normal distribution. The characteristic exponent $\alpha \in (0, 2]$ governs the impulsiveness of the distribution; the pdf has heavier tails as α approaches 0, while it corresponds to the zero-mean normal distribution with variance 2γ when $\alpha = 2$. It can be the Cauchy distribution when $\alpha = 1$. SαS noise samples are generated from the following random variable [317]:

$$p_\alpha(t) = \gamma^{\frac{1}{\alpha}} \frac{\sin(\alpha t)}{(\cos t)^{\frac{1}{\alpha}}} \left(\frac{\cos((1-\alpha)t)}{A} \right)^{\frac{1-\alpha}{\alpha}}, \quad \alpha \neq 1 \tag{5.3}$$

where t is uniform in $(-\frac{\pi}{2}, \frac{\pi}{2})$ and A is the standard exponential.

The SαS noise has infinite variance which makes the use of the standard signal-to-noise ratio (SNR) meaningless. A new SNR measure, geometric signal-to-noise ratio (GSNR), should be used to indicate the strength between the information-bearing signal and the SαS noise [278]. The GSNR is defined as [258],[277],[278],[282]

$$\text{GSNR} = \frac{1}{2C_g} \left(\frac{\sqrt{E}}{S_0} \right)^2 \tag{5.4}$$

where $C_g \simeq 1.78$ is the exponential of the Euler's constant, and $S_0 = (C_g\gamma)^{\frac{1}{\alpha}}/C_g$ denotes the geometric power. The normalization constant $2C_g$ in (5.4) is used to ensure that the definition of the SNR corresponds to that of the SNR in the Gaussian case. The dispersion parameter γ can be estimated from the measurements by only the sample mean and variance [313],[272]. It is therefore regarded as a known parameter. Fig. 5.1 shows noise samples and the power spectrum of a SαS noise random variable, generated empirically at $\gamma = 1$. As expected, the noise power increases as α decreases. In general when α approaches zero, the noise power goes to infinity, which causes

significant performance degradation to the receiver. For example, we observe in Fig. 5.1 that the noise power is $P_{\alpha\alpha} \simeq 460$ when $\alpha = 1.5$, while it is $P_{\alpha\alpha} \simeq 2$ when $\alpha = 2.0$.

5.3 Adaptive Decision Feedback Equalization

The importance of adaptive equalization in modern digital communications has increased because of high data rates and severe channel distortions. Supervised (or training-data-aided) adaptive equalization is of particular interest to this chapter. In the following subsections we first describe a conventional decision feedback equalizer (DFE) structure and its training algorithm suitable for impulsive noise. Next, a formulation for a decision-feedback recurrent neural equalizer is given.

5.3.1 DFE and Normalized Least Mean p-Norm Algorithm

A communication system with a DFE is depicted in Fig. 5.2. The DFE is trained using a sequence perfectly known to the receiver. The channel with length N and a nonlinear distortion $g(\cdot)$ is modeled as

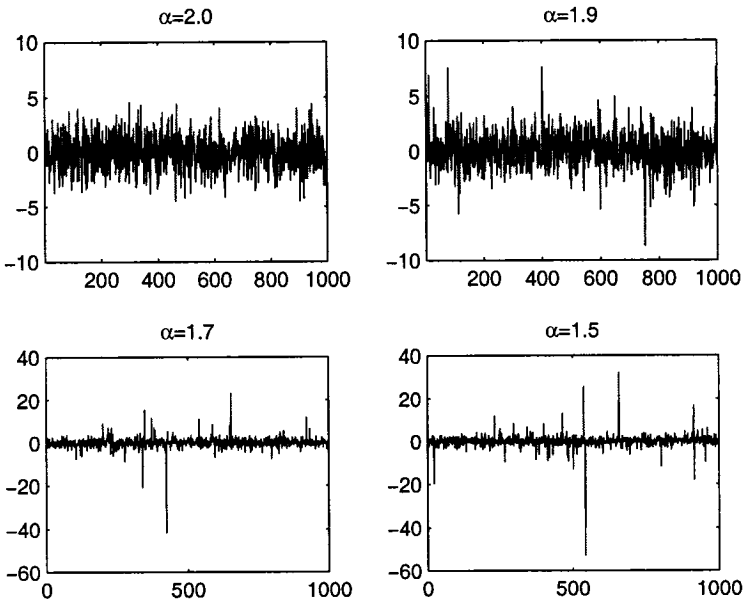
$$\begin{aligned} r_0[k] &= g(\hat{r}[k]) + v[k] \\ &= g\left(\sum_{i=0}^{N-1} h_i s[k-i]\right) + v[k]. \end{aligned} \tag{5.5}$$

To ensure that all receivers in practice have a finite dynamic input range, the nonlinear clipping function, or clipper, $q(\cdot)$ is added at the front end of the receiver [276],[277]

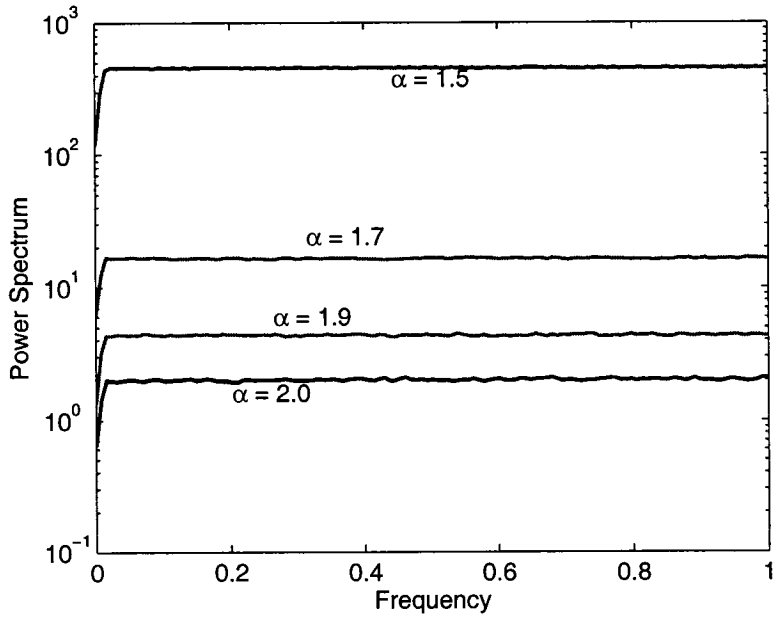
$$r[k] = q(r_0[k], G) = \begin{cases} r_0[k], & |r_0[k]| \leq G \\ G \cdot \text{sign}(r_0[k]), & \text{otherwise} \end{cases} \tag{5.6}$$

with G being the saturation value of the clipper.

The DFE is characterized by three parameters, m, n and d, known as the feedforward order, feedback order and decision delay, respectively. The inputs to the DFE consist of the forward inputs $\mathbf{r}[k] = [r[k], r[k-1], \cdots, r[k-m+1]]^T$ and feedback inputs $\mathbf{u}[k] = [u[k-1], \cdots, u[k-n]]^T$. The output of the DFE, $y[k]$, relies only upon the K symbols in (5.1) and it is passed through a memoryless decision device to determine the estimated symbol $\hat{s}[k-d]$. The delay, d, must be



(a) Noise samples



(b) Power spectrum

Fig. 5.1 Characteristics of a SαS noise process ($\gamma = 1$).

properly chosen within the range $d \in \{0, 1, \cdots, K-1\}$ to achieve satisfactory performance. The feedforward and the feedback order are determined as $m \geq d+1$ and $n = N + m - d - 2$, respectively [72].

A typical DFE consists of the m-tap feedforward filter and the n-tap feedback filter where each filter is constructed by a linear transversal filter. The DFE is trained by an adaptive filtering algorithm. Under the Gaussian assumption, random signals can be treated in a Hilbert space framework which requires the existence of the L_2 (or ℓ_2) norm to use various optimization criteria. Popular adaptive filtering techniques, such as least-mean-square (LMS) or recursive least squares (RLS) algorithms [12], are developed in a Hilbert space. Quadratic criteria such as the mean-squared error (MSE) cost function are used to optimize the tap weights of the DFE. However, a quadratic cost function is meaningless for impulsive noise generated by the SαS distribution, because the SαS noise process is of infinite variance for $\alpha < 2$. Instead, the SαS noise process can be treated in a Banach space when $1 \leq \alpha < 2$, where only the existence of the L_p norm for $p < \alpha$ is required [272].

The least mean p-norm (LMP), a direct generalization of the least mean square (LMS) algorithm, was proposed for α-stable environments [272]. The LMP algorithm was derived to minimize the p-norm cost function

$$\mathcal{J}[k] = \mathcal{E}[|e[k]|^p], \quad 0 < p < \alpha. \tag{5.7}$$

where $\mathcal{E}$ is the expectation operator and $e[k] = s[k-d] - y[k]$ is the error between the desired symbol $s[k-d]$ with decision delay d and the DFE output $y[k]$. The tap-weight vector $\mathbf{w}$ of the DFE can be updated with the LMP recursion

$$\hat{\mathbf{w}}[k+1] = \hat{\mathbf{w}}[k] + \eta \langle e[k] \rangle^{p-1} \mathbf{z}[k] \tag{5.8}$$

where $\eta > 0$ is the step-size parameter and $\langle e[k] \rangle^{p-1} = \text{sgn}(e[k])|e[k]|^{p-1}$. The input vector to the DFE, $\mathbf{z}[k]$, includes both the feedforward and feedback inputs as follows

$$\mathbf{z}[k] = [\mathbf{r}^T[k] \ \mathbf{u}^T[k]]^T. \tag{5.9}$$

To speed up the convergence of the DFE, the normalized LMP (NLMP) algorithm is desirable.

The NLMP algorithm [287] optimize the tap weights with the following updating rule

$$\hat{\mathbf{w}}[k+1] = \hat{\mathbf{w}}[k] + \eta \frac{\langle e[k] \rangle^{p-1}}{\|\mathbf{z}[k]\|_p^p + \lambda_r} \mathbf{z}[k] \tag{5.10}$$

where $\|\cdot\|_p$ denotes the L_p norm and $\lambda_r (\lambda_r > 0)$ is a regularization parameter used for avoiding large updates, which happen when the inputs are of occasionally small values. In the case of $p = 1$ in (5.10), the algorithm is called the normalized least mean absolute deviation (NLMAD). Furthermore, the NLMP algorithm reduces to the normalized LMS algorithm if $p = 2$ and $\lambda_r = 0$. The DFE trained with the NLMP algorithm is herein referred to as the DFE-NLMP.

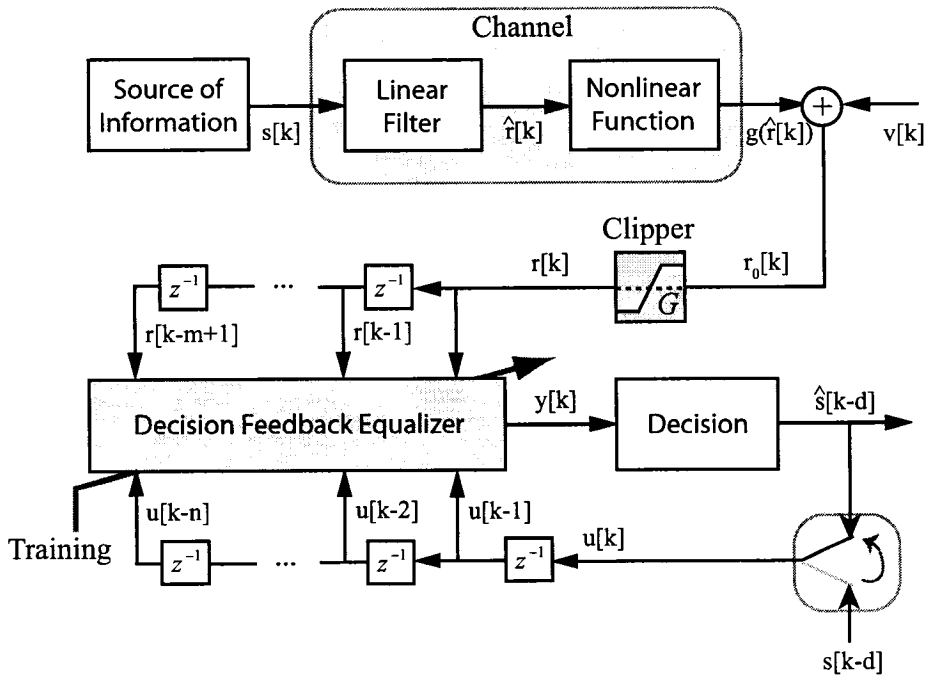


Fig. 5.2 Communication system with clipper and DFE.

5.3.2 Decision-Feedback RMLP Equalizer

Neural networks can be used as a DFE in Fig. 5.2. As a DFE, let us consider a RMLP model with one hidden layer as shown in Fig. 5.3. Let the vector $\mathbf{z}[k]$ denote the input vector, the vector

$\mathbf{x}[k+1]$ denote the output of the hidden layer, and $y[k+1]$ denote the output of the RMLP equalizer. The dynamic behavior of the RMLP equalizer may be described by a state-space model as follows:

$$\mathbf{x}[k+1] = \mathbf{f}(\mathbf{x}[k], \mathbf{z}[k]; \mathbf{w}_f) \tag{5.11}$$

$$y[k+1] = \mathbf{g}(y[k], \mathbf{x}[k+1]; \mathbf{w}_g) \tag{5.12}$$

where $\mathbf{f}(\cdot)$ and $\mathbf{g}(\cdot)$ denote the activation functions of the hidden layer and the output layer, respectively. The activation functions are implemented with parameter vectors $\mathbf{w}_f$ and $\mathbf{w}_g$. Throughout the chapter we use a hyperbolic tangent for the hidden layer and a pure linear function for the output layer. The dynamics of the RMLP given in equations (5.11) and (5.12) are recast by stochastic state-space equations, suitable for a Kalman filter formulation:

$$\mathbf{w}[k+1] = \mathbf{w}[k] + \boldsymbol{\omega}[k] \tag{5.13}$$

$$y_p[k+1] = \mathbf{c}(\mathbf{w}[k], \mathbf{x}[k+1], y[k], \mathbf{z}[k]) + \nu[k] \tag{5.14}$$

where the parameter vector of the RMLP equalizer, $\mathbf{w}[k]$, at time k is reformulated from two parameter vectors

$$\mathbf{w}[k] = \begin{bmatrix} \mathbf{w}_f[k] \\ \mathbf{w}_g[k] \end{bmatrix}. \tag{5.15}$$

The first equation (5.13), known as the process equation, specifies the state of the network when characterized as a stationary process corrupted by process noise $\boldsymbol{\omega}[k]$. The state of the system is given by the weight parameters of the RMLP equalizer, $\mathbf{w}[k]$. The second equation (5.14), the measurement equation, represents the desired output of the state-space recurrent model, $y_p[k+1]$, as a nonlinear function of the weight vector $\mathbf{w}[k]$, the input vector to the network $\mathbf{z}[k]$, the state feedback $\mathbf{x}[k]$, and the output feedback $y[k]$; this equation is augmented by a random measurement noise $\nu[k]$. The process noise $\boldsymbol{\omega}[k]$ is typically characterized as a zero-mean white noise with covariance given by $\mathcal{E}\{\boldsymbol{\omega}_i \boldsymbol{\omega}_j^T\} = \mathbf{Q}[k]$ where $\mathbf{Q}[k]$ is a diagonal matrix. Similarly, the measurement noise $\nu[k]$ is also typically characterized as a zero-mean white noise with covariance given by $\mathcal{E}\{\nu_i \nu_j^T\} = \mathbf{R}[k]$ where $\mathbf{R}[k]$ is a diagonal matrix.[1] The learning problem of the state-space RMLP model can be thought of as the estimation of the parameters $\mathbf{w}[k]$.

[1] $\mathbf{R}[k]$ in the RMLP equalizer can be treated as a constant because the equalizer's output is scalar.

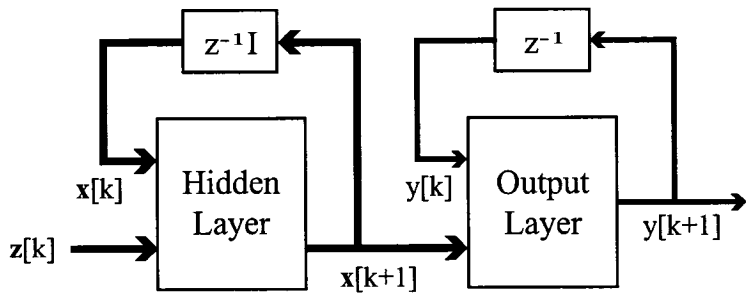


Fig. 5.3 Recurrent multilayer perceptron (RMLP).

5.4 Training the RMLP Equalizer with the UKF

5.4.1 The EKF's Limitations

EKF algorithms have been extensively used to train various types of neural networks [306],[309]. An in-depth description of the EKF methods for training recurrent neural equalizers can be found in [318],[102]. Obviously, the EKF has some limitations despite its popularity. A primary technique in the EKF formulation is a linearization to approximate the nonlinear system. In particular, the covariance matrix is computed at each time-step based on a Taylor series expansion to associate the network output to the weight vector. For the RMLP equalizer in (5.14) which is a nonlinear system, the EKF provides the *first-order* approximation of the system using a first-order truncation of the Taylor series expansion. Intuitively, the approximation may be valid if and only if all the higher order derivatives of the nonlinear functions are zero. Furthermore, the probabilistic spread of the states, captured by the covariance matrix which is often denoted as $\mathbf{P}$, is totally ignored during the first-order linearization process, albeit it plays a crucial role in the validity of EKF linearization [15]. The approximation can cause large errors in the true posterior mean and covariance of the transformed Gaussian (or non-Gaussian) random variable. These errors may lead to suboptimal performance and sometimes divergence of the filter [15],[22]. For the sake of comparison, the performance of the RMLP equalizer trained with the EKF, which is referred to as the RMLP-EKF, is evaluated in Section 5.5.

5.4.2 Unscented Transform

A novel procedure for dealing with estimation in nonlinear state space models has been proposed by Julier and Uhlmann [13]. This procedure belongs to the class of statistical linearization schemes in which densities are truncated instead of the models $\mathbf{f}(\cdot)$ and $\mathbf{g}(\cdot)$ given in (5.11) and (5.12). Higher order moments of the filter density are neglected, i.e. only mean and covariance are used. A sample set with the same mean and covariance is generated and propagated through the full state-space model, unlike the EKF which is approximated [319].

The foundation to the UKF is the unscented transformation (UT). The UT is a method for calculating the statistics of a random variable which undergoes a nonlinear transformation [13],[14]. Consider an L-by-1 random variable $\mathbf{x}$ that is nonlinearly transformed to yield a random variable $\mathbf{y}$ through a nonlinear function, $\mathbf{y} = f(\mathbf{x})$. In order to calculate the statistics of $\mathbf{y}$, a matrix $\boldsymbol{\chi}$ of $2L + 1$ sigma vectors χ_i is formed as follows:

$$\chi_i = \begin{cases} \bar{\mathbf{x}}, & i = 0 \\ \bar{\mathbf{x}} + (\sqrt{(L+\lambda)\mathbf{P}_{\mathbf{xx}}})_i, & i = 1, \ldots, L \\ \bar{\mathbf{x}} - (\sqrt{(L+\lambda)\mathbf{P}_{\mathbf{xx}}})_{i-L}, & i = L+1, \ldots, 2L \end{cases} \tag{5.16}$$

where $\bar{\mathbf{x}}$ and covariance $\mathbf{P}_{\mathbf{xx}}$ are the mean and covariance of $\mathbf{x}$, respectively. A scaling factor λ is chosen as [15]

$$\lambda = \theta^2(L + \kappa) - L. \tag{5.17}$$

The constant θ determines the spread of the sigma points around $\bar{\mathbf{x}}$; it is set to a small positive value, typically in the range $0.001 < \theta < 1$. The constant κ is a secondary scaling factor that is usually set to $3 - L$. The sigma points $\{\chi_i\}_{i=0}^{2L}$ are propagated through the nonlinear function

$$\mathcal{Y}_i = f(\chi_i), \qquad i = 0, \ldots, 2L. \tag{5.18}$$

This propagation produces a corresponding vector set that can be used to estimate the mean and covariance matrix of the nonlinear transformed vector $\mathbf{y}$. We can approximate the mean and covariance matrix of $\mathbf{y}$ using a weighted sample mean and covariance of the a posteriori sigma points

[15],

$$\bar{\mathbf{y}} = \sum_{i=0}^{2L} W_i^m \mathcal{Y}_i \tag{5.19}$$

$$\mathbf{P}_{\mathbf{yy}} = \sum_{i=0}^{2L} W_i^c (\mathcal{Y}_i - \bar{\mathbf{y}})(\mathcal{Y}_i - \bar{\mathbf{y}})^T \tag{5.20}$$

where the weighting factors are given by

$$\begin{aligned} W_0^m &= \frac{\lambda}{L+\lambda} \\ W_0^c &= \frac{\lambda}{L+\lambda} + (1 - \theta^2 + \beta) \\ W_i^m &= W_i^c = \frac{1}{2(L+\lambda)}, \quad i = 1, 2, \ldots, 2L. \end{aligned} \tag{5.21}$$

In the above equations, the superscripts m and c refer to the mean and covariance, respectively. β is used to take account of prior knowledge on the distribution of $\mathbf{x}$.

5.4.3 The UKF for RMLP Training

From the state-space model of the RMLP given in (5.13) and (5.14), the cost function to be minimized in the MSE sense is:

$$\mathcal{J}[k] = e[k]^T \mathbf{R}^{-1}[k] e[k] \tag{5.22}$$

where the error signal $e[k]$ is defined from (5.14)

$$e[k] = y_p[k] - \mathbf{c}(\mathbf{w}[k], \mathbf{x}[k], y[k-1], \mathbf{z}[k]). \tag{5.23}$$

We define $\mathbf{R}[k]$ and $\mathbf{Q}[k]$ as

$$\mathbf{R}[k] = \mu \mathbf{I} \tag{5.24}$$

$$\mathbf{Q}[k] = (\lambda_f^{-1} - 1)\mathbf{P}[k] \tag{5.25}$$

where μ can be set arbitrarily, and $\lambda_f \in (0, 1]$ is referred to as the forgetting factor in RLS algorithms. In (5.25), the covariance matrix $\mathbf{P}$ of the weight vector is initialized and updated in (5.27)

and (5.35), respectively.

The UKF effectively evaluates the Jacobian through its sigma-point propagation, without the need to perform any analytical derivative calculation. Specific equations for the RMLP equalizer using the UKF algorithm, called the RMLP-UKF, are shown below. The weight vector of the RMLP equalizer and its covariance matrix are initialized with

$$\hat{\mathbf{w}}[0] = \mathcal{E}\{\mathbf{w}\} \tag{5.26}$$

$$\mathbf{P}[0] = \mathcal{E}\{(\mathbf{w} - \hat{\mathbf{w}}[0])(\mathbf{w} - \hat{\mathbf{w}}[0])^T\}. \tag{5.27}$$

The sigma-point calculation is given by [15],[102]

$$\mathbf{\Gamma}[k] = (L + \lambda)(\mathbf{P}[k] + \mathbf{Q}[k]) \tag{5.28}$$

$$\mathcal{W}[k] = [\hat{\mathbf{w}}[k], \hat{\mathbf{w}}[k] + \sqrt{\mathbf{\Gamma}[k]}, \hat{\mathbf{w}}[k] - \sqrt{\mathbf{\Gamma}[k]}] \tag{5.29}$$

where the sigma-point matrix $\mathcal{W}$ has the $L \times (2L + 1)$ dimension. Each sigma-point vector is propagated to the system and the output is computed as

$$\mathcal{D}_i[k] = \mathbf{c}\left(\mathcal{W}_i[k],\, \mathbf{x}[k], y[k-1], \mathbf{z}[k]\right), \quad i = 0, 1, \ldots, 2L \tag{5.30}$$

where $\mathcal{W}_i$ is the ith column of the sigma-point matrix $\mathcal{W}$. The output associated with the estimated weight vector $\hat{\mathbf{w}}$ is also computed as

$$y[k] = \mathbf{c}(\hat{\mathbf{w}}[k],\, \mathbf{x}[k], y[k-1], \mathbf{z}[k]). \tag{5.31}$$

These outputs $\mathcal{D}[k]$ and $y[k]$ are used to compute covariance matrices in the following measurement-

update procedures. The measurement-update equations are

$$\mathbf{P}_{yy}[k] = \sum_{i=0}^{2L} W_i^c(\mathcal{D}_i[k] - y[k])(\mathcal{D}_i[k] - y[k])^T + \mathbf{R}[k] \tag{5.32}$$

$$\mathbf{P}_{\mathbf{w}y}[k] = \sum_{i=0}^{2L} W_i^c(\mathcal{W}_i[k] - \hat{\mathbf{w}}[k])(\mathcal{D}_i[k] - y[k])^T \tag{5.33}$$

$$\hat{\mathbf{w}}[k+1] = \hat{\mathbf{w}}[k] + \boldsymbol{\Upsilon}[k]e[k] \tag{5.34}$$

$$\mathbf{P}[k+1] = \mathbf{P}[k] - \boldsymbol{\Upsilon}[k]\mathbf{P}_{yy}[k]\boldsymbol{\Upsilon}^T[k] \tag{5.35}$$

where W_i^c is the weighting factor defined in (5.22) and $\boldsymbol{\Upsilon}[k] = \mathbf{P}_{\mathbf{w}y}[k]\mathbf{P}_{yy}^{-1}[k]$. The weight vector of the RMLP equalizer is updated sequentially with the above procedure. Due to the recursive structure of a Kalman filter framework, the probability of stopping in a local minimum of the cost function is greatly reduced compared to gradient descent methods.

5.4.4 The Pros and Cons of the UKF

Compared to the EKF, the pros of the UKF can be summarized here (for more details, refer to [14],[15],[320]). First, the UKF provides a higher order accuracy than the EKF. For instance, the UKF achieves third-order approximations for Gaussian distributions and at least second-order approximations for non-Gaussian ones, whereas the EKF achieves first-order approximations. Second, the UKF works with derivative-free optimization. Although this may not be valuable in training the RMLP equalizer because it is differentiable, it plays an important role in applications to systems which include non-differentiable components. Lastly, the UKF provides ease of computational implementation, due to the use of matrix algebra operations only.

Notwithstanding the pros of the UKF, a limitation in utilizing the UKF is the computational overhead. The UKF requires $\mathcal{O}(L^3)$ complexity due to the computation of a matrix square root at each time step, while the EKF needs $\mathcal{O}(L^2)$ complexity. A recursive square-root formulation of the UKF can achieve the same order of complexity as the EKF, yet it takes a slightly longer computation time. Another possible limitation could be the uncertainty in choosing the UT parameters (θ, β, and κ) which affect the performance and the convergence for parameter estimation.

5.5 Simulation Results

5.5.1 Non-Gaussian Channel

Before evaluating the performance of equalizers over non-Gaussian SαS noise environments, it is meaningful to address how the probability of error is changed over a SαS noise channel without ISI, where the receiver does not need to be equipped with an equalizer. The probability of error in a communication system is closely related to the *probability of exceeding* computed from the underlying noise distribution function. For the Gaussian ($\alpha = 2$) and the Cauchy ($\alpha = 1$) case, the probability of exceeding $P_{x>\tau}(\alpha, \gamma)$ [276], which means that the random variable x exceeds a given value τ, is

$$P_{x>\tau}(\alpha,\gamma) = \begin{cases} \frac{1}{2} - \frac{1}{2}\mathrm{erf}\left(\frac{\tau}{2\sqrt{\gamma}}\right), & \alpha = 2 \\ \frac{1}{2} - \frac{1}{\pi}\tan^{-1}\left(\frac{\tau}{\gamma}\right), & \alpha = 1 \end{cases} \tag{5.36}$$

where the error function erf(z) is defined as $\mathrm{erf}(z) = \frac{2}{\sqrt{\pi}}\int_0^z e^{-x^2}dx$. Except for the limiting cases, the Gaussian and the Cauchy, it is impossible to analytically calculate the probability of error. However, we can compute it experimentally.

To compute the probability of error via Monte Carlo simulations we consider the SαS noise channel and different values of α. Fig. 5.4 shows the probability of error (P_b) per bit, called the bit error rate (BER), as a function of the GSNR given in (5.4). In the simulations, 10^6 uncoded binary data symbols are transmitted and the MAP decision rule is applied at the receiver to detect the symbols. The clipper at the receiving end is set to $G = 4.0$. Compared with the Gaussian case, a performance degradation can be observed in the probability of error with the SαS noise. The performance degradation comes from the different shapes of tails between the Gaussian and the SαS distributions. The Gaussian distribution has exponential tails, while the SαS distribution has inverse power (i.e. algebraic) tails. We observe that the performance degradation is obvious even when $\alpha = 1.99$ where the distribution deviates only slightly from the Gaussian. As the value of α goes smaller, P_b becomes a straight line with its slope proportional to $-\alpha$. It is noteworthy to mention that the smaller the value of α, the heavier the tails are and the poorer the performance in terms of the probability of error.

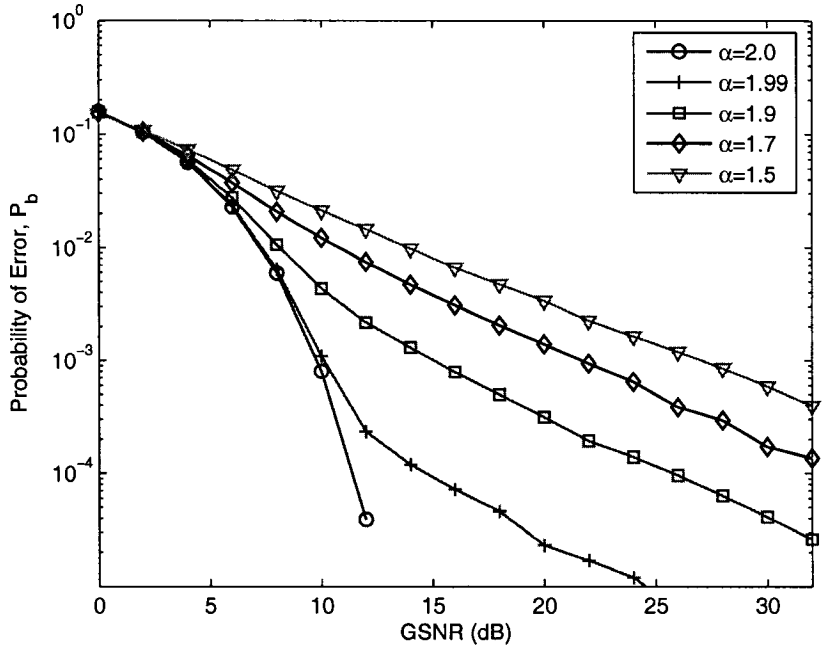


Fig. 5.4 Probability of error of a SαS channel without ISI (BPSK modulation used).

5.5.2 Stationary Dispersive Channels

Next, we consider stationary symmetric channels to evaluate the performance of DFEs. Four channel models are considered and are summarized in Table 5.1. Channel I is nonminimum-phase and its transfer function [72],[104] is

$$H_1(z) = 0.3482z^0 + 0.8704z^{-1} + 0.3482z^{-2} \tag{5.37}$$

with $\sum_i |h_i|^2 = 1$. Channel II is a nonlinear channel composed of the same CIR as the Channel I, followed by a memoryless nonlinearity $g(\hat{r}[k]) = \hat{r}[k] + 0.2\hat{r}^2[k]$ [104]. Such a nonlinear channel can be encountered in digital satellite communications [105] and digital storage systems [321]. Channel III is also a nonminimum-phase channel, with the transfer function [106],[44]

$$H_2(z) = 0.407z^0 + 0.815z^{-1} + 0.407z^{-2} \tag{5.38}$$

which has a deep spectral null at a high frequency and, therefore, results in severe ISI. In Channel IV a nonlinear distortion is applied as $g(\hat{r}[k]) = \tanh(\hat{r}[k])$ on top of Channel III where $\tanh(\cdot)$ takes into account saturation effects of transmission amplifiers.

Table 5.1 Stationary nonminimum-phase channels.

Channel	CIR Model	Distortion	Remarks
I	$H_1(z)$	None	Linear
II	$H_1(z)$	$g(\hat{r}) = \hat{r} + 0.2\hat{r}^2$	Nonlinear
III	$H_2(z)$	None	Linear
IV	$H_2(z)$	$g(\hat{r}) = \tanh(\hat{r})$	Nonlinear

The performance of three equalizers, the DFE-NLMP, the RMLP-EKF and the RMLP-UKF equalizers, is evaluated and compared with respect to BER and convergence rates. We fix the decision delay $d = 2$ to achieve satisfactory BER performance for all the channels considered. For the DFE-NLMP, the feedforward and feedback taps have $m = 5$ and $n = 4$, respectively. The parameters for the NLMP algorithm given in (5.10) are set to $p = 1.5$, $\eta = 0.2$ and $\lambda_r = 0.1$ by trial and error. For the RMLP equalizers, we used three forward inputs ($m = 3$) and two decision feedback inputs ($n = 2$). The structure of the RMLP equalizers consists of two hidden neurons and a single output neuron. Therefore, the RMLP equalizers have 20 trainable weights. For a set of the parameters needed for the EKF algorithm, refer to Chapter 3 or [102]. The parameters for the UKF algorithm are chosen empirically as $\theta = 0.1$ in (5.17), $\beta = 2$ in (5.22), $\mu = 0.5$ in (5.24) and $\lambda_f = 0.99$ in (5.25). The information symbols transmitted are generated by binary phase shift keying (BPSK) signaling throughout the simulations. The dynamic range of all the equalizers is fixed to $G = 4$.

Consider first the nonminimum-phase Channels I and II. In Fig. 5.5 we display the clusters of the received signals $\mathbf{r}[k]$ for 5000 symbols which are transmitted over Channel I and contaminated by different impulsive noise $\mathbf{v}[k]$ at GSNR = 24 dB. We observe that as α decreases, the clusters blur severely. We compare the convergence rate of the RMLP-UKF with the DFE-NLMP.[2] Fig. 5.6 shows convergence curves for Channel I at GSNR of 24 dB, ensemble-averaged over 100

[2]We do not show the convergence curve of the RMLP-EKF in Fig. 5.6 because the convergence rates between the RMLP-EKF and the RMLP-UKF are indistinguishable.

independent runs. The DFE-NLMP converges to the steady-state MSE value, which is around -14 dB, after 200 iterations (or training symbols), while the MSE value of the RMLP-UKF reaches around -22.5 dB at 200 iterations and is continuously reduced as iterations go by. The convergence trend of the equalizers for Channel II has very similar shapes as in Fig. 5.6. However, due to the nonlinear distortion, MSE levels of convergence for Channel II are slightly higher than for Channel I.

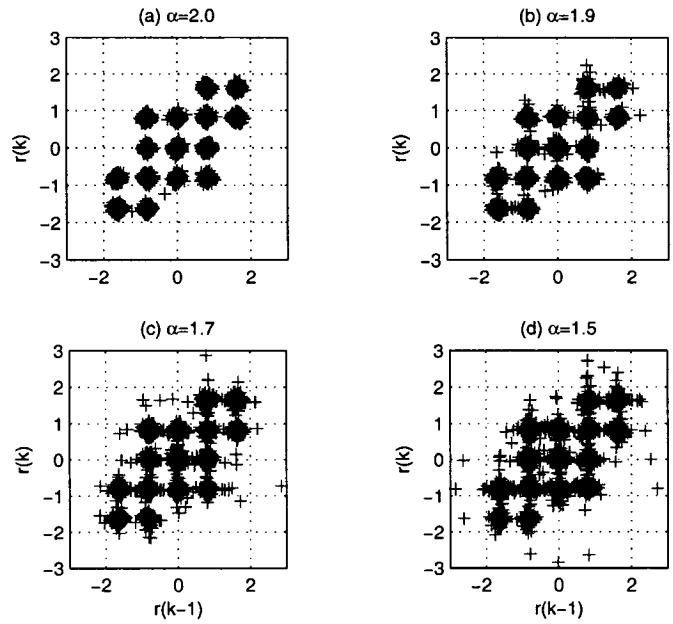
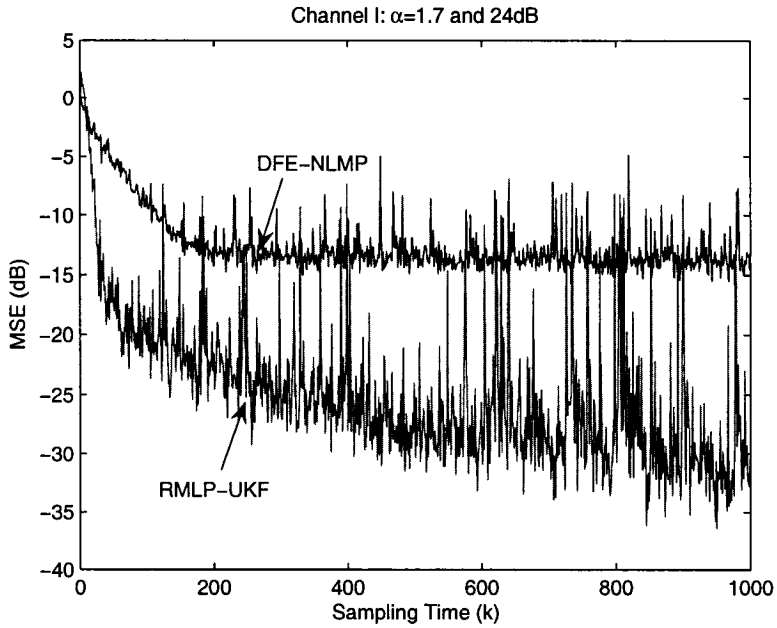
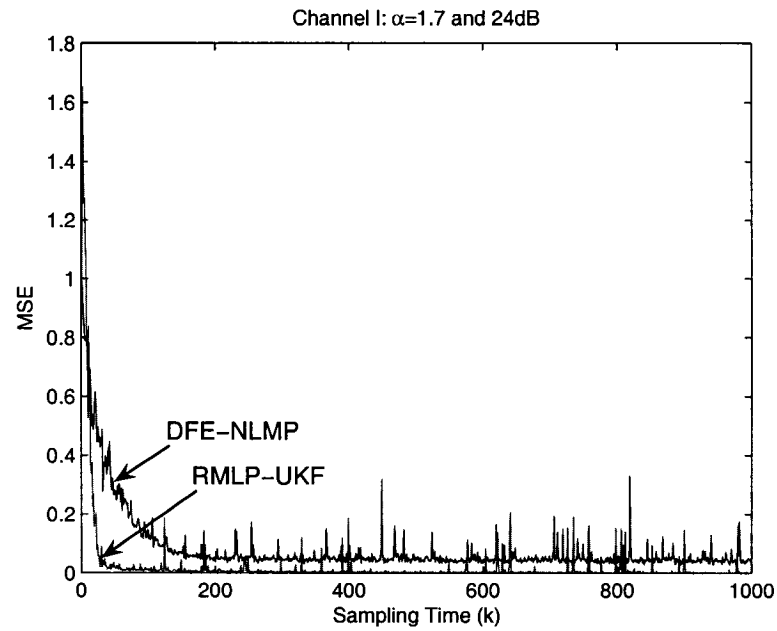


Fig. 5.5 Clusters of the received signals for Channel I at GSNR = 24 dB.

Fig. 5.7 shows BER convergence curves of the three equalizers for Channels I and II, where BER values are ensemble-averaged over 500 independent runs at GSNR = 24 dB and each run transmits 10^4 symbols, excluding different lengths of training symbols. For Channel I, the DFE-NLMP requires 150 training symbols to reach the steady-state BER of 1.57×10^{-3}. The RMLP-UKF, however, needs only 50 training symbols to reach the same steady-state BER. The RMLP-EKF requires 100 training symbols to reach the steady-state BER, 1.7×10^{-3}, which is higher than the BER achieved by the RMLP-UKF and the DFE-NLMP. The right side figure in Fig. 5.7 shows the BER convergence curves for Channel II. After 100 training symbols, the RMLP-UKF and the RMLP-EKF converge to the steady-state BERs of 1.68×10^{-3} and 1.75×10^{-3},



(a) y-axis: Log scale



(b) y-axis: Linear scale

Fig. 5.6 Convergence curves of the equalizers for Channel I.

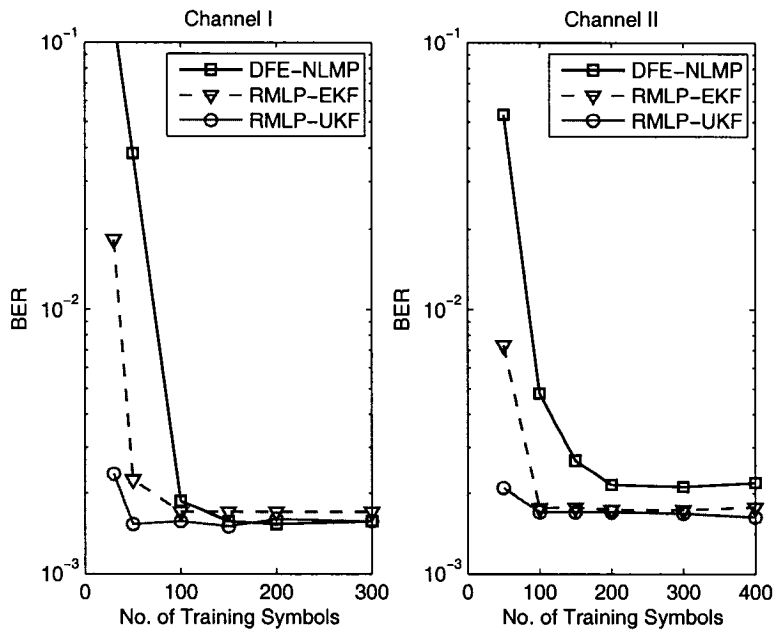


Fig. 5.7 BER vs. the number of training symbols for Channel I and II.

respectively. In contrast, after 200 training symbols the DFE-NLMP converges to the steady-state BER of 2.2×10^{-3} which is much higher than those obtained with the RMLP equalizers. It is noted that for Channel I the RMLP-UKF reaches the same steady-state BER as the DFE-NLMP using only 50 training symbols, while the RMLP-EKF does not.

For Channel I, in Fig. 5.8 we show the BER versus GSNR performance of the DFE-NLMP with different values of α for each case of 50, 100 and 150 training symbols. The BER performance of the two RMLP equalizers is depicted in Fig. 5.9, when only 50 training symbols are used. The results were ensemble-averaged over 100 independent trials, where each BER at a specific GSNR was obtained using 10^4 data symbols. As seen in the BER convergence in Fig. 5.8, the DFE-NLMP does not compensate the effects of ISI and non-Gaussian noise, when using 50 training symbols with any value of α considered. When using 150 training symbols, the transmitted symbols are recovered from the channel by the DFE-NLMP. We observe in Fig. 5.9 that the BER performance of the RMLP-UKF is comparable to that of the DFE-NLMP, yet it requires only a third of the training symbols required by the DFE-NLMP. However, it is noted that the RMLP-EKF produces a performance inferior to the RMLP-UKF for all the different values of α when 50 training symbols

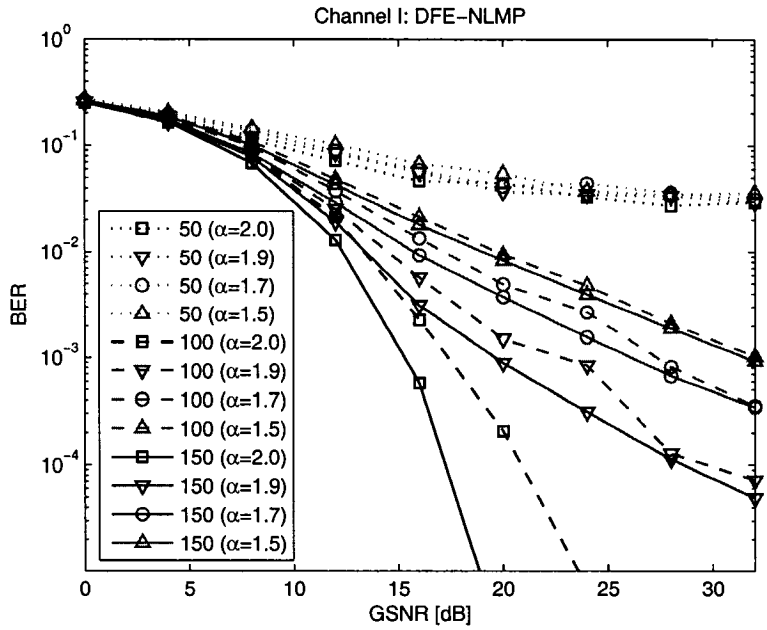


Fig. 5.8 BER performance of the DFE-NLMP for Channel I.

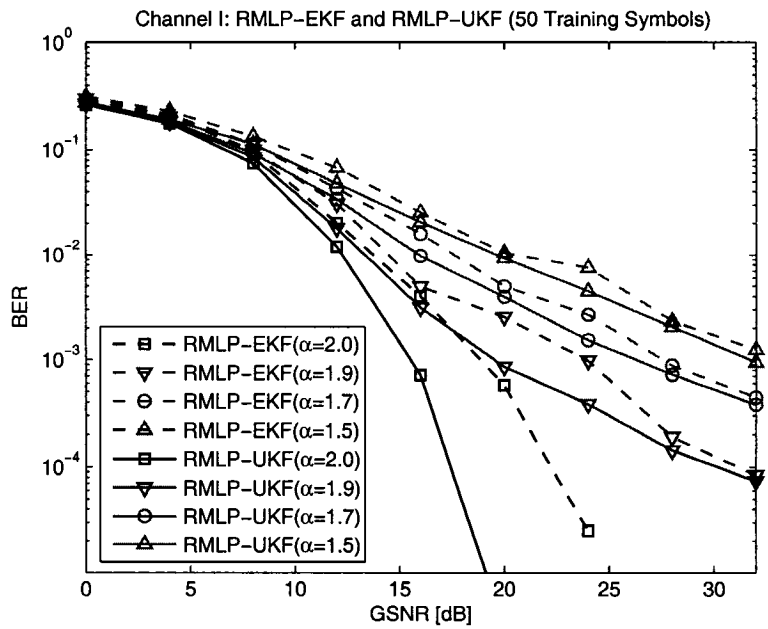


Fig. 5.9 BER performance of the RMLP equalizers for Channel I.

are used.

Fig. 5.10 shows the BER performance of the three equalizers for Channel II, where the RMLP equalizers use 100 training symbols, and the DFE-NLMP uses 200 training symbols. The RMLP-UKF shows a better BER performance than both the RMLP-UKF and the DFE-NLMP, albeit the training symbols are doubled for the DFE-NLMP. For instance, when GSNR = 20 dB, the RMLP-UKF achieves a performance gain of 1.2 dB over the RMLP-EKF and 2.5 dB over the DFE-NLMP at a BER of 10^{-3}. It can be noticed that when $\alpha = 2.0$ (Gaussian noise), the BER performance of the DFE-NLMP is far away from the performance of the two RMLP equalizers. This is why the NLMP algorithm was suited for non-Gaussian noise environments. Typical scatter plot patterns of equalizer outputs for Channel II are displayed in Fig. 5.11 at GSNR = 24 dB, where 200 symbols are used for training and only 4000 data symbols are drawn in the figures for ease of visualization. The scattering of noisy received signals fed to the equalizer is shown in Fig. 5.11(a). The equalized outputs by the DFE-NLMP, the RMLP-EKF and the RMLP-UKF are given in Fig. 5.11(b) - 5.11(d), respectively. The outputs of the RNLP-UKF are well deployed at around ± 1 which are the desired positions, whereas those of the DFE-NLMP are widely spread. This result shows clearly that the RMLP-UKF has a better classification ability than the other two equalizers.

Now, we evaluate the performance for Channels III and IV. For Channel III, Fig. 5.12 shows the convergence properties of all the equalizers in the same way as Fig. 5.6. After 300 iterations the DFE-NLMP converges to a steady-state MSE value of around -12.5 dB. We observe that the RMLP-UKF passes below the steady-state MSE level of the DFE-NLMP after only 50 iterations. Furthermore, the RMLP-DFE reaches the MSE value of around -28 dB with 1000 iteration. Not only is this MSE level far away from the steady-state MSE of the DFE-NLMP, but also it is not the steady-state MSE. Further iterations of the training still help lower the MSE level of the RMLP-UKF. BER convergence curves for Channels III and IV are depicted in Fig. 5.13 at $\alpha = 1.7$ and GSNR = 24 dB. For Channels III and IV, the RMLP equalizers converge to their steady-state BER with much less training symbols than for the DFE-NLMP, approximately one third. The results confirm that for nonlinear and non-Gaussian environments, the RMLP-UKF shows the best performance in terms of both the steady-state BER and the number of training symbols needed.

Plots of BER performance are depicted in Figs. 5.14 and 5.15 for Channels III and IV, respectively. It is interesting to note that the DFE-NLMP uses 200 and 300 training symbols for Channel

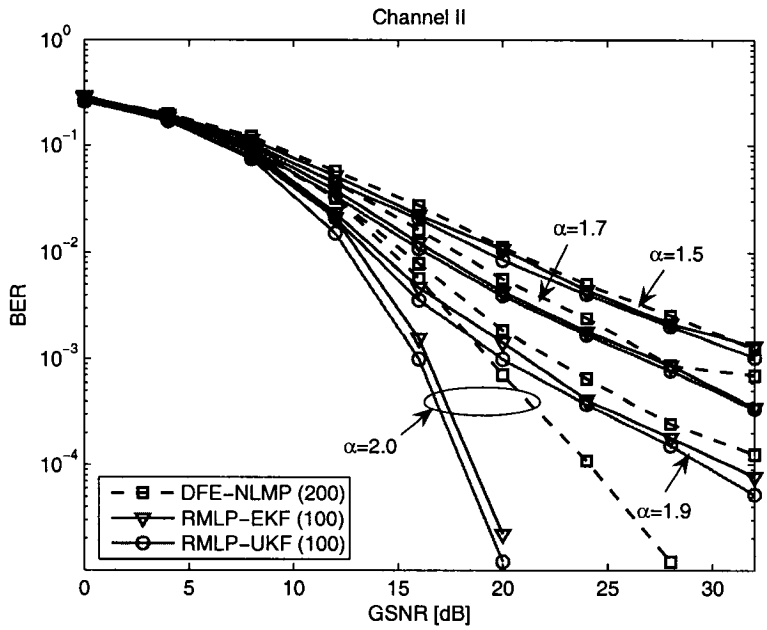


Fig. 5.10 BER performance of the equalizers for Channel II.

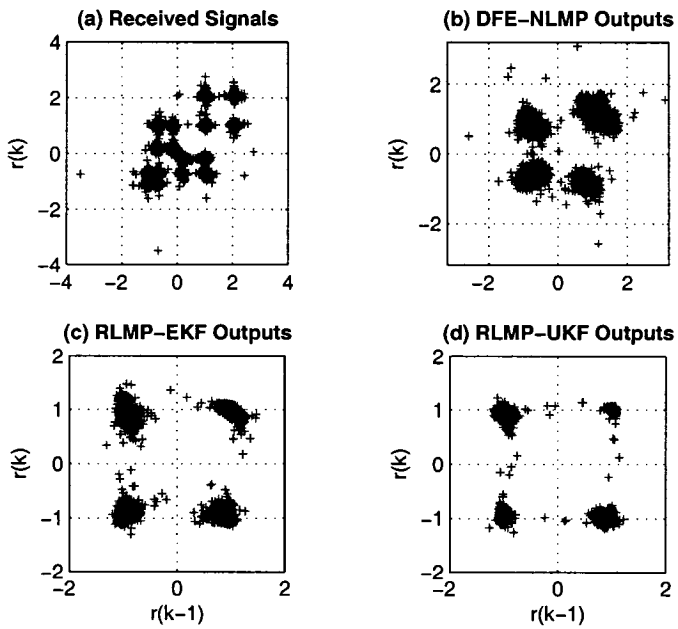


Fig. 5.11 Filtered outputs of the equalizers for Channel II.

III and IV, respectively, while the RMLP equalizers use only 100 training symbols in Figs. 5.14 and 5.15. In the case of $\alpha = 2$ which corresponds to the Gaussian noise, the RMLP-UKF is marginally better in BER performance than the RMLP-EKF, but it outperforms the DFE-NLMP. From a comparison of Fig. 5.14 with Fig. 5.15, we also observe that a performance loss of 4 dB exists between the linear and the nonlinear channel, due to the nonlinear distortion. In the case of impulsive non-Gaussian noise ($\alpha < 2.0$), as expected there is a definite performance degradation of the equalizers in comparison with the Gaussian noise case. We observe that at high GSNRs (greater than 20 dB), the performance gap between the RMLP-UKF and the other two equalizers is clear for both the linear and nonlinear channels. For instance, for Channel III in a noise environment with $\alpha = 1.7$, the RMLP-UKF achieves about 3.0 dB of performance gain over the RMLP-EKF and about 1.4 dB over the DFE-NLMP at a BER of 10^{-3}. For nonlinear Channel IV in a noise environment with $\alpha = 1.9$, the RMLP-UKF achieves about 1.5 dB of performance gain over the RMLP-EKF and about 2.0 dB over the DFE-NLMP at a BER of 10^{-3}.

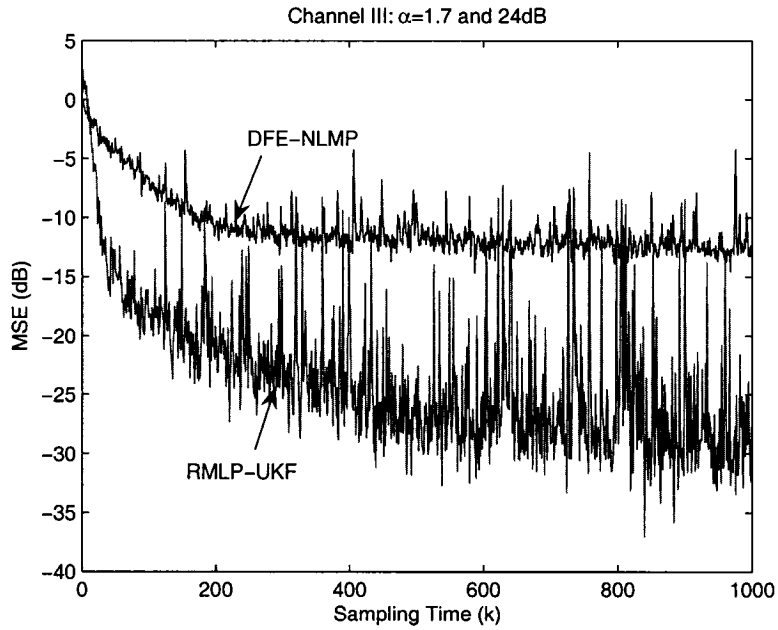


Fig. 5.12 Convergence curves of the equalizers for Channel III.

We have seen that the performance gain achieved by the RMLP-UKF in comparison with both the RMLP-EKF and the DFE-NLMP is visible in stationary ISI channels contaminated by im-

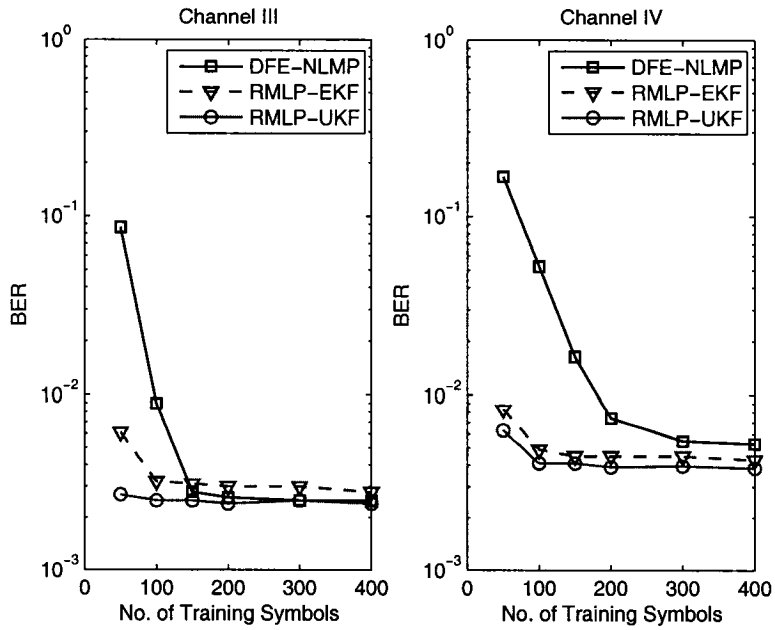


Fig. 5.13 BER vs. the number of training symbols for Channel III and IV.

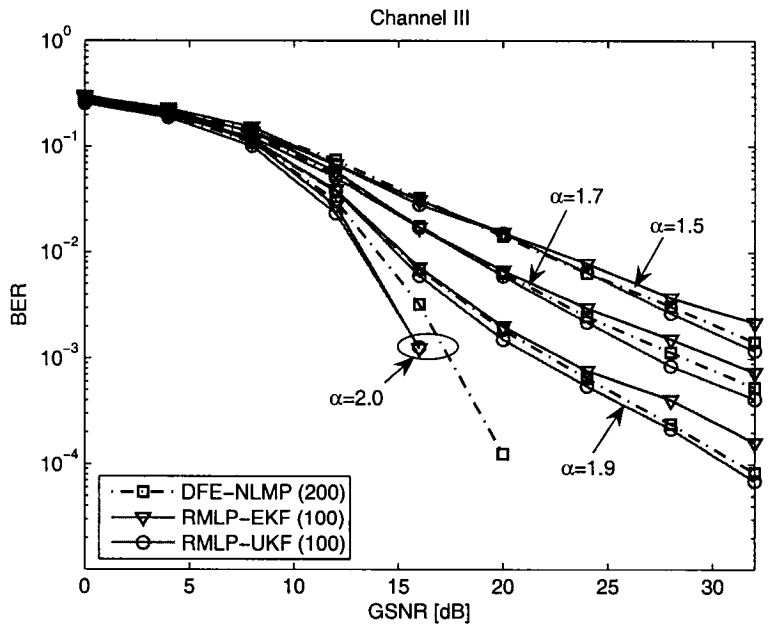


Fig. 5.14 BER performance for Channel III.

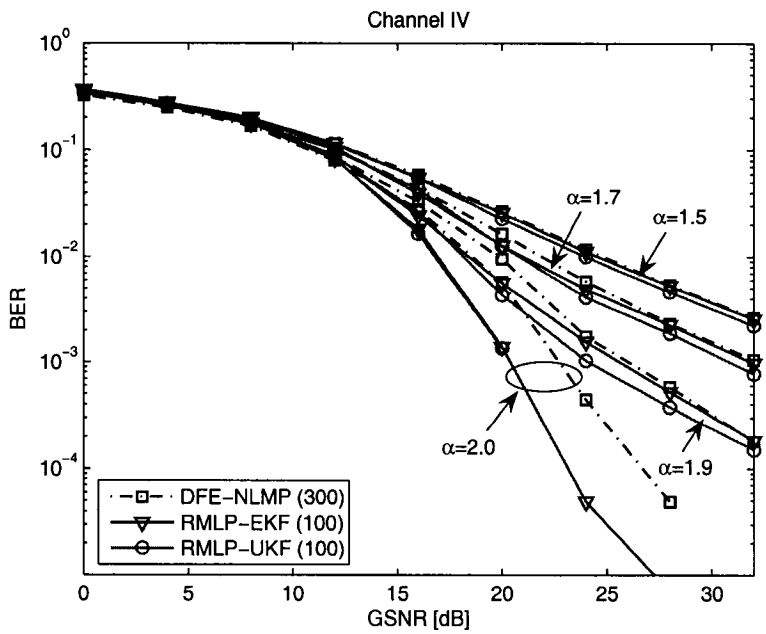


Fig. 5.15 BER performance for Channel IV.

pulsive non-Gaussian noise. The results support the superiority of the UKF to both the EKF and the NLMP in terms of filtering (or parameter estimation) performance in non-Gaussian environments. Based on the experiments, our observations for the stationary channels are summarized in the following:

Remark 1: For linear and nonlinear channels, the RMLP-UKF produces the best performance with the least training symbols, in terms of convergence and BER. It leads to high data rates of transmission.

Remarks 2: For linear channels, the DFE-NLMP shows a better BER performance with longer sequences of training symbols than the RMLP-EKF.

Remark 3: For the nonlinear channels, the BER performance of the RMLP-EKF is superior to that of the DFE-NLMP. This is because the RMLP architecture combined with the EKF is effective to compensate a nonlinear distortion, while the linear transversal filter constituting the DFE-NLMP is not suited to compensate the nonlinear distortion.

Remark 4: In non-Gaussian noise, some instability in the convergence curves for all the equalizers was observed due to the impulsiveness of the noise. Therefore, the BER performance is not

improved when further training symbols are provided.

Remark 5: Based on the superiority of the UKF to the EKF, we may conclude empirically that the UKF better captures statistical properties than the EKF in non-Gaussian noise environments.

Remark 6: The shape of BER curves in stationary dispersive channels also corresponds to the shape for the non-dispersive channel (i.e. no ISI channel). As α decreases, the BER curves become approximately a linear line with its slope inversely proportional to α.

5.5.3 Nonstationary Dispersive Channel

A linear time-varying channel model may be represented as

$$H_3(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} \tag{5.39}$$

where the CIR coefficients $h_i[k](i = 0, 1, 2)$ are varying with time k. These time-varying coefficients are generated by convolving white Gaussian noise and a Butterworth filter response [38],[102]. The channel output is passed through the nonlinear device used in Channel IV. Fig. 5.16 shows a prototype of a time-varying channel, drawn for a fading rate of 0.5 Hz.

To examine the tracking capability of the equalizers for the proposed time-varying channel, we consider both a training phase and a tracking phase at $\alpha = 1.7$ and GSNR = 24 dB. Each equalizer is trained by teacher-forcing symbols $s[k]$ in the training mode until $k = 1500$, and then it is switched to the decision-directed tracking mode at $k = 1501$. In the tracking mode, an estimate of the transmitted symbols, $\hat{s}[k]$, is fed back to the equalizer. For a GSNR of 24 dB, tracking performance in terms of the MSE in decibel is showed in Fig. 5.17, where the MSE value is ensemble-averaged over 100 trials and filtered with a window size of 10 samples to discriminate between the RMLP equalizers. The results verify that the convergence rate of the RMLP equalizers is much lower than that of the DFE-NLMP for both the training and the tracking modes. Furthermore, the RMLP-UKF presents a marginally faster recovery than the RMLP-EKF in the tracking mode.

BER versus GSNR performance for the equalizers is illustrated in Fig. 5.18, for the cases of $\alpha = 1.9$, 1.7 and 1.5. The BER is averaged over 500 independent trials, where 100 training symbols and 2000 data symbols are employed. It is noted that unlike the simulations for the

stationary channels, the weight vector of all the equalizers is still updated in order to track the fading characteristic of the channel. As seen in the result, the RMLP equalizers have a difficulty to overcome the channel at lower GSNR areas (GSNR $<$ 20 dB). However, at higher GSNR areas (GSNR $>$ 20 dB), the RMLP equalizers are able to drastically reduce the BER, and the RMLP-UKF outperforms the RMLP-EKF. On the other hand, we observe that the DFE-NLMP is not able to reduce the error floor even at higher GSNR, even if the number of training symbols increases. Evidently, the DFE-NLMP is also not fit to equalize a fading channel with the nonlinear distortion.

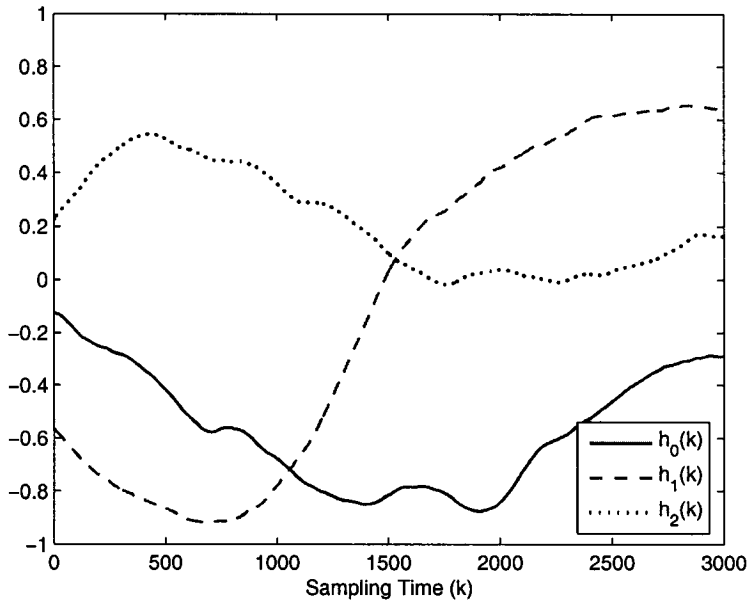


Fig. 5.16 Time-varying channel coefficients at a fading rate of 0.5 Hz.

5.6 Concluding Remarks

We presented an adaptive equalization technique using a RMLP network for symmetric α-stable noise environments, for which neural network-based equalizers had never been investigated. For the purpose of comparison, we introduced a DFE trained with the NLMP algorithm, developed for non-Gaussian noise. The performance of the RMLP equalizers was evaluated for both stationary and nonstationary ISI channels including a nonlinear distortion. In experimental results, the perfor-

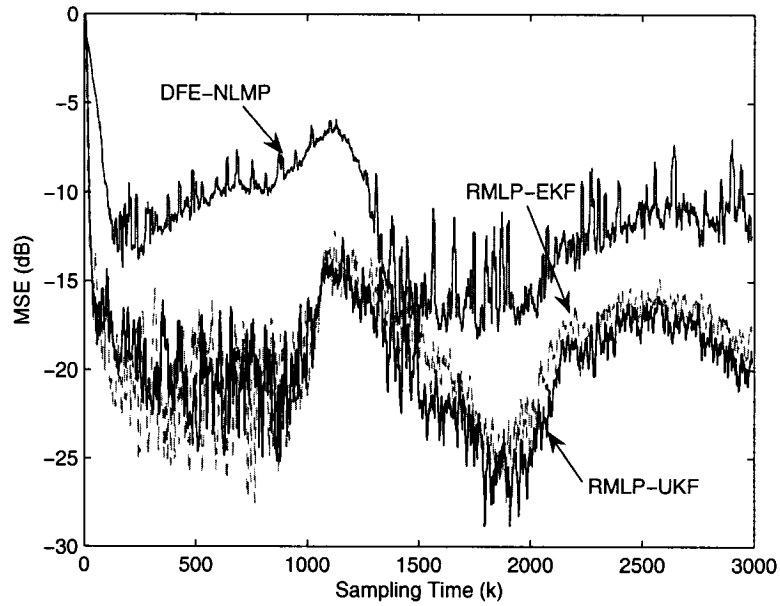


Fig. 5.17 Convergence properties of the equalizers for a GSNR of 24 dB.

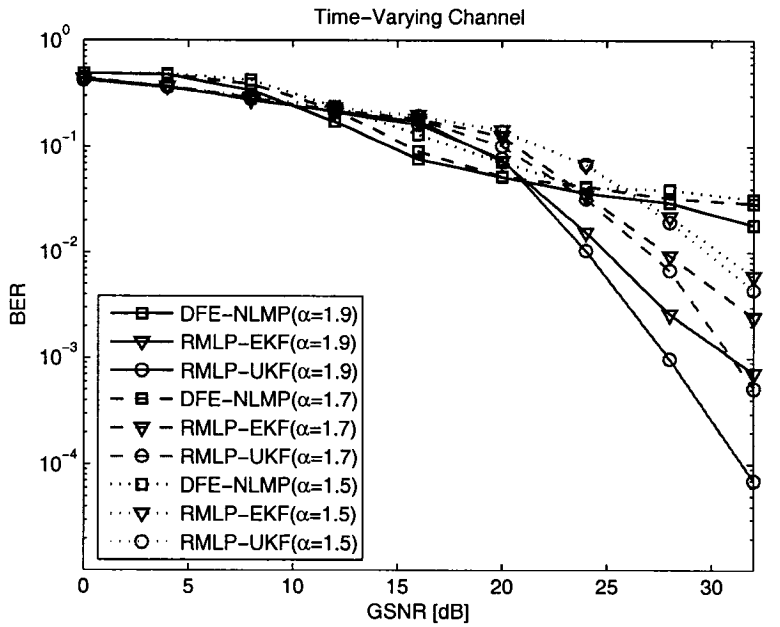


Fig. 5.18 BER performance of the equalizers for a time-varying channel.

mance of the RMLP-UKF showed its superiority to the RMLP-EKF in terms of BER performance. Furthermore, the RMLP-UKF needed significantly fewer training symbols to achieve better performance, compared to the RMLP-EKF and the DFE-NLMP. This result is important to achieve high-rate data transmission in digital communication systems. Furthermore, the RMLP-UKF is more robust to nonlinear distortions and non-Gaussian noise than the RMLP-EKF and the DFE-NLMP. The results support the fact that the UKF can capture higher order approximations than the EKF and perform better filtering task than the NLMP for nonlinear distortions and non-Gaussian random distributions. In addition, the RMLP-UKF is a suitable solution for compensating the effects of nonlinear distortion and non-Gaussian noise in time-varying channels.

Part III

Channel Estimation in MIMO Wireless Communications

Chapter 6

Iterative Channel Estimation For Turbo-BLAST

6.1 Introduction

Communication systems employing multiple-input multiple-output (MIMO) wireless links have emerged as a significant technical breakthrough in modern communications [8]. Information theory supports that MIMO fading channels offer not only the potential of high data rates but also the promise of high reliability. The results reported in a pioneering work [6] showed that enormous spectral efficiency can be achieved through the use of multiple antenna elements at both ends, i.e. transmitter and receiver. The major conclusion is that the capacity of a MIMO system far exceeds that of a single-transmit single-receive antenna system. Particularly, a MIMO wireless link has an asymptotic capacity that increases linearly with the number of transmit and receive antennas in a Rayleigh flat fading environment, assuming that the channel state information (CSI) is statistically independent and perfectly known to the receive antennas. Tremendous bandwidth efficiencies provided with MIMO systems were demonstrated with the Bell-Labs Layered Space-Time (BLAST) architecture using real-life channel data [5],[9]. In real scenarios, the perfect CSI is not available to the receiver. In practice the CSI is thus obtained through pilot symbol-aided or blind channel estimation. Therefore, channel estimation of MIMO links is a challenging topic since the performance is dependent on the quality of the channel estimates.

Recently, iterative (turbo) receivers for MIMO wireless links, which perform joint detection and decoding, have received great attention due to their performance gain. Turbo-BLAST

[322],[323], a promising iterative receiver which exploits the BLAST architecture, the random layered space-time coding, and the sub-optimum turbo-like receiver structure, demonstrated high spectral efficiency using real-life indoor channel measurements. However, the performance of turbo-BLAST with iterative channel estimation has not been investigated yet, although various types of iterative channel estimators are employed for other iterative receivers in the literature [213],[200],[197].

In a block Rayleigh fading environment, the channel information can be improved by iterative channel estimation using both dedicated pilot symbols and estimated code symbols based on soft information provided by the channel decoders. In this chapter, three iterative channel estimators for turbo-BLAST are investigated, which utilize least-squares (LS) and linear minimum mean-square error (MMSE) methods for snapshot estimation and a least mean-square (LMS) method for avoiding the matrix inversion of the snapshot approaches. In the literature [216], there exists an argument for the unreliability of the use of hard decision feedback, instead of soft decision feedback. Thus we evaluate the performance of the iterative channel estimators with both soft and hard decision feedback of the code symbols. Through Monte Carlo simulations, we show that the iterative channel estimators performed with hard decision feedback achieve better performance gain than those with soft decision feedback.

Adaptive filtering techniques may be a natural choice for iterative channel estimation, because of their computational efficiency compared to snapshot approaches [216],[200], such as LS and MMSE methods which need matrix inversions. At a cost of complexity, both recursive least-squares (RLS) [134],[135] and Kalman filter (KF) [185],[221] algorithms are extensively used for channel estimation in wireless communications, while the LMS algorithm is rarely used despite its computational efficiency and feasibility. In [134] and [135], RLS methods are used for iterative channel estimation of MIMO systems, but only the soft estimate of data symbols was evaluated. Although in [185] and [221], KF techniques are applied for tracking MIMO channels, they are not in an iterative estimation mode. The performance of adaptive filtering-based iterative channel estimators is compared in [202], but it is performed in a single-transmit single-receive (STSR) link. One should note that soft iterative channel estimation does not always guarantee the refinement of channel estimates, especially at a lower signal-to-noise ratio (SNR).

This chapter is organized as follows. In Section 6.2, basic characteristics of multiple-antenna

wireless systems are introduced, including benefits of MIMO systems, signal model, channel capacity, and impact of channel estimation error. In Section 6.3, the architecture and algorithms of the turbo-BLAST transceiver are given. In Section 6.4, snapshot-based iterative channel estimation is described and adaptive filtering-based estimation techniques is given in Section 6.5. The performance for turbo-BLAST with iterative channel estimation is evaluated in Section 6.6. Concluding remarks are given in Section 6.7.

6.2 Multiple-Antenna Wireless Communications

MIMO technology offers a substantial performance improvement. For instance, a 10-by-10 MIMO system can deliver 1 Gbps performance with only 20 MHz bandwidth and still support 80% of the reference range. The spectral efficiency achieved over a 20 MHz bandwidth by the 10-by-10 MIMO channel is 50 bps/Hz, which shows that high transmit power is not necessarily required to reach spectral efficiencies in excess of 10 bps/Hz. Furthermore, a MIMO system does not require additional transmit power or receive SNR to deliver such performance gains. We note that the disadvantage of using a MIMO system is the increased transceiver complexity.

The performance improvements resulting from the use of MIMO systems are due to array gain, diversity gain, spatial multiplexing gain, and interference reduction [324].

- *Array gain* can be made available through processing at the transmitter and the receiver and results in an increase in the average received SNR due to a coherent combining effect. However, the transmit/receive array gain requires *channel knowledge* in the transmitter and the receiver, respectively.

- *Diversity gain* can be achieved by diversity techniques. Spatial (or antenna) diversity of $M_T M_R$th order can be achieved if the $M_T M_R$ links composing the MIMO channel fade independently and the transmitted signal is suitably constructed. By using *space-time coding* techniques, spatial diversity gain in the absence of channel knowledge at the transmitter is possible using appropriately designed transmit signals. For more details, see [325].

- *Spatial multiplexing gain* can be realized by transmitting independent data signals from the individual antennas. Under conducive channel conditions such as rich scattering, MIMO

channels offer a linear (in $\min(M_T, M_R)$) increase in capacity without additional power or bandwidth expenditure.

- *Interference reduction* requires knowledge of the desired signal's channel. When multiple antennas are used, exact knowledge of the interferer's channel may not be necessary. Thus interference reduction (or avoidance) can also be implemented at the transmitter.

Generally, it is not possible to exploit all the leverages of MIMO technology simultaneously due to conflicting demands on the spatial degrees of freedom (or number of antennas). The degree to which these conflicts are resolved depends upon the signaling scheme and the transceiver design.

6.2.1 Signal Model

Consider a MIMO communication system with M_T transmit and M_R receive antennas, denoted as an (M_T, M_R) system over a frequency-nonselective Rayleigh fading channel. We assume that the channel is modeled as a flat-fading Rayleigh random process and the fading is to be uncorrelated across antennas. We focus on the baseband model of a system, which employs $\mathcal{M}$-ary phase shift keying (PSK) with zero ISI design. Let the $M_T \times 1$ transmitted signal vector in the kth symbol interval be $\mathbf{s}[k] = [s_1[k]\ s_2[k]\ \ldots\ s_{M_T}[k]]^T$. The received signal vector $\mathbf{r} \in \mathcal{C}^{M_R \times 1}$ on the receiver at symbol time k can be

$$\mathbf{r}[k] = \sqrt{\frac{E_s}{M_T}}\mathbf{H}[k]\mathbf{s}[k] + \mathbf{v}[k] \tag{6.1}$$

where $\mathbf{H} \in \mathcal{C}^{(M_R \times M_T)}$ denotes the channel matrix

$$\mathbf{H}[k] = \begin{bmatrix} h_{1,1}[k] & \ldots & h_{1,M_T}[k] \\ \vdots & \ddots & \vdots \\ h_{M_R,1}[k] & \ldots & h_{M_R,M_T}[k] \end{bmatrix}, \tag{6.2}$$

$\mathbf{s} \in \mathcal{C}^{M_T \times 1}$ is the symbol vector simultaneously transmitted by the M_T transmit antennas, and $\mathbf{v} \in \mathcal{C}^{M_R \times 1}$ denotes the complex additive white Gaussian noise vector with zero mean and covariance matrix $\sigma_v^2 \mathbf{I}_{M_R}$. The channel matrix remains constant during transmission of a frame of symbols (i.e., block fading) and changes randomly from block to block. Let us assume that channel estimates $\hat{\mathbf{H}}$ and symbol estimates $\hat{\mathbf{s}}$ have been obtained. At the beginning of the transmission,

the channel estimates could be obtained by using training symbols. Throughout the chapter we assume that transmission and receiving are perfectly synchronized with symbol timing.

6.2.2 Capacity of MIMO Channels

In what follows, we assume that the channel matrix $\mathbf{H}$ is perfectly known to the receiver [1]. Assuming the idealized Gaussian model, the capacity of a deterministic MIMO channel is defined by [4],[5]

$$C = \max_{\mathbf{R}_{ss}} \{I\}, \text{ b/s/Hz} \tag{6.3}$$

where the mutual information I for the transmitted signals s having covariance matrix $\mathbf{R}_{ss}$ is given by

$$I = \log_2 \det \left(\mathrm{I}_{M_R} + \frac{\rho}{M_T} \mathbf{H} \mathbf{R}_{ss} \mathbf{H}^H \right), \text{ b/s/Hz} \tag{6.4}$$

The maximization in (6.3) is performed over all possible covariance matrices $\mathbf{R}_{ss} = \mathcal{E}\{\mathrm{ss}^H\}$ of the transmitted signal vector s, satisfying $\mathrm{Tr}(\mathbf{R}_{ss}) = M_T$, and $\rho = E_s/N_0$ is the average SNR at each receiver input. In the absence of CSI at the transmitter, it is reasonable to choose s to be spatially white, i.e., $\mathbf{R}_{ss} = \mathrm{I}_{M_T}$. Thus, the *ergodic* capacity of the MIMO channel at time k is defined by the logarithm-determinant (log-det) formula, subject to a constant transmit power [4],[5]

$$C = \mathcal{E} \left[\log_2 \left\{ \det \left(\mathrm{I}_{M_R} + \frac{\rho}{M_T} \mathbf{H} \mathbf{H}^H \right) \right\} \right], \text{ b/s/Hz} \tag{6.5}$$

Due to the nonstationary characteristic of the MIMO channel, the ergodic capacity C may vary with $\mathbf{H}$. Note that if $M_T = M_R = M$, then the ergodic capacity of a flat-fading MIMO wireless channel grows linearly with M as M approaches infinity [6].

For a frequency-selective fading MIMO channel, the channel capacity can be obtained by dividing the frequency band of interest into L subchannels, each having bandwidth B/L Hz. If L is sufficiently large, each subchannel can be frequency-flat fading. Assuming that the transmit power is assigned uniformly across space (transmit antennas) and frequency, the ergodic capacity of the

[1] In real scenarios, channel knowledge at the receiver can be attained via training and tracking.

frequency-selective fading MIMO channel is given by [7]

$$C_{FS} = \mathcal{E}\left[\frac{1}{L}\sum_{i=1}^{L}\log_2\left\{\det\left(\mathrm{I}_{M_R} + \frac{E_{s,i}}{M_T N_0}\mathbf{H}_i\mathbf{H}_i^H\right)\right\}\right], \text{ b/s/Hz} \tag{6.6}$$

where $E_{s,i}$ is the energy allocated to the ith subchannel.

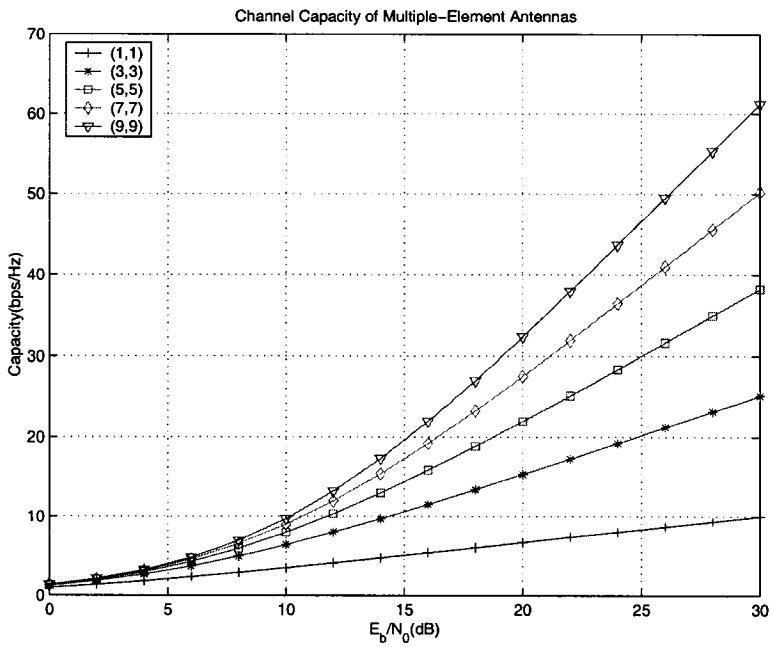


Fig. 6.1 The ergodic capacity of various MIMO link configurations.

6.2.3 Impact of Channel Estimation Errors

Although tremendous capacity gains have been achieved for MIMO channels under ideal assumptions, more realistic assumptions can significantly impact the potential capacity gains of MIMO techniques [326]. It is well understood that channel variability is the predominant cause of performance degradation in a realistic channel. In real world, the capacity predictions may rely on the accuracy of CSI, which can be estimated (or tracked) at both the receiver and the transmitter over time-varying circumstances. Since perfect CSI is difficult to obtain in MIMO systems due to an increased number of channel parameters to estimate at the receiver and to be fed back to the transmitter, the capacity gain is reduced.

Médard [327] investigated how imperfect channel knowledge affects on the channel capacity, where the author derives a lower bound of mutual information with channel estimation error for single-transmit single-receive (STSR) channels. For MIMO links, the channel capacity with imperfect CSI at the receiver can be lower-bounded by that of the known CSI receiver [207],[328],[329]. Based on the work done in [327], Yoo and Goldsmith derived lower and upper bound of channel capacity with channel estimation error [330],[331],[332]. On the other hand, constant channel estimation errors yield error floors at high signal to noise ratios. At low SNR, errors are dominated by noise, whereas at high SNR errors are dominated by channel estimation errors. For the capacity at low SNR, the reader can refer to [333].

The channel matrix $\mathbf{H}$ describes the effect of fading between two ends of the wireless link. The estimated channel matrix $\hat{\mathbf{H}}$ is related to the channel matrix $\mathbf{H}$ through

$$\mathbf{H} = \hat{\mathbf{H}} + \tilde{\mathbf{H}} \tag{6.7}$$

where $\tilde{\mathbf{H}}$ is the channel estimation error matrix. We assume that $\mathbf{H}$ and $\tilde{\mathbf{H}}$ are uncorrelated and that the elements of $\tilde{\mathbf{H}}$ are i.i.d. complex Gaussian random variables with zero mean and variance $\sigma_e^2/2$ in each dimension [331]. The variance σ_e^2 indicates the quality of channel estimation and is assumed to be known at the receiver. The performance of the considered spatially-multiplexed MIMO systems may be evaluated with the ergodic capacity limit. The ergodic capacity of a MIMO channel with estimation errors is given by [332]

$$C = \mathcal{E}\left[\log_2 \det\left(\mathbf{I} + \frac{P}{2M_T\sigma_v^2}\hat{\mathbf{H}}\hat{\mathbf{H}}^H \frac{2\sigma_v^2}{2\sigma_v^2 + \sigma_e^2 P}\right)\right] \tag{6.8}$$

where P is the average transmit power and σ_v^2 is the noise variance. Due to the ergodicity of the channel, the statistical expectation can by approximated by an ensemble-average over many realization of $\hat{\mathbf{H}}$. In Fig. 6.2, we illustrate the impact of channel estimation errors on the capacity for (2,2) and (4,4) MIMO configurations. The capacity is ensemble-averaged over 10^5 channel realizations in frequency-flat Rayleigh fading channels where each channel coefficient is assumed to be uncorrelated. The capacity curve for channel estimation errors of 10 percent ($\sigma_e^2 = 0.1$) and 20 percent ($\sigma_e^2 = 0.2$) at the receiver saturates in the high SNR region, while the capacity for perfect CSI ($\sigma_e^2 = 0$) continues to increase.

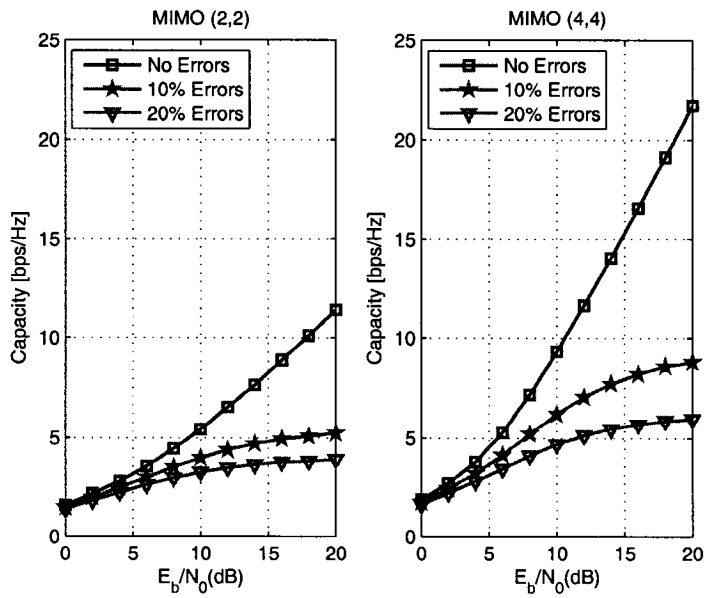


Fig. 6.2 Comparison of the ergodic capacity for frequency-nonselective block fading.

6.3 Turbo-BLAST: A MIMO Transceiver

A main feature of turbo-BLAST proposed in [322][323] is random layered space-time (RLST) codes with a space-time interleaver. The RLST codes were devised to achieve two objectives: 1) an iterative receiver for jointly decoding the simultaneously transmitted substreams with low complexity and 2) the realization of MIMO channel capacity in a computationally tractable manner. We briefly describe the architecture and algorithms of turbo-BLAST in this section.

6.3.1 Transmitter Structure

A high-level description of the turbo-BLAST transmitter employing the random layered space-time (RLST) codes is depicted in Fig. 6.3(a). User information stream $\mathbf{b}$ is demultiplexed into M_T substreams $\{\mathbf{b}_m\}_{m=1}^{M_T}$ of an equal data rate. Each data substream is independently block-encoded, where the block encoders use the same linear block codes with code generator $G(D)$. The encoded substreams are represented as

$$\mathbf{C} = [\mathbf{b}_1 G \ \mathbf{b}_2 G \ \ldots \ \mathbf{b}_{M_T} G] = [\mathbf{c}_1 \, \mathbf{c}_2 \ \ldots \ \mathbf{c}_{M_T}]^T \tag{6.9}$$

where $\{\mathbf{b}_m\}_{m=1}^{M_T}$ are the J-dimensional information sequences, $\{\mathbf{c}_m\}_{m=1}^{M_T}$ are the K-dimensional code sequences, and G is a $(J \times K)$ binary code generator. The encoded substreams are bit-interleaved using a pre-designed space-time random interleaver, Π, and the interleaved substreams are denoted by

$$\mathbf{A} = \Pi(\mathbf{C}) \tag{6.10}$$

The space-time interleaver comprises two stages [322]: 1) A space interleaver based on the diagonal layering of independently coded substreams and 2) M_T different, independent time interleavers of length K. The space-time interleaved substreams are independently mapped into the symbol matrix $\mathbf{S}$,

$$\mathbf{S} = f(\mathbf{A}) \tag{6.11}$$

where $f(\cdot)$ depends on the modulation scheme used. The modulated substreams are transmitted using the M_T transmit antennas.

6.3.2 Iterative Receiver Structure

The RLST codes designed by concatenating random block codes and the space-time interleaver can be viewed as serially concatenated codes separated by an interleaver: an *inner code* composed of a $M_R \times M_T$ channel matrix and an *outer code* composed of M_T parallel channel codes. At the receiver, the inner and the outer codes are detected by the inner detector and decoded by the outer decoders, respectively, in a turbo-like fashion. Fig. 6.3(b) illustrates the iterative detection and decoding (IDD) receiver of turbo-BLAST with a channel estimator. In the IDD receiver, the inner detector receives the soft *intrinsic* inputs from the outer decoders, and produces the soft *extrinsic* outputs. The outer decoders do the same work. Thus the soft information is exchanged between the inner detector and the outer decoders during the iteration.

Inner SISO Detector

A maximum *a posteriori* probabilities (MAP) algorithm can be used as an optimal detector. However, due to the computational complexity of the MAP detector, a multistream detector based on minimum mean-square error (MMSE) and parallel soft-interference cancellation (PSIC) is an alternative choice. This MMSE soft-input soft-output (SISO) detector optimizes jointly the weights

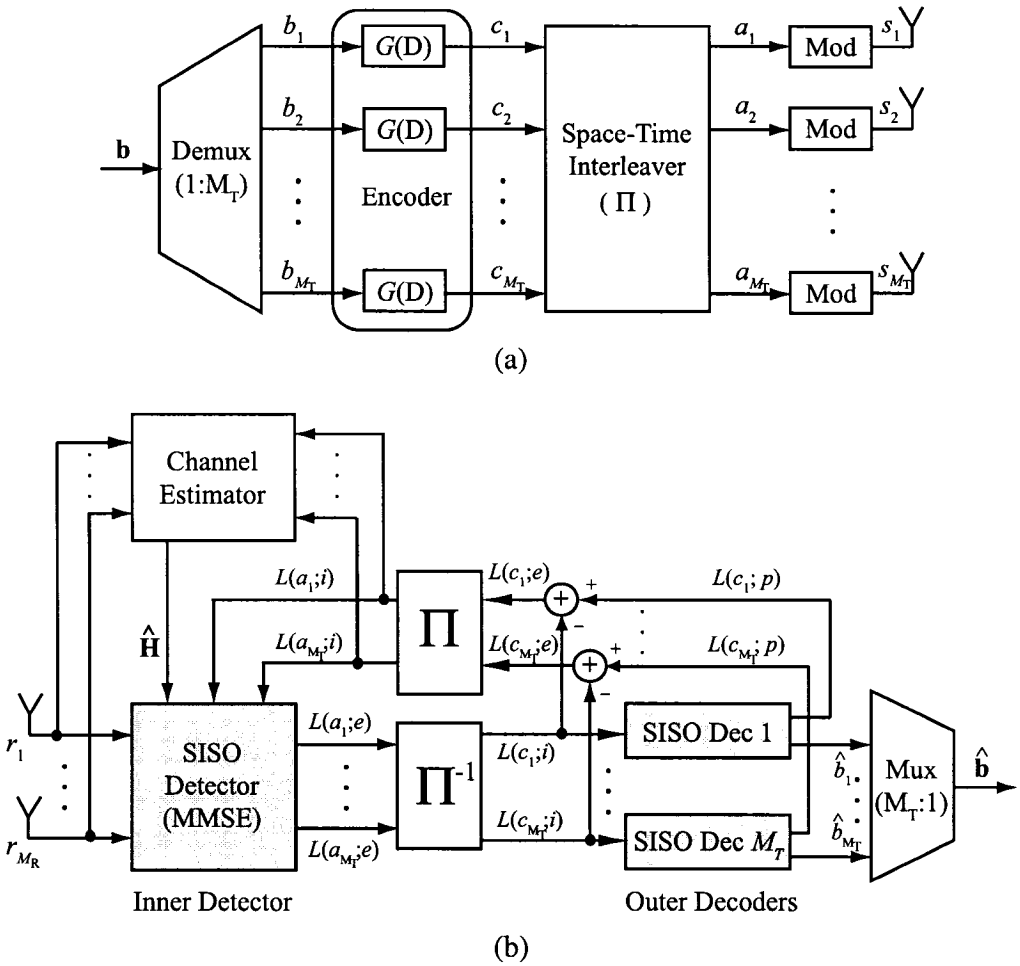


Fig. 6.3 Turbo-BLAST: (a) Spatial-multiplexing transmitter and (b) Iterative receiver.

of the linear detector and the co-antenna interference (CAI) estimate.

Assuming $s_m[k]$ to be the desired signal, equation (6.1) can be recast as

$$\mathbf{r}[k] = \mathbf{h}_m s_m[k] + \mathbf{H}_m \mathbf{s}_m[k] + \mathbf{v}[k],\ m = 1, 2, \ldots, M_T \tag{6.12}$$

where

$$\mathbf{H}_m = [\mathbf{h}_1 \ldots \mathbf{h}_{m-1}\, \mathbf{h}_{m+1} \ldots \mathbf{h}_{M_T}] \in \mathcal{C}^{M_R \times (M_T - 1)}$$

with its mth column removed and

$$\mathbf{s}_m[k] = [s_1[k] \ldots\ s_{m-1}[k]\, s_{m+1}[k] \ldots s_{M_T}[k]]^T \in \mathcal{C}^{(M_T-1)\times 1}$$

with its mth entry removed. The linear beamformer output using a linear filter $\mathbf{w}_m$ for the mth substream at sampling interval k is

$$y_m = \mathbf{w}_m^H \mathbf{h}_m s_m + \mathbf{w}_m^H \mathbf{H}_m \mathbf{s}_m + \mathbf{w}_m^H \mathbf{v}_m \tag{6.13}$$

where the sampling index k is omitted for brevity. The CAI can be removed from the linear beamformer output y_m, and the improved estimate x_m of a transmitted symbol s_m using PSIC based on the channel information is given by [322]

$$x_m = \mathbf{w}_m^H \mathbf{r} - \mathbf{w}_m^H \mathbf{H}_m \mathcal{E}\{\mathbf{s}_m\} \tag{6.14}$$

$$\cong (\mathbf{h}_m^H \mathbf{h}_m + \sigma_v^2)^{-1} \mathbf{h}_m^H (\mathbf{r} - \mathbf{H}_m \mathcal{E}\{\mathbf{s}_m\}) \tag{6.15}$$

where $\mathcal{E}\{\mathbf{s}_m\}$ are the statistical expectations of the interfering substreams, i.e. the vector of estimated symbols. The expectations of the transmitted data are computed using the soft outputs of the symbols, which are log-likelihood ratios, provided by the outer SISO decoders.

Outer SISO Decoders

The outer SISO decoders for convolutionally encoded symbols perform symbol-by-symbol MAP decoding realized by the BCJR (Bahl-Cocke-Jelinek-Raviv) algorithm [122]. For the computational simplicity, a log-MAP decoder is used to implement the BCJR algorithm. Instead of

probabilities, it computes log-likelihood ratios (LLRs). The *a posteriori* LLR of a coded bit c_m, $m = 1, 2, \ldots, M_T$, conditioned on the received symbol vector $\mathbf{r}$, is defined as

$$\begin{aligned} \mathcal{L}(c_m; p) &= \ln \frac{P[c_m = +1|\mathbf{r}]}{P[c_m = -1|\mathbf{r}]} \\ &= \ln \frac{P[c_m = +1]}{P[c_m = -1]} + \ln \frac{P[\mathbf{r}|c_m = +1]}{P[\mathbf{r}|c_m = -1]} \\ &= \mathcal{L}(c_m; i) + \mathcal{L}(c_m; e) \end{aligned} \tag{6.16}$$

where Bayes' rule is applied, $\mathcal{L}(c_m; i)$ and $\mathcal{L}(c_m; e)$ denote *a priori* (or *intrinsic*) information and *extrinsic* information of the coded bit c_m, respectively.

The extrinsic information of the inner SISO detector, $\mathcal{L}(\mathbf{A}; e)$, is deinterleaved to compensate for the pseudo-random interleaving used in the transmitter before feeding to the outer SISO decoders, yielding

$$\mathcal{L}(\mathbf{C}; i) = \Pi^{-1}(\mathcal{L}(\mathbf{A}; e)) \tag{6.17}$$

In turn, the M_T parallel SISO decoders process the soft input $\mathcal{L}(\mathbf{c}_m; e)$ and compute the refined estimates of soft information on both code bits $\mathbf{c}_m$ and information bits $\mathbf{b}_m$ based on the trellis structure of the channel code constraint. The *a posteriori* information for the code and information bits is computed as

$$\mathcal{L}(c_m; p) = \ln \frac{P\{c_m = +1|\mathcal{L}(\mathbf{C}; i), \mathcal{L}(\mathbf{A}; i), \text{decoding}\}}{P\{c_m = -1|\mathcal{L}(\mathbf{C}; i), \mathcal{L}(\mathbf{A}; i), \text{decoding}\}} \tag{6.18}$$

$$\mathcal{L}(b_m; p) = \ln \frac{P\{b_m = +1|\mathcal{L}(\mathbf{C}; i), \mathcal{L}(\mathbf{A}; i), \text{decoding}\}}{P\{b_m = -1|\mathcal{L}(\mathbf{C}; i), \mathcal{L}(\mathbf{A}; i), \text{decoding}\}} \tag{6.19}$$

where the input $\mathcal{L}(\mathbf{A}; i)$ is initialized to zero, assuming equally likely source information bits. The extrinsic information of code bits from the M_T outer decoders is given by

$$\mathcal{L}(\mathbf{C}; e) = \mathcal{L}(\mathbf{C}; p) - \mathcal{L}(\mathbf{C}; i) \tag{6.20}$$

This extrinsic information yields the intrinsic information for the inner SISO detector after interleaving as follows

$$\mathcal{L}(\mathbf{A}; i) = \Pi(\mathcal{L}(\mathbf{C}; e)) \tag{6.21}$$

The expectations of the transmitted bits are computed from $\mathcal{L}(\mathbf{A}; i)$ as follows

$$\mathcal{E}\{a_m\} = \tanh\left(\frac{\mathcal{L}(a_m; i)}{2}\right), \; m = 1, 2, \ldots, M_T \tag{6.22}$$

These soft estimates are used to produce $\mathcal{E}\{\mathrm{s}_m\}$ in (6.15). An estimate of the information bit matrix B is obtained by slicing the LLR, $\mathcal{L}(\mathbf{B}; p)$, at the outer decoders

$$\widehat{\mathrm{B}} = \mathcal{D}(\mathcal{L}(\mathbf{B}; p)) \tag{6.23}$$

where $\mathcal{D}(\cdot)$ is a decision device (or slicer).

The procedures explained so far are repeated until the decoding algorithm converges to a satisfactory performance.

6.4 Iterative Channel Estimation using Snapshot Approaches

In many cases where only a known pilot sequence is used, a channel estimate may not be accurate, due to the relatively short pilot sequence. It may in turn cause a significant performance degradation at the receiver which may not be fully compensated by the IDD receiver [197]. Iterative channel estimation is performed using the dedicated pilot symbols and the estimated code symbols. There are two soft information available for iterative channel estimation. These are the *extrinsic* information and the *a posteriori* information both from the outer decoders. It has been revealed that the former performs better than the latter for CDMA multiuser receivers [200]. As shown in Fig. 6.3(b), we use the extrinsic information for iterative channel estimation.

The pilot symbols located in a preamble take the first L symbols of the whole symbol frame of length P. In order to describe iterative channel estimation, it is convenient to define a stacked model of the received signals. The received signal vector at the n-th antenna is given by

$$\mathbf{r}_n = \mathbf{S}\,\mathbf{h}_n + \mathbf{v}_n \tag{6.24}$$

where the symbol matrix S consists of the pilot and the transmitted symbols from each antenna

given by

$$\mathbf{S} = \begin{bmatrix} s_1[1] & s_2[1] & \dots & s_M[1] \\ s_1[2] & s_2[2] & \dots & s_M[2] \\ \vdots & \vdots & \ddots & \vdots \\ s_1[P] & s_2[P] & \dots & s_M[P] \end{bmatrix},$$

$\mathbf{h}_n = [h_{n1}h_{n2}\cdots h_{nM_T}]^H$, and $\mathbf{v}_n = [v_n[1]\, v_n[2]\cdots v_n[P]]^H$. Eventually, we collect all vectors for all M_R receive antennas into a single $(PM_R \times 1)$ vector $\underline{\mathbf{r}}$ to represent the stacked model, as follows

$$\underline{\mathbf{r}} = (\mathbf{I}_{M_R} \otimes \mathbf{S})\,\mathbf{h} + \underline{\mathbf{v}} \triangleq \mathcal{S}\,\mathbf{h} + \underline{\mathbf{v}} \tag{6.25}$$

where $\underline{\mathbf{r}} = [\mathbf{r}_1^T\mathbf{r}_2^T\cdots\mathbf{r}_{M_R}^T]^T$, $\mathbf{I}_{M_R}$ is a M_R-by-M_R identity matrix, $\otimes$ is the Kronecker tensor product, $\underline{\mathbf{v}} = [\mathbf{v}_1^T\mathbf{v}_2^T\cdots\mathbf{v}_{M_R}^T]^T$, and $\mathcal{S}$ is the block-diagonal $(PM_R \times M_RM_T)$ matrix. In this way, all the unknown channel impulse responses are collected in the $(M_RM_T \times 1)$ vector $\mathbf{h}$.

At the first iteration, an initial channel estimate uses only known pilot symbols. As the iteration proceeds between the SISO detector and the SISO decoders, both the pilot and the estimated code symbols are contributed to iterative channel estimation. In this Chapter we propose the use of two types of the estimated code symbols: *soft* and *hard* decisions. The soft and hard estimates of the soft information $\mathcal{L}(\mathbf{A};i)$ from the SISO decoders are obtained by,

$$\widetilde{\mathbf{A}} = \tanh\left(\frac{\mathcal{L}(\mathbf{A};i)}{2}\right) \tag{6.26}$$

$$\widehat{\mathbf{A}} = \mathcal{D}(\mathcal{L}(\mathbf{A};i)) \tag{6.27}$$

The estimates, $\widetilde{\mathbf{A}}$ and $\widehat{\mathbf{A}}$, are reorganized to constitute the estimated symbol matrices, $\widetilde{\mathbf{S}}$ and $\widehat{\mathbf{S}}$, respectively. During the iteration, the block-diagonal matrix $\mathcal{S}$ is replaced with $\widetilde{\mathcal{S}}$ (or $\widehat{\mathcal{S}}$) in the channel estimators given in the succeeding subsections.

6.4.1 Least-Squares (LS) Estimation

Assume that no assumptions are made about the statistics of the fading nor of the additive white Gaussian noise. The LS solution minimizing the squared error quantity yields the following esti-

mator

$$\hat{\mathbf{h}}_{\text{LS}} = \arg\min_{\mathbf{h}}\{\| \underline{\mathbf{r}} - \mathcal{S}\mathbf{h} \|^2\} = (\mathcal{S}^H\mathcal{S})^{-1}\mathcal{S}^H\underline{\mathbf{r}} \tag{6.28}$$

6.4.2 Linear Minimum Mean-Square Error (MMSE) Estimation

We constrain the MMSE channel estimator to be a linear function of $\underline{\mathbf{r}}$. By using the Wiener-Hopf equation [12] and an approximation $\mathcal{E}\{\mathcal{S}\mathcal{S}^H\} \cong \mathcal{S}\mathcal{S}^H$, the approximated linear MMSE solution is given by

$$\hat{\mathbf{h}}_{\text{MMSE}} = (\mathcal{S}^H\mathcal{S} + \sigma_v^2\mathbf{I})^{-1}\mathcal{S}^H\underline{\mathbf{r}} \tag{6.29}$$

The MMSE estimator above assumes *i.i.d.* and complex Gaussian channel vectors. It requires the knowledge of the noise variance. The LS estimator in (6.28) is therefore a special case of the MMSE estimator with no requirement of the noise power.

6.5 Adaptive Filtering-Based Estimation

Under the assumption of spatially uncorrelated channels for sufficiently large antenna spacings, an $(M_R \times M_T)$ MIMO channel can be separated into M_R multiple-input single-output (MISO) sub-channels. Therefore, channel estimation can be performed independently for each receive antenna, leading to computational efficiency in implementing the following adaptive filters. In what follows, it is convenient to use the notation of the channel vector $\mathbf{h}[k] \triangleq \text{vec}\{\mathbf{H}^H[k]\} = [\mathbf{h}_1[k]\,\mathbf{h}_2[k]\,\ldots\,\mathbf{h}_{M_R}[k]]$ of $(M_T M_R \times 1)$ dimension that is obtained by stacking all columns of the Hermitian transpose of the channel matrix $\mathbf{H}[k]$. The wireless channel can be modeled as a stochastic 1st-order autoregressive (AR) process of the form:

$$\mathbf{h}_n[k] = \mathbf{F}\mathbf{h}_n[k-1] + \mathbf{w}_n[k] \in \mathcal{C}^{(M_T \times 1)} \tag{6.30}$$

where $n = 1, 2, \cdots, M_R$ and $\mathbf{w}_n[k]$ is the driving noise vector of the process, and the state transition matrix $\mathbf{F}$ modeling the spatio-temporal correlations of the channel is defined as

$$\mathbf{F} = \alpha\mathbf{I} \in \mathcal{R}^{(M_T \times M_T)}$$

with $\alpha = J_o(2\pi f_D T_s) \leq 1$, where $J_0(\cdot)$ is the 0th-order Bessel function of the first kind, and f_D and T_s denote the Doppler spread and the reciprocal bandwidth (e.g., symbol period), respectively.

6.5.1 Least Mean-Square (LMS) Estimation

As it follows from Eqs. (6.28) and (6.29), the iterative channel estimators include a matrix inversion which leads to a heavy computational burden. To reduce the computational complexity, a stochastic adaptation of the estimate can be used. For convenience of computation, we separate the estimated channel matrix $\widehat{\mathbf{H}}$ into sub-vectors, *i.e.*,

$$\widehat{\mathbf{H}}^H = [\hat{\mathbf{h}}_1\, \hat{\mathbf{h}}_2\, \ldots, \hat{\mathbf{h}}_{M_R}]$$

where $\hat{\mathbf{h}}_n = [\hat{h}_{n1}\, \hat{h}_{n2} \ldots \hat{h}_{nM_T}]^H$ $(n = 1, 2, \ldots, M_R)$. Three different error signals at time k can be defined as

$$e[k] = r_n[k] - \hat{\mathbf{h}}_n^H[k]\mathbf{s}[k] \tag{6.31}$$

$$\tilde{e}[k] = r_n[k] - \hat{\mathbf{h}}_n^H[k]\tilde{\mathbf{s}}[k] \tag{6.32}$$

$$\hat{e}[k] = r_n[k] - \hat{\mathbf{h}}_n^H[k]\hat{\mathbf{s}}[k] \tag{6.33}$$

where $\tilde{\mathbf{s}}[k]$ and $\hat{\mathbf{s}}[k]$ denote the soft and the hard estimates of the code symbols provided by the outer decoders. Channel state vectors $\hat{\mathbf{h}}_n$ are updated according to the LMS solution [12]

$$\hat{\mathbf{h}}_n[k+1] = \hat{\mathbf{h}}_n[k] + \mu \mathbf{s}[k] e^*[k] \tag{6.34}$$

where μ is the step-size parameter. During the iteration, the error $e[k]$ is replaced with $\tilde{e}[k]$ or $\hat{e}[k]$.

6.5.2 RLS-based Estimation

The performance index to be minimized by the RLS algorithm is defined as an exponentially windowed sum of the squared error:

$$\mathcal{J}[k] = \sum_{i=1}^{k} \lambda^{k-i} |e_n[i]|^2.$$

Estimating the channel $\mathbf{h}_n[k]$ minimizing $\mathcal{J}[k]$ can be realized by a recursive form. It can be noted that the standard RLS recursion is modified to an extended RLS form, when the process equation with the 1st-order AR model described in (6.30) is assumed. The extended RLS algorithm is represented as follows [12]:

$$\begin{aligned}
\mathbf{k}_n[k] &= \alpha \mathbf{P}_n[k, k-1]\mathbf{s}[k]\mathbf{A}_n[k] \\
e_n[k] &= r_n[k] - \hat{\mathbf{h}}_n^H[k|k-1]\mathbf{s}[k] \\
\hat{\mathbf{h}}_n[k+1|k] &= \alpha \hat{\mathbf{h}}_n[k|k-1] + \mathbf{k}_n[k]e_n^*[k] \\
\mathbf{P}_n[k] &= \mathbf{P}_n[k, k-1] - \mathbf{B}_n[k] \\
\mathbf{P}_n[k+1, k] &= \alpha^2 \mathbf{P}_n[k] + \mathbf{R}_n[k]
\end{aligned}$$

where $\mathbf{R}_n[k]$ is the correlation matrix of the process noise vector, and

$$\begin{aligned}
\mathbf{A}_n[k] &= \left(\mathbf{s}^H[k]\mathbf{P}_n[k, k-1]\mathbf{s}[k] + 1\right)^{-1} \\
\mathbf{B}_n[k] &= \frac{\mathbf{P}_n[k, k-1]\mathbf{s}[k]\mathbf{s}^H[k]\mathbf{P}_n[k, k-1]}{1 + \mathbf{s}^H[k]\mathbf{P}_n[k, k-1]\mathbf{s}[k]}.
\end{aligned}$$

6.5.3 KF-based Estimation

Equations (6.30) and (6.24) constitute the state-space model of the system, suitable for KF applications. To estimate the channel $\mathbf{h}_n[k]$, the KF algorithm based on a one-step prediction is represented as follows:

$$\begin{aligned}
\mathbf{K}_n[k] &= \mathbf{F}[k+1, k]\mathbf{P}_n[k, k-1]\mathbf{s}^H[k]\mathbf{C}_n[k] \\
\hat{\mathbf{h}}_n[k+1|k] &= \mathbf{F}[k+1, k]\hat{\mathbf{h}}_n[k|k-1] + \mathbf{K}_n[k]e_n[k] \\
\mathbf{P}_n[k] &= \left(\mathbf{I} - \mathbf{F}[k, k-1]\mathbf{K}_n[k]\mathbf{s}^H[k]\right)\mathbf{P}_n[k, k-1] \\
\mathbf{P}_n[k+1, k] &= \mathbf{F}[k+1, k]\mathbf{P}_n[k]\mathbf{F}^H[k+1, k] + \mathbf{R}_n[k]
\end{aligned}$$

where

$$\begin{aligned}
\mathbf{C}_n[k] &= \left(\mathbf{s}^H[k]\mathbf{P}_n[k, k-1]\mathbf{s}[k] + \mathbf{Q}_n[k]\right)^{-1} \\
e_n[k] &= r_n[k] - \mathbf{s}^H[k]\hat{\mathbf{h}}_n[k|k-1].
\end{aligned}$$

In the above recursion, $\mathbf{R}_n$ denotes the correlation matrix of the process noise, and $\mathbf{Q}_n$ denotes the correlation matrix of the measurement noise which has to be estimated.

6.6 Performance Evaluation

Monte Carlo simulations for turbo-BLAST with different channel estimators are performed in a MIMO link with $M_T = 4$ transmit and $M_R = 4$ receive antennas. At the transmit end, each substream of 100 information bits is independently encoded with a rate-1/2 convolutional code generator $(7, 5)_{\text{Oct}}$, and then interleaved via a space-time interleaver. The interleaved substreams are modulated with BPSK for simplicity. Pilot symbols of length 20 are dedicated to channel estimation. The number of iterations is restricted to 5 throughout the simulations. For performance comparison we use the average normalized square error (ANSE)[2], the bit error rate (BER) and the frame error rate (FER). In particular, the BER or the FER are a measure to evaluate the QoS defined for the link between two communicating users.

Plots of the ANSE versus the iteration for LS and MMSE estimators are drawn in Fig. 6.4. In the figures, the ANSE at the first iteration depicts initial estimation using only pilot symbols. We observe that the LS and MMSE estimators using soft decision feedback do not converge at a lower SNR (0 dB) and start to converge slowly at higher SNRs ($\geq$ 6 dB) after the second iteration. However, the LS and MMSE estimators with hard decision feedback converge fast towards a lower ANSE. At the initial estimation, the result of the MMSE estimator is marginally better than that of the LS estimator, thanks to using the knowledge of the noise variance. Fig. 6.5 depicts the convergence behavior of the LMS estimator. We also observe in the LMS estimator that hard decision feedback is superior to soft decision feedback. As argued in [216], we expected that hard decisions could be unreliable in the initial iteration, and thus may lead to erroneous estimates which propagate through the iteration process. Nevertheless, the superiority of all the estimators with hard decisions is in agreement. Intuitively, both the hard and the soft decisions become more reliable as the iteration proceeds, but the soft decisions still remain erroneous estimates which bring a negative effect to the channel estimators. This fact may be justified with Fig. 6.6 that sketches the histograms of the evolution for the soft and the hard decisions vs. the iteration, when

[2]The ANSE is defined as ANSE $= \mathcal{E}\{\| \mathbf{h} - \hat{\mathbf{h}} \|^2 / \| \mathbf{h} \|^2\}$[200][216].

the LMS estimator is employed at a SNR of 4 dB.

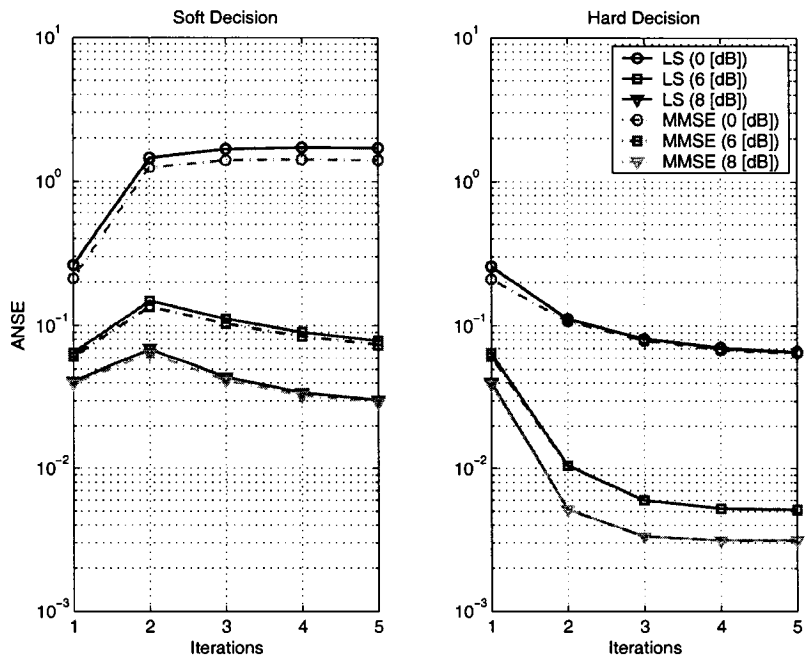


Fig. 6.4 Channel estimation errors of LS and MMSE estimators.

The BER performance after the first, second and fifth iteration for the LS, MMSE and LMS estimators is illustrated in Fig. 6.7, 6.8 and 6.9, respectively. For the LMS estimator, LS estimation is used for the initial channel estimation, where we assume that the noise power is not available at the receiver. As shown in the figures, the iteration gain is large. We observe that after the fifth iteration the BER performance of all the estimators using hard decision feedback approaches that of the perfectly known receiver. The performance gap of hard decision-based estimators at a BER of 10^{-4} is within $0.2 \sim 0.3$ dB, while that of the soft decision-based estimators is around $0.5 \sim 0.7$ dB. Like the results with respect to ANSE, the hard decision-based estimators after the fifth iteration are superior to the soft decision-based estimators in terms of BER performance.

The FER performance is shown in Fig. 6.10 for the three estimators. In general, all the hard decision-based estimators outperform the soft decision-based ones with respect to FER performance. In particular, the performance of the hard decision based-LMS estimator is comparable with that of the perfectly known channel at a lower SNR ($\leq$4 dB), and it is superior to the hard

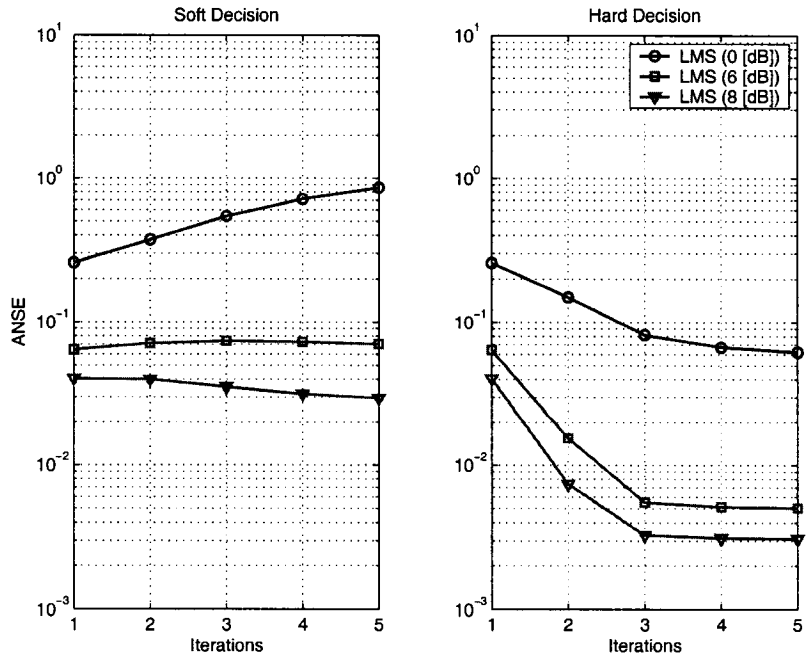


Fig. 6.5 Channel estimation errors of LMS estimator.

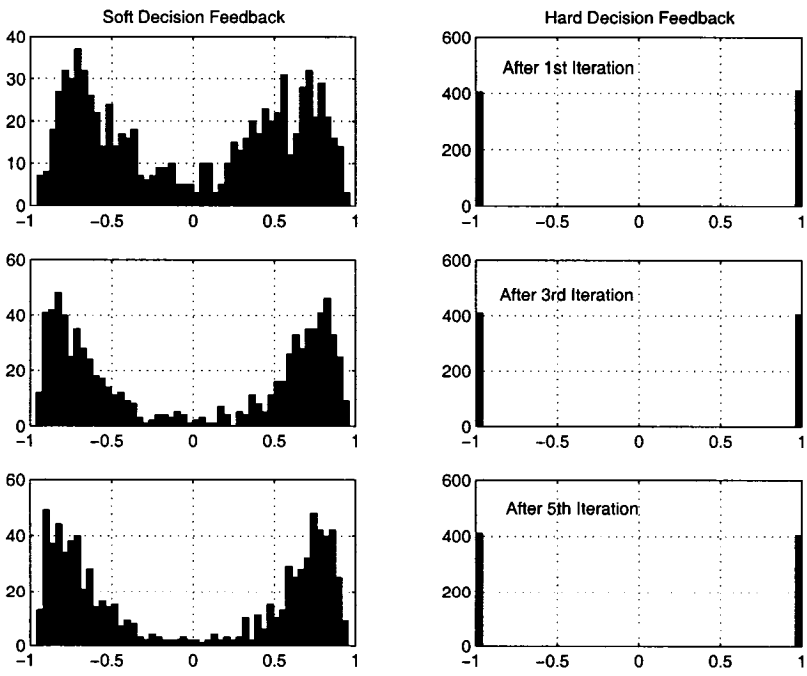


Fig. 6.6 Histograms of soft/hard Decisions for LMS estimator.

decision based-LS and MMSE estimators over all SNRs.

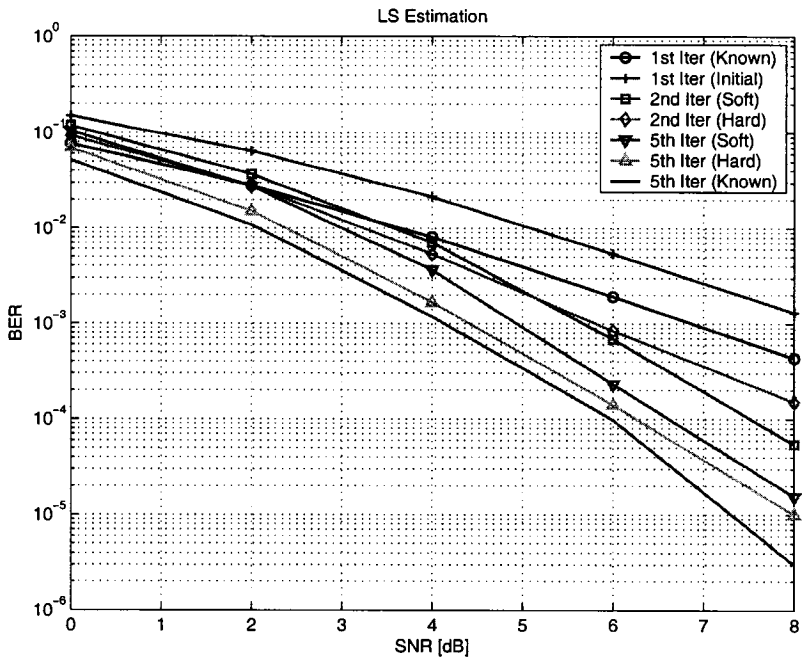


Fig. 6.7 BER performance of T-BLAST with LS estimator.

Fig. 6.11 shows the channel estimation error vs. the iteration for different SNR values and with the three adaptive filtering-based estimators. In the simulation, the ANSE is ensemble-averaged by 100 trials where $\mu = 0.005$ and $\lambda = 0.999$ for the LMS and the RLS filter, respectively. The ANSE at the first iteration depicts an initial estimate by the LS algorithm using only the pilot symbols. We observe that the RLS and the KF estimators converge faster than the LMS estimator until the second iteration. At the third iteration and beyond, the LMS estimator reaches almost identical ANSEs in steady-state performance, compared with the RLS and the KF estimators.

The BER performance of the three estimators is depicted in Fig. 6.12 where 5000 data blocks in total are transmitted. As seen, the iteration gain is large. We observe that after only three iterations, the BER performance gap between the receiver with the known CSI and the receivers with the iterative channel estimators is within 0.5 dB. In other words, more than 1 dB improvement is achieved by iterative channel estimation. As supported by the channel estimation error, the LMS estimator has attained an equivalent BER performance to the RLS and the KF estimators. As more

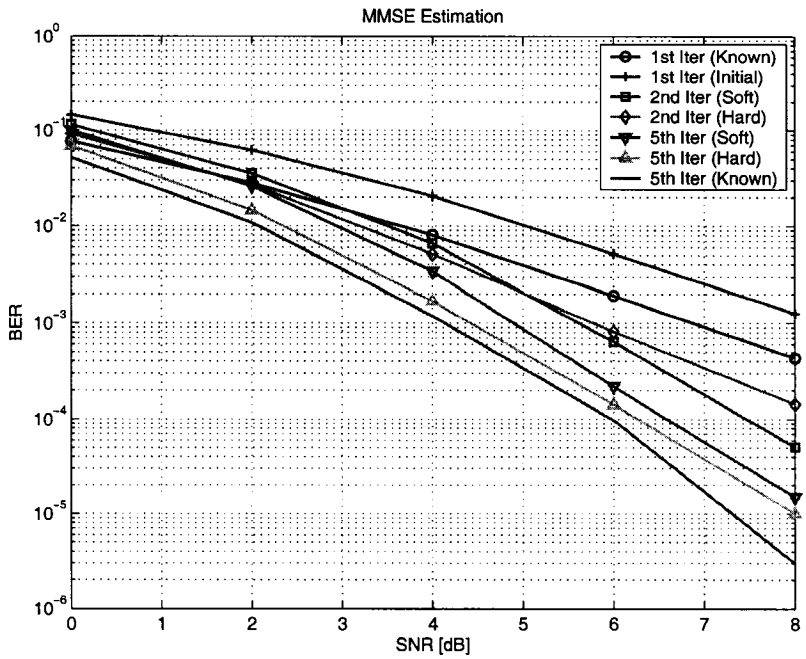


Fig. 6.8 BER performance of T-BLAST with MMSE estimator.

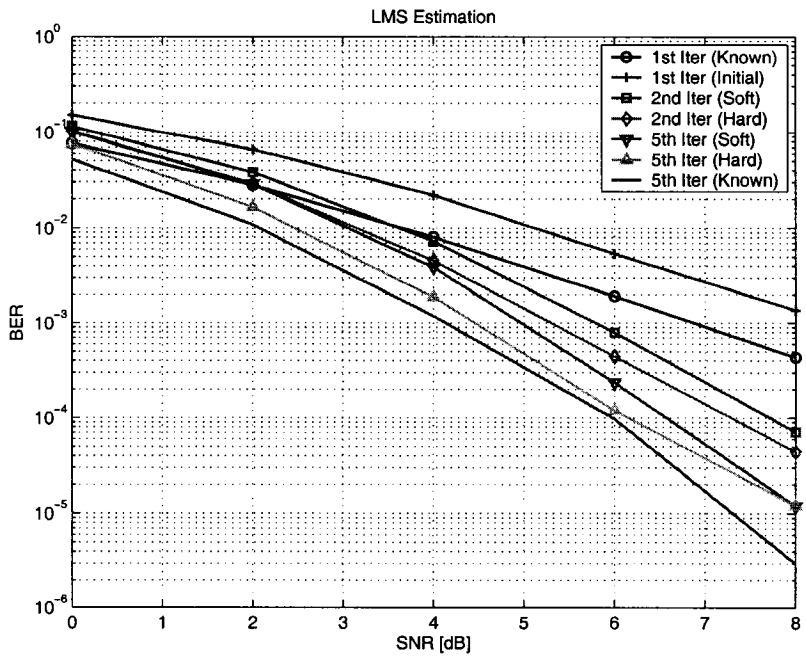


Fig. 6.9 BER performance of T-BLAST with LMS estimator.

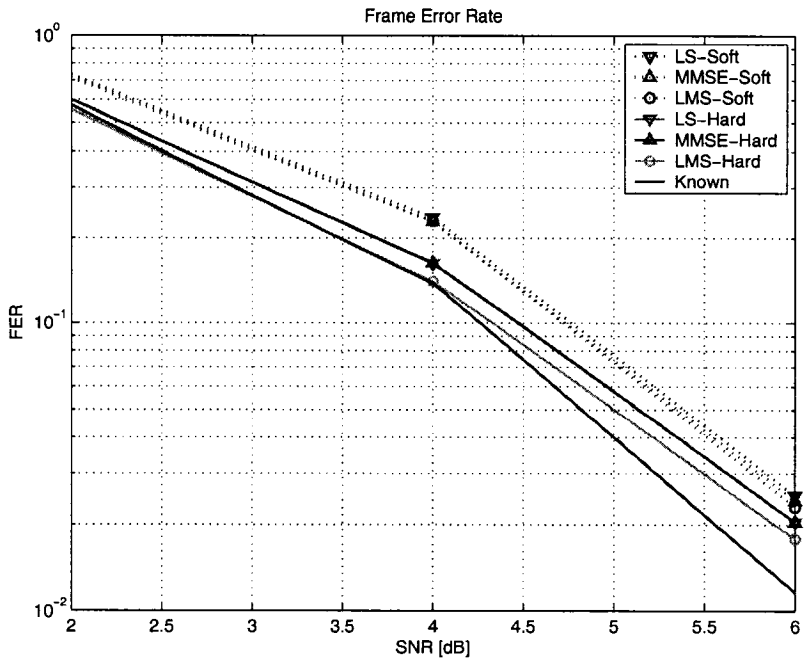


Fig. 6.10 FER performance comparison of T-BLAST with iterative estimators.

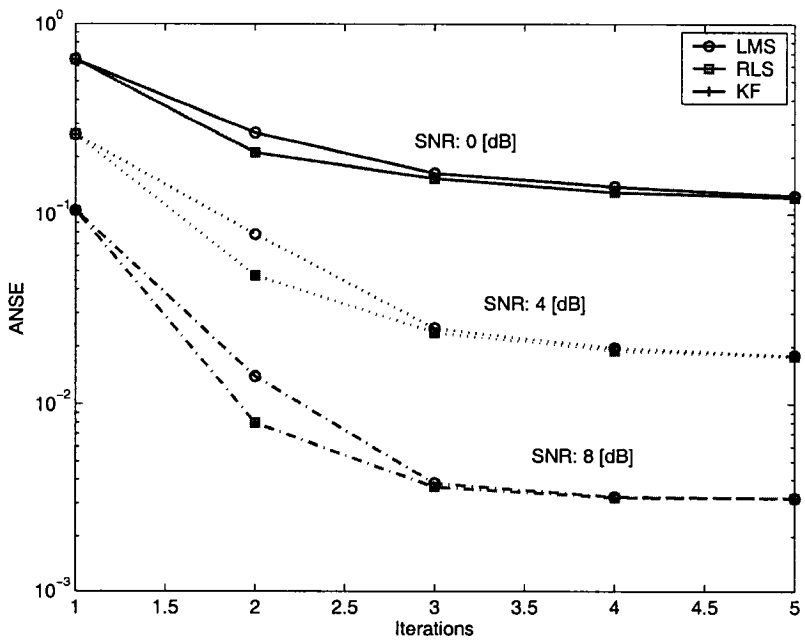


Fig. 6.11 Channel estimation errors.

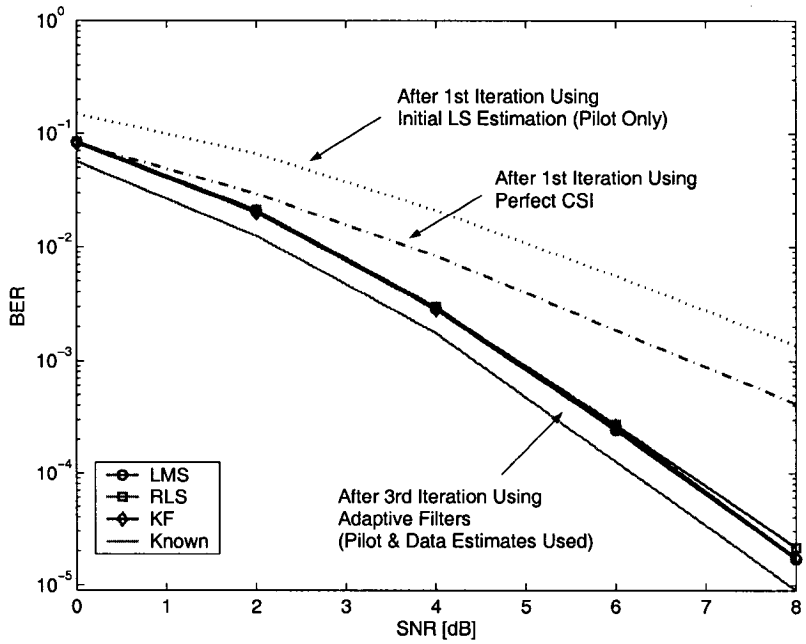


Fig. 6.12 BER performance comparison.

iterations proceed the performance is improved, but it may not be significant because the estimators reach a steady state within three or four iterations and a short block length is used.

6.7 Concluding Remarks

We have investigated the performance of iterative channel estimation techniques for turbo-BLAST over block-fading MIMO wireless channels. In addition to the training symbols, the extrinsic information aided soft and hard decisions of the transmitted symbols, provided by the outer SISO decoders, are fed back into the channel estimator in order to achieve more reliable CSI. The presented techniques with hard decision feedback, which are LS, MMSE and LMS estimators, outperform the equivalent techniques with soft feedback in terms of ANSE, BER, and FER. In particular, the hard decision-based LMS estimator with less complexity has revealed a better performance than the other two estimators. The reason why soft decision feedback-based estimators show inferior performance may be due to the use of erroneous soft decision feedback.

Selecting an adaptive filtering algorithm depends on system environments such as the fading

rate and the frame structure, and it depends on the parameters to be carefully chosen to implement themselves. In a quasi-static fading MIMO channel, we conclude that the LMS estimator is a reasonable choice in terms of computational efficiency, without the loss of performance gain compared to the RLS and the KF estimators.

For the work in this chapter we only considered a narrowband quasi-static Rayleigh fading environment. For a MIMO time-varying channel within a packet (or block) where channel tracking is needed, further research on iterative channel estimation using the KF is ongoing. Regarding channel coding of turbo-BLAST, only convolutional codes were used. Other advanced coding schemes, such as turbo codes or low density parity check codes, are also applicable to turbo-BLAST.

Chapter 7

Particle Filtering for Iterative MIMO Channel Tracking

7.1 Introduction

The use of multiple-input multiple-output (MIMO) wireless communications offers high data rates and spectral efficiency through increased computational complexity, but without additional transmit power and channel bandwidth. When coherent signal detection is employed, channel state information (CSI) is estimated at the receiver in real wireless scenarios. Accurate CSI is a prerequisite for the full exploitation of the advantages of MIMO systems since channel estimation errors at the receiver reduce the MIMO channel capacity [332].

We consider an iterative MIMO receiver, known as turbo-BLAST [322]. Due to the iterative nature of turbo-BLAST, where the received signal is iteratively processed between the detector and the decoders, it is natural to use an iterative channel estimation scheme in a semiblind procedure. This is an efficient method for channel estimation by the use of both dedicated pilot symbols and estimated data symbols [334]. In wireless communications, the channel variations are governed by the Doppler spread. In order to track time variations of the channel, we use an iterative channel estimator with particle filtering [32], rooted on recursive Bayesian estimation. As the channel estimator, we exploit two particle filters, sequence importance resampling (SIR) filter and gradient proposal particle filter (GPF) [17]. The performance of the channel tracking algorithms using particle filters is demonstrated for time-varying MIMO channels in both Gaussian and non-Gaussian

noise.

This chapter is organized as follows. In Section 7.2 we give a description of particle filtering algorithms. Section 7.3 presents the system model of MIMO wireless systems, including ambient noise and channel modeling. Section 7.4 discusses pilot symbol placement for channel estimation and illustrates iterative channel tracking algorithms using particle filters. The performance of the turbo-BLAST with particle filtering channel tracking is demonstrated in Section 7.5. Finally, Section 7.6 concludes the chapter.

7.2 Particle Filtering

7.2.1 The Bayesian Solution

Signals and systems of sequential signal processing are represented by a dynamic state-space model (DSSM),

$$\mathbf{x}[k+1] = \mathbf{f}_k(\mathbf{x}[k], \mathbf{d}[k]) \tag{7.1}$$

$$\mathbf{y}[k] = \mathbf{g}_k(\mathbf{x}[k], \mathbf{v}[k]) \tag{7.2}$$

where $\mathbf{x}[k+1]$ is the state vector; $\mathbf{y}[k]$ is the observed vector at the receiver; $\mathbf{f}_k$ and $\mathbf{g}_k$ are generic vector-valued functions, which are possibly time-variant; $\mathbf{d}[k]$ and $\mathbf{v}[k]$ denote the dynamic and measurement noise processes, respectively. The state in (7.1) characterizes the state transition probability $p(\mathbf{x}[k+1] \mid \mathbf{x}[k])$, whereas the measurement in (7.2) describes the likelihood $p(\mathbf{y}[k] \mid \mathbf{x}[k])$. In general, there are three densities of interest, called the filtering, predictive, and smoothing densities. All the information about the state $\mathbf{x}[k]$ is captured by these densities, respectively. For instance, the filtering density $p(\mathbf{x}[k] \mid \mathbf{y}[1:k])$ can be obtained from $p(\mathbf{x}[k-1] \mid \mathbf{y}[1:k-1])$. Thus the main goal is their tracking, given the set of observation $\mathbf{y}[1:k] = \{\mathbf{y}[1], \mathbf{y}[2], \ldots, \mathbf{y}[k]\}$.

The filtering density is estimated recursively in two stages: prediction and update. In the prediction stage, the filtering density $p(\mathbf{x}[k]|\mathbf{y}[1:k-1])$ is propagated into the future by the transition density $p(\mathbf{x}[k-1]|\mathbf{x}[1:k-1])$ and it is obtained as follows:

$$p(\mathbf{x}[k]|\mathbf{y}[1:k-1]) = \int p(\mathbf{x}[k]|\mathbf{x}[k-1])p(\mathbf{x}[k-1]|\mathbf{y}[1:k-1])d\mathbf{x}[k-1]. \tag{7.3}$$

where the required density $p(\mathbf{x}[k-1]|\mathbf{y}[1:k-1])$ is available at time $k-1$. The transition density $p(\mathbf{x}[k]|\mathbf{x}[k-1])$ is defined in terms of the probabilistic model governing the state's evolution (see (7.1)) and the process noise statistics.

The update stage involves the application of Bayes' rule when the data are observed. The marginal density becomes

$$p(\mathbf{x}[k]|\mathbf{y}[1:k]) = \frac{p(\mathbf{y}[k]|\mathbf{x}[k])p(\mathbf{x}[k]|\mathbf{y}[1:k-1])}{p(\mathbf{y}[k]|\mathbf{y}[1:k-1])}. \tag{7.4}$$

where the likelihood density $p(\mathbf{y}[k]|\mathbf{x}[k])$ is defined in terms of the measurement model in (7.2), and the normalizing constant

$$p(\mathbf{y}[k]|\mathbf{y}[1:k-1]) = \int p(\mathbf{y}[k]|\mathbf{x}[k])p(\mathbf{x}[k]|\mathbf{y}[1:k-1])d\mathbf{x}[k]. \tag{7.5}$$

The normalizing constant, often referred to as the evidence density, plays a key role in optimization schemes. The prediction and update strategy given in (7.3) and (7.4) provides an optimal solution to the estimation problem. Unfortunately, it includes multidimensional integrations, the source of the practical difficulties inherent in the Bayesian methodology.

Many signal processing problems, such as communication systems, financial time series, medical prognosis, target tracking, and geographical data analysis, involve elements of non-Gaussianity, nonlinearity, and nonstationarity. Consequently, it is not possible to derive exact closed-form estimators based on the standard criteria of maximum likelihood (ML), maximum *a posteriori* probability (MAP), or minimum mean-squared error (MMSE). Monte carlo (MC) simulation methods have become an important alternative to the conventional methods, including the Kalman filter, Gaussian approximations (EKF and mixtures of Gaussian densities), and direct numerical integration [335]. The accuracy of the analysis can be improved by the MC simulation methods, which provide a complete description of the probability distribution of the data. Various MC simulation-based filtering methods have been developed, such as the bootstrap filter or the MC filter [16]. These filtering algorithms can be treated under the generalized particle filtering framework.

7.2.2 Sequential Importance Sampling Filter

The key concept of particle filtering is to represent the posteriori density function by a set of random samples, generated by MC simulations, with associated weights, and to compute the estimates based on these samples and weights. Assume that the distribution of the target is $p(\mathbf{x})$ and its approximation of random measure is $\{\mathbf{x}^i, w^i\}_{i=1}^N$ with the particles $\mathbf{x}^i$ and their weights w^i, and the number of particles N used in the approximation. In many situations, it is often hard to draw samples from the target distribution $p(\mathbf{x})$. To avoid this difficulty, importance sampling can be used. To do that, we introduce a known function $q(\mathbf{x})$, known as a proposal distribution (or importance density), which is close in shape to the $p(\mathbf{x})$. Then $\{\mathbf{x}^i, w^i\}_{i=1}^N$ approximates the distribution $p(\mathbf{x})$ by

$$\hat{p}(\mathbf{x}) \approx \sum_{i=1}^{N} w^i \delta(\mathbf{x} - \mathbf{x}^i) \tag{7.6}$$

where the $w^i = p(\mathbf{x}^i)/q(\mathbf{x}^i)$ are called the importance weights and $\delta(\cdot)$ is the Dirac delta function.

Let $\{\mathbf{x}^i[0:k], w^i[k]\}_{i=1}^N$ denote a random measure characterizing the posterior density $p(\mathbf{x}[0:k]|\mathbf{y}[1:k])$, The posterior density $p(\mathbf{x}[0:k]|\mathbf{y}[1:k])$ at k can be approximated as

$$p(\mathbf{x}[0:k]|\mathbf{y}[1:k]) \approx \sum_{i=1}^{N} w^i[k] \delta(\mathbf{x}[0:k] - \mathbf{x}^i[0:k]) \tag{7.7}$$

The weights are chosen using the principle of importance sampling. Therefore, if the samples $\mathbf{x}^i[0:k]$ were drawn from an importance density $q(\mathbf{x}[0:k]|\mathbf{y}[1:k])$, then the weights in (7.7) can be computed as [32]

$$\begin{aligned} w^i[k] &\propto \frac{p(\mathbf{x}^i[0:k]|\mathbf{y}[1:k])}{q(\mathbf{x}^i[0:k]|\mathbf{y}[1:k])} \\ &\propto \frac{p(\mathbf{y}[k]|\mathbf{x}^i[k])p(\mathbf{x}^i[k]|\mathbf{x}^i[k-1])p(\mathbf{x}^i[0:k-1]|\mathbf{y}[1:k-1])}{q(\mathbf{x}^i[k]|\mathbf{x}[0:k-1],\mathbf{y}[1:k])q(\mathbf{x}^i[0:k-1]|\mathbf{y}[1:k-1])} \\ &= w^i[k-1]\frac{p(\mathbf{y}[k]\mathbf{x}^i[k])p(\mathbf{x}^i[k]|\mathbf{x}^i[k-1])}{q(\mathbf{x}^i[k]|\mathbf{x}^i[0:k-1],\mathbf{y}[1:k])}. \end{aligned} \tag{7.8}$$

By applying $q(\mathbf{x}^i[k]|\mathbf{x}^i[0:k-1],\mathbf{y}[1:k]) = q(\mathbf{x}^i[k]|\mathbf{x}^i[k-1],\mathbf{y}[k])$, the importance weights can

be recursively estimated as follows:

$$w^i[k] = w^i[k-1]\frac{p(\mathbf{y}[k]|\mathbf{x}^i[k])p(\mathbf{x}^i[k]|\mathbf{x}^i[k-1])}{q(\mathbf{x}^i[k]|\mathbf{x}^i[0:k-1],\mathbf{y}[k])}. \tag{7.9}$$

This equation underlies the basic principle of sequential importance sampling (SIS) filter. The posterior filtering density $p(\mathbf{x}[k]|\mathbf{y}[1:k])$ can be approximated as

$$p(\mathbf{x}[k]|\mathbf{y}[1:k]) \approx \sum_{i=1}^{N} \tilde{w}^i[k]\delta(\mathbf{x}[k] - \mathbf{x}^i[k]) \tag{7.10}$$

where the normalized weights are

$$\tilde{w}^i[k] = \frac{w^i[k]}{\sum_{j=1}^{N} w^j[k]}. \tag{7.11}$$

It can be noted that the approximation (7.10) approaches the true posterior density as $N \rightarrow \infty$. This SIS filter consists of a recursive propagation of the particles and their weights as each measurement is received sequentially. The SIS filter forms the basis for most particle filtering algorithms developed so far.

Degeneracy Problem

The SIS filter has a serious limitation, which is often referred to as a *weight degeneracy* problem. What happens is that only a few particles have nonzero importance weights after a few iterations. Due to this degeneracy phenomenon, the variance of the importance weights increases over time when the observations are regarded as random [26]. To understand why the variance increases, suppose that we want to sample from the posterior density. In that case, we want the proposal density to be very close to the posterior density, i.e. $p(\mathbf{x}[0:k]|\mathbf{y}[1:k]) \approx q(\mathbf{x}[0:k]|\mathbf{y}[1:k])$. When this happens, the mean and the variance of the importance weights are

$$\mathcal{E}_{q(.|\mathbf{y}[1:n])}\left[\frac{p(\mathbf{x}[0:k]|\mathbf{y}[1:k])}{q(\mathbf{x}[0:k]|\mathbf{y}[1:k])}\right] = 1 \tag{7.12}$$

$$var_{q(.|\mathbf{y}[1:n])}\left[\frac{p(\mathbf{x}[0:k]|\mathbf{y}[1:k])}{q(\mathbf{x}[0:k]|\mathbf{y}[1:k])}\right] = \mathcal{E}_{q(.|\mathbf{y}[1:n])}\left[\left(\frac{p(\mathbf{x}[0:k]|\mathbf{y}[1:k])}{q(\mathbf{x}[0:k]|\mathbf{y}[1:k])} - \mathcal{E}_{q(.|\mathbf{y}[1:n])}\left[\frac{p(\mathbf{x}[0:k]|\mathbf{y}[1:k])}{q(\mathbf{x}[0:k]|\mathbf{y}[1:k])}\right]\right)^2\right] = 0. \quad (7.13)$$

In other words, we expect that the variance is to be close to zero in order to obtain reasonable estimates. Typically, what we observe is that after a few iterations, one of the normalized importance weights tends to one, whereas the remaining weights tend to zero. Therefore, a large computational effort is devoted to updating the particles which do not contribute to the final estimate.

The degeneracy can be monitored by the effective sample size N_{eff}, which is defined as [26],[25]

$$\begin{aligned} N_{\text{eff}} &= \frac{N}{1 + var_{q(.|\mathbf{y}[1:n])}[\breve{w}(\mathbf{x}[0:k])]} \\ &= \frac{N}{\mathcal{E}_{q(.|\mathbf{y}[1:n])}[(\breve{w}(\mathbf{x}[0:k]))^2]} \leq N \end{aligned} \quad (7.14)$$

where $\breve{w}(\cdot)$ is referred to as the "true weight." In practice, N_{eff} can not be evaluated exactly, but its estimate $\hat{N}_{\text{eff}}$ can be approximated by

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N}(w^i[k])^2}. \quad (7.15)$$

A practical method to reduce the degeneracy phenomenon is to use *resampling* whenever a significant degeneracy is observed, i.e. when $\hat{N}_{\text{eff}} < N_{\text{th}}$ where N_{th} is a predefined threshold. It is noted that a small $\hat{N}_{\text{eff}}$ ($\hat{N}_{\text{eff}} \ll N$) indicates a severe degeneracy. Intuitively, in order to reduce the degeneracy effect, the brutal force method is to employ a large N. However, it is definitely undesirable. Other methods to reduce this undesirable effect are described next.

Resampling

To avoid the degeneracy of the SIS particle filter, a resampling step may be used to eliminate the samples with low importance weights and to multiply the samples with high importance weights. Efficient algorithms have been developed for accomplishing this task in $\mathcal{O}(N)$ operations [27],[29]. In the resampling step, a new particle set $\{\mathbf{x}^j[k]\}_{j=1}^N$ is generated by resampling N times from the previous particle set.

Resampling includes mapping the random measure $\{\mathbf{x}^i[k], \tilde{w}^i[k]\}$ into an equally weighted random measure $\{\mathbf{x}^j[k], N^{-1}\}$. This is accomplished by sampling uniformly from the discrete set $\{\mathbf{x}^i[k]\}_{i=1}^N$ with probabilities $\{\tilde{w}^i[k]\}_{i=1}^N$, where the index i is drawn from a uniform distribution. A mathematical proof of this can be found in [23]. A process of sampling from the discrete set is illustrated in Fig. 7.1. After constructing the cumulative distribution function (cdf) of the discrete set, a uniformly drawn sampling index i is projected onto the distribution range and then onto the distribution domain. The intersection with the domain constitutes the new sample index j. The particles $\mathbf{x}^j[k]$ are accepted as the new samples. Clearly, the particles with the larger sampling weights will end up with more copies after the resampling process. In the subsequent prediction stage, random disturbances are added to the samples, adding variety to the dominant samples. As suggested in [26], resampling should be performed only when the estimated effective sample size $\hat{N}_{\text{eff}}$ is below a predefined heuristic threshold N_{th}. A systematic resampling algorithm is described in Table 7.1 [336],[32], where $\mathcal{U}[0, N^{-1}]$ is the uniform distribution on the interval $[0, N^{-1}]$.

It is noted that resampling does not prevent the weight degeneracy problem, but it reduces the effects of the problem by discarding the particles with low importance weights. However, the resampling step introduces the following side effects [32]:

- *Limitation of parallel processing*: The parallel processing of the algorithm employing the resampling stage is limited, since all the particles must be combined.

- *Loss of diversity*: Since resampling replicates the particles with high importance weights many times, it also introduces correlations within them when there are only a few dominant weights. This leads to a *loss of diversity*. This problem is known as *sample impoverishment* and is severe in the case of small process noise. For instance, for the case of very small process noise, all the particles will collapse to a single particle within a few iterations. The problem of sample impoverishment can be solved by the *resample-move* algorithm [337] or *regularization* [16].

- *Degeneration of the estimates*: The particles actually represent paths through the state space by storing the trajectory taken by each particle. Since the diversity of the paths of the particles is diminished, any smoothed estimate of the state degenerates. To neutralize this effect, MCMC techniques [30] can be used.

Despite these limitations, resampling is the basis for numerous works and many strategies have been developed to increase its efficiency. A number of different resampling schemes have been proposed in the literature [28],[16],[336], including sequential importance sampling/multinomial resampling, residual resampling and satisfied sampling.

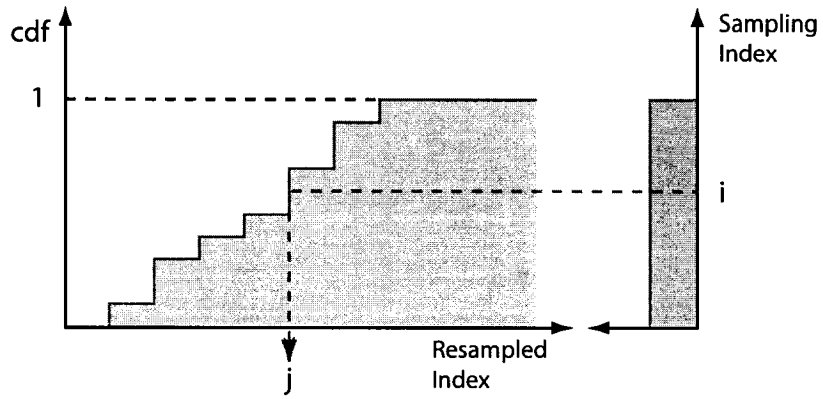


Fig. 7.1 Resampling process.

Table 7.1 Resampling algorithm.

1. *Construction of cdf*
   - At $i = 0$
     - Initialize the cdf: $c_1 = 0$
   - For $i = 2, \ldots, N$
     - Construct the cdf: $c_i = c_{i-1} + w^i[k]$
2. *Sampling Step*
   - Start at bottom of the cdf: $i = 1$
   - Draw a starting point: $u_1 \sim \mathcal{U}[0, N^{-1}]$
   - For $i = 1, \ldots, N$
     - Move along with the cdf: $u_j = u_1 + N^{-1}(j-1)$
     - If $(u_j > c_i)$, then $i = i + 1$
     - Assign sample: $\mathbf{x}^j[k] = \mathbf{x}^i[k]$
     - Assign weight: $w^j[k] = N^{-1}$

Choice of Proposal Distribution

The choice of a proposal distribution (or importance density) plays an important role in determining the performance of estimation, and is one of the most critical design issues in importance sampling algorithms. Choosing a good proposal distribution reduces the effects of weight degeneracy. In general, the choice of a proposal distribution is problem-dependent and demands a good understanding of the problem at hand. A good proposal distribution satisfies some goals:

- *Optimal distribution*: The proposal distribution should be chosen to minimize the variance of $w^i[k]$ so that N_{eff} is maximized. In [25] it has been shown that the proposal distribution

$$q(\mathbf{x}[k]|\mathbf{x}^i[0:k-1],\mathbf{y}[1:k]) = p(\mathbf{x}[k]|\mathbf{x}^i[k-1],\mathbf{y}[k]) \tag{7.16}$$

  is the optimal importance density that minimizes the variance of the importance weights conditional on $\mathbf{x}[0:k-1]$ and $\mathbf{y}[1:k]$. In this optimal case, the importance weights can be recursively calculated as:

$$w^i[k] = w^i[k-1]p(\mathbf{y}[k]|\mathbf{x}^i[k-1]) \tag{7.17}$$

  which follows from (7.9) and the Bayes' rule. Unfortunately, sampling from $p(\mathbf{x}[k]|\mathbf{x}^i[k-1],\mathbf{y}[k])$ and evaluating

$$p(\mathbf{y}[k]|\mathbf{x}^i[k-1]) = \int p(\mathbf{y}[k]|\mathbf{x}[k])p(\mathbf{x}[k]|\mathbf{x}^i[k-1])d\mathbf{x}[k] \tag{7.18}$$

  are not possible, except for special cases with analytic (approximate) solutions [25].

- *Ease of implementation*: The most popular choice of a proposal distribution is to use the prior density

$$q(\mathbf{x}[k]|\mathbf{x}^i[0:k-1],\mathbf{y}[1:k]) = p(\mathbf{x}[k]|\mathbf{x}^i[k-1]). \tag{7.19}$$

  By using this importance density, the importance weights are evaluated as follows:

$$w^i[k] \propto w^i[k-1]p(\mathbf{x}[k]|\mathbf{x}^i[k-1]) \tag{7.20}$$

often resulting in a reduced computational complexity of the sampling procedure. Although it results in higher MC variation than the optimal proposal distribution, it is easier to implement because the most recent observations are not incorporated in the proposal distribution. However, it may be inefficient and especially sensitive to outliers.

- *Heavy-tailed behavior*: The proposal distribution has a heavy-tailed behavior to account for outliers, which ensures that the unnormalized importance weights are upper bounded.
- *Close shape to the true posterior density*: The proposal distribution is as close as possible in shape to the true posterior density.

However, satisfying all of these goals is not easy. In practice, the choice of different proposal distributions leads to different kinds of particle filters.

In Fig. 7.2, we illustrates a pictorial description of a generic particle filter, including the random measures, the actual probability distributions of interest, and the three steps of particle filtering: 1) particle generation, 2) weight update, and 3) resampling.

7.2.3 Sampling Importance Resampling Filter

A variety of particle filters proposed in the literature have been derived from the SIS algorithm, by an approximate choice of proposal distribution and/or modification of the resampling step. The sampling importance resampling (SIR) filter, known as Bayesian *bootstrap filter* [28], can be applied to recursive Bayesian filtering problems. The derivation of the SIR filter assumes that: 1) the functions $\mathbf{f}_k$ and $\mathbf{g}_k$ in equations (7.1) and (7.2) are known; 2) it is possible to sample realizations from the process noise distribution $p_d(\mathbf{d}[k])$ and from the prior density $p(\mathbf{x}[k]|\mathbf{x}^i[k-1])$; and 3) the likelihood function is available in a functional form. By the use of two appropriate choices: 1) the resampling step at every time index, and 2) the prior density $p(\mathbf{x}[k]|\mathbf{x}^i[k-1])$ as the importance density $q(\mathbf{x}[k]|\mathbf{x}^i[k-1], \mathbf{y}[1:k])$, the SIR filter can be derived from the SIS filter.

The SIR algorithm is as follows:

1. **Initialization**: The filter is initialized by drawing a sample of size N from the prior density at time $k = 0$. The algorithm is then started with the filtering step.

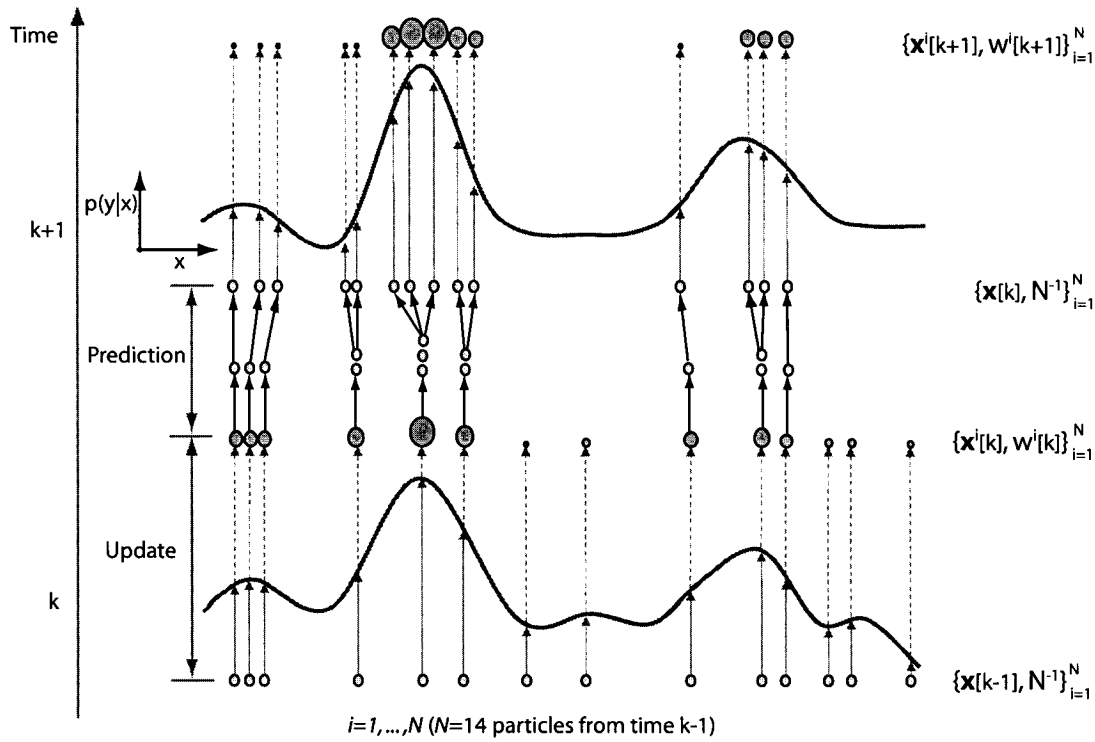


Fig. 7.2 A pictorial description of particle filtering with importance sampling and resampling.

2. **Preliminaries**: At time k, assume that $\{\mathbf{x}^i[k-1]\}_{i=1}^N$ is a swarm of N particles, approximately distributed as an independent sample from $p(\mathbf{x}[k-1]|\mathbf{y}[1:k-1])$.

3. **Prediction**: First, sample N values $\{\mathbf{d}^i[k-1]\}_{i=1}^N$ from the distribution of the process noise $\mathbf{d}[k]$

$$\mathbf{d}[k-1] \sim p_d(\mathbf{d}[k-1]). \tag{7.21}$$

Then, use these to generate a new swarm of particles, $\{\tilde{\mathbf{x}}^i[k]\}_{i=1}^N$

$$\tilde{\mathbf{x}}^i[k] \sim p(\mathbf{x}[k]|\mathbf{x}^i[k-1]) \tag{7.22}$$

by setting $\tilde{\mathbf{x}}^i[k] = \mathbf{f}_k(\mathbf{x}^i[k-1], \mathbf{d}^i[k-1])$.

4. **Filtering**: Assign for each $\tilde{\mathbf{x}}^i[k]$ a weight $w^i[k]$ given by

$$w^i[k] \propto w^i[k-1]p(\mathbf{y}[k]|\tilde{\mathbf{x}}^i[k]) \tag{7.23}$$

$$\propto p(\mathbf{y}[k]|\tilde{\mathbf{x}}^i[k]). \tag{7.24}$$

   The second line of the above equation follows from $w^i[k-1] = 1/N$ since resampling is applied at every time step. The weights in (7.24) are normalized before the resampling step.

5. **Resampling**: Resample independently N times, with replacement, from the distribution obtained in the filtering stage. The resulting particles $\{\mathbf{x}^i[k]\}_{i=1}^N$ form an approximate sample from $p(\mathbf{x}[k]|\mathbf{y}[1:k])$.

The pseudo-code for an implementation of SIR particle filtering is given in Table 7.2. Due to the resampling stage applied at every iteration, the SIR filter may lead to a rapid loss of diversity in particles. However, the SIR filter takes advantage of the fact that the important weights are easily evaluated and the importance density can be easily sampled. Many improvement schemes, such as satisfied sampling [16] and auxiliary variable [29], have been developed for the SIR filter.

7.2.4 Gradient Proposal Particle Filter

The SIR filter does not take account of the recent observation into the proposal distribution. To surmount this, an improved particle filter so-called gradient proposal particle filter (GPF) was proposed in [17], which takes advantage of the current observation. The idea of gradient proposal is borrowed from the hybrid gradient descent/SIR (HySIR) algorithm [338] that was originally derived for training neural networks. The GPF introduces a gradient "MOVE"-step in the sampling of the proposal distribution, which is plugged in after the importance sampling step in the SIR filter.

With the measurement equation in (7.2), the gradient MOVE-step is represented by

$$\mathbf{x}[k] = \check{\mathbf{x}}[k] - \eta \frac{\partial(\mathbf{y}[k] - \mathbf{g}_k(\mathbf{x}))^2}{\partial \mathbf{x}}\bigg|_{\mathbf{x}=\check{\mathbf{x}}[k]} \tag{7.25}$$

where $\eta\,(0 < \eta < 1)$ is a step-size parameter. In (7.25), $\check{\mathbf{x}}[k]$ denotes the predicted estimate from the state equation, i.e. the *a priori* sample drawn from $p(\cdot|\mathbf{x}[k-1])$, and $\mathbf{x}[k]$ denotes the sample

Table 7.2 SIR particle filtering algorithm.

1. *Initialization*
   At $k = 0$
   - For $i = 1, \ldots, N$,
     - Draw samples (or particles) $\mathbf{x}^i[0] \sim p(\mathbf{x}[0])$.
     - Set $w^i[0] = 1/N$.
2. *Importance Sampling Step* (Prediction and Filtering)
   For $k = 1, 2, \ldots$
   - For $i = 1, \ldots, N$,
     - Sample $\tilde{\mathbf{x}}^i[k]$ according to (7.22).
     - Set $\tilde{\mathbf{x}}^i[0:k] = (\mathbf{x}^i[0:k-1], \tilde{\mathbf{x}}^i[k])$.
   - For $i = 1, \ldots, N$,
     - Evaluate the importance weights $w^i[k]$ according to (7.24).
   - For $i = 1, \ldots, N$,
     - Normalize the importance weights $w^i[k]$ according to (7.11).
3. *Resampling Step*
   - Generate N new particles $\mathbf{x}^i[k]$ from the set $\{\tilde{\mathbf{x}}^i[k]\}$ according to the importance weights $\tilde{w}^i[k]$.

after the MOVE-step which is regarded as the *a posteriori* sample. The inclusion of the current measurement $\mathbf{y}[k]$ and gradient information will try to push the samples to a high-likelihood region, providing more reliable predictive samples for the next step. In this case, the proposal distribution can be derived as

$$\begin{aligned} q(\mathbf{x}[k]|\mathbf{x}^i[k-1], \mathbf{y}[k]) &= \int p(\mathbf{x}[k]|\check{\mathbf{x}}[k], \mathbf{y}[k]) p(\check{\mathbf{x}}[k]|\mathbf{x}[k-1]) d\check{\mathbf{x}}[k] \\ &= \int \delta(\mathbf{x}[k] - \check{\mathbf{x}}[k] + \eta\Phi(\mathbf{y}[k])) p(\check{\mathbf{x}}[k]|\mathbf{x}[k-1]) d\check{\mathbf{x}}[k] \\ &= p(\mathbf{x}[k] + \eta\Phi(\mathbf{y}[k])|\mathbf{x}[k-1]) \\ &= p(\check{\mathbf{x}}[k]|\mathbf{x}[k-1]) \end{aligned} \tag{7.26}$$

where $\Phi(\mathbf{y}[k]) = \frac{\partial(\mathbf{y}[k] - \mathbf{g}_k(\check{\mathbf{x}}[k]))^2}{\partial \check{\mathbf{x}}[k]}$. By using this proposal distribution, the importance weights can be calculated as

$$w^i[k] = w^i[k-1] p(\mathbf{y}[k]|\mathbf{x}^i[k]) \frac{p(\mathbf{x}[k]|\mathbf{x}^i[k-1])}{p(\check{\mathbf{x}}[k]|\mathbf{x}^i[k-1])}. \tag{7.27}$$

The implementation procedure of the GPF algorithm is summarized in Table 7.3. The additional computational complexity to implement the gradient MOVE-step is only $\mathcal{O}(NN_x)$ operations, where N_x is the dimension of states.

Table 7.3 Gradient proposal particle filtering algorithm.

1. *Initialization*
   At $k = 0$
   • For $i = 1, \ldots, N$,
   - Draw samples (or particles) $\mathbf{x}^i[0] \sim p(\mathbf{x}[0])$.
   - Set $w^i[0] = 1/N$.
2. *Importance Sampling and MOVE Step*
   For $k = 1, 2, \ldots$
   • For $i = 1, \ldots, N$,
   - Sample $\breve{\mathbf{x}}^i[k]$ according to (7.26).
   • For each sample $\{\breve{\mathbf{x}}^i[k]\}$,
   - Obtain the sample $\{\mathbf{x}^i[k]\}$ by running the MOVE-step (7.25) using the measurement $\mathbf{y}[k]$.
   • For $i = 1, \ldots, N$,
   - Evaluate the importance weights $w^i[k]$ according to (7.27).
   • For $i = 1, \ldots, N$,
   - Normalize the importance weights $w^i[k]$ according to (7.11).
   • Evaluate the effective sample size according to (7.15) in order to decide whether the resampling step is performed or not.
3. *Resampling Step*
   • Apply resampling as in the SIR filter.

7.3 System Model

We consider a MIMO wireless link with M_T transmit and M_R receive antennas in a flat-fading environment. We assume that the channel matrix is modeled as Rayleigh random processes and the fading is to be uncorrelated across antennas where each individual realization of the channel path is independent for all time steps k. To define the problem of channel tracking, let us consider

the DSSM for a MIMO communication system

$$\mathbf{h}[k] = \mathbf{f}(\mathbf{h}[k-1]) + \mathbf{d}[k-1] \tag{7.28}$$

$$\mathbf{r}[k] = \mathbf{g}(\mathbf{h}[k], \mathbf{s}[k]) + \mathbf{v}[k] \tag{7.29}$$

which describes the process (or state) and measurement equations, respectively. In the above equations, $\mathbf{h}[k] \in \mathcal{C}^{(M_R M_T \times 1)}$ represents the state of interest, i.e, the channel state vector; $\mathbf{s}[k] \in \mathcal{C}^{M_T \times 1}$ is the known input vector that appears in the measurement equation, i.e. the symbol vector simultaneously transmitted by the M_T transmit antennas; $\mathbf{r}[k] \in \mathcal{C}^{M_R \times 1}$ represents the measurement vector which is the received signal vector; $\mathbf{f}$ and $\mathbf{g}$ are generic vector-valued functions, which are potentially time-varying; $\mathbf{d}[k]$ and $\mathbf{v}[k]$ represent the process and measurement noise processes of variance σ_d^2 and σ_v^2, respectively. With a symbol rate $1/T_s$, the received signal vector at time k can be written as

$$\mathbf{r}[k] = \mathbf{S}[k]\mathbf{h}[k] + \mathbf{v}[k] \tag{7.30}$$

where the $(M_R \times M_T)$ matrix $\mathbf{S}[k]$ is a wide matrix with the transmitted symbols repeated diagonally, according to the Kronecker product

$$\mathbf{S}[k] = [s_1[k] \dots s_{M_T}[k]] \otimes \mathbf{I}_{M_R}. \tag{7.31}$$

7.3.1 Ambient Noise

The ambient noise $\mathbf{v}[k] \in \mathcal{C}^{M_R \times 1}$ can be modeled by circularly symmetric complex Gaussian (or non-Gaussian) random variables. As an ideal noise environment, the noise is modeled as a complex-normal distribution given by

$$p_{\mathbf{v}} = \mathcal{CN}(0, \sigma_v^2) \tag{7.32}$$

with zero mean and variance σ_v^2. For a realistic noise model, the probability density function of a Gaussian mixture model, which serves as an approximation to the Middleton's Class A model [3], is given by

$$p_{\mathbf{v}} = (1-\epsilon)\mathcal{CN}(0, \varsigma^2) + \epsilon\mathcal{CN}(0, \kappa\varsigma^2) \tag{7.33}$$

with $\varsigma > 0$, $0 \leq \epsilon < 1$, and $\kappa > 1$. The $\mathcal{CN}(0, \varsigma^2)$ term represents the nominal background noise, and the $\mathcal{CN}(0, \kappa\varsigma^2)$ term represents the impulsive component with ϵ representing the probability that an impulse occurs. To maintain a fixed total noise variance σ_v^2, we may vary the parameters ϵ and κ such that

$$\sigma_v^2 = (1 - \epsilon)\varsigma^2 + \epsilon\kappa\varsigma^2. \tag{7.34}$$

7.3.2 Channel Modeling

Since the fading channel is a Rayleigh process, the stochastic characteristics of each of the $M_T M_R$ channel coefficients depend on the maximum Doppler shift $f_D = vf_c/c$, where v is the mobile speed, f_c is the carrier frequency and c is the speed of light. When v is constant, each channel coefficient is modeled by the Jakes' model as a wide-sense stationary, circular complex Gaussian process with zero mean and time-autocorrelation properties, governed by the Doppler rate $f_D T_s$ as

$$R_h[\tau] = \mathcal{E}\{h_{ij}[k]h_{ij}^*[k - \tau]\} = J_0(2\pi f_D T_s \tau) \tag{7.35}$$

where h_{ij} is the channel coefficient for the path from the i^{th} transmit antenna to the j^{th} receive antenna and J_0 is the zero-order Bessel function of the first kind. The time evolution of the vector process $\{\mathbf{h}[k]\}$ can be approximated by an autoregressive (AR) process with satisfactory accuracy.

Thus the MIMO channel variation $\mathbf{h}[k]$ is modeled with the AR process of order p

$$\mathbf{h}[k] = \sum_{i=1}^{p} \mathbf{F}(i)\mathbf{h}[k - i] + \mathbf{d}[k - 1] \tag{7.36}$$

where $\mathbf{d}[k]$ is the complex Gaussian noise vector with zero mean and variance σ_d^2 which drives the process. Due to the wide sense stationary uncorrelated scattering (WSSUS) assumption, the matrices $\mathbf{F}(i)$, $i = 1, \ldots, p$ must be diagonal, i.e. $\mathbf{F}(i) = \alpha_i \mathbf{I} \in \mathcal{R}^{(M_R \times M_T)}$.

It has been shown in [37] that a first order AR (AR(1)) model is sufficient to accurately model the local behavior of the time variations of the channel[1]. In this chapter, we use the AR(1) process

[1] A higher order AR model can model more accurately long-term channel variations, but it leads to intractable solutions.

as our channel model

$$\mathbf{h}[k] = \alpha \mathbf{h}[k-1] + \mathbf{d}[k-1] \tag{7.37}$$

where $\alpha = J_0(2\pi f_D T_s) \leq 1$ is the static AR(1) coefficient that may vary between zero and one according to Doppler spread, and it is assumed to be known. The driving noise is $\mathbf{d}[k] \overset{i.i.d.}{\sim} \mathcal{CN}(0, (1-\alpha^2)\sigma_d^2 \mathbf{I})$. When $\mathbf{d}[k]$ is a white Gaussian sequence, the model (7.37) is also referred to as the first-order Gauss-Markov model.

In summary, the state-space model of the (M_T, M_R) MIMO system can be represented by the equations (7.37) and (7.30) which are the process and measurement equation, respectively. Channel tracking will be developed on this state-space model.

7.4 Iterative Channel Tracking with Particle Filtering

As in the previous chapter, we exploit turbo-BLAST [322],[334] over time-varying fading channel environments. In turbo-BLAST, space-time coded signals are sent by the transmitters and the received signals are then iteratively processed by the receiver. Fig. 7.3 illustrates the simplified turbo-BLAST receiver with particle filtering channel estimation. The main components of the receiver consist of a SISO (soft-input soft-output) detector, M_T-parallel SISO decoders, and a channel estimator.

The SISO detector is implemented by a suboptimal multistream detector based on minimum mean-square error (MMSE) and parallel soft-interference cancellation. This MMSE-based SISO detector optimizes jointly the weights of the linear detector and the co-antenna interference estimate. At a given time k, the estimate x_m of a transmitted symbol s_m from the m^{th} transmit antenna is given by [322]

$$x_m = (\mathbf{h}_m^H \mathbf{h}_m + \sigma_v^2)^{-1} \mathbf{h}_m^H (\mathbf{r} - \mathbf{H}_m \mathcal{E}\{\mathbf{s}_m\}) \tag{7.38}$$

where $\mathbf{h}_m$ is the m^{th} column of the channel matrix $\mathbf{H}_k$, $\mathbf{H}_m$ is the $M_R \times (M_T - 1)$ matrix where the m^{th} column is removed from $\mathbf{H}_k$, and $\mathcal{E}\{\mathbf{s}_m\}$ where the m^{th} entry is removed is the $(M_T - 1) \times 1$ estimated symbol vector provided by the SISO decoders from the previous iteration.

The output of the SISO detector is deinterleaved (Π^{-1}) and is fed to the M_T-parallel SISO decoders. The SISO decoders perform symbol-by-symbol log-MAP decoding and produce the

refined soft information of the transmitted data. The soft information passed through the interleaver (Π) is fed back to the SISO detector for the next iteration. The estimated symbol matrix $\widehat{\mathbf{S}}$ from the soft output of the SISO decoders is used to refine the channel estimate in the channel estimator.

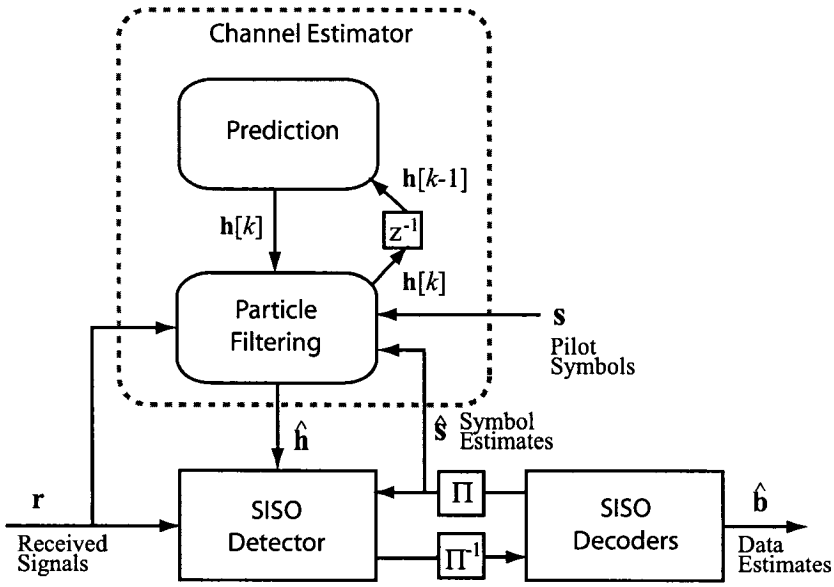


Fig. 7.3 Turbo-BLAST receiver with channel estimator.

7.4.1 Pilot Symbol Placement

A channel estimation task is usually based on pilot (or training) symbols known to the receiver. Pilot-assisted transmission (PAT) multiplexes pilot symbols with information bearing data. This pilot symbol assisted modulation (PSAM) [339] is now a widely-used term. There are two types of training for single carrier systems: time division multiplexing (TDM) training and superimposed training [137]. Pilot symbols in a TDM system are inserted into the data stream according to a certain placement pattern, and the channel estimate is updated using the pilot symbols. On the other hand, in superimposed training, pilot and data symbols are added and transmitted together, and the channel estimate is updated at each symbol. Pilot symbol design affects the system performance for time-varying channels. Intuitively, superimposed training may have the advantage when the channel fades rapidly, but the superimposed data interferes with pilot-assisted channel estimation, which may lead to an undesirable performance floor in the high SNR regime [340]. Therefore,

TDM training herein is considered.

For the TDM methods, regular periodic placement (RPP) schemes are commonly used. In general, any pilot placement with n clusters of pilot symbols in a period of frame length T can be specified by a 2-tuple $\mathcal{P} = (\mathbf{p}, \mathbf{d})$, where $\mathbf{p} = [p_1, \ldots, p_n]$ is the pilot cluster length vector and $\mathbf{d} = [d_1, \ldots, d_n]$ is the data block length vector, as illustrated in Fig. 7.4. Note that T_1 is the period of one placement and $T = \sum_{i=1}^{n}(p_i + d_i)$. The RPP scheme with the length of pilot clusters p is denoted as RPP-p. For time-varying channels, the single pilot RPP scheme (referred to as the RPP-1 scheme) was first analyzed in [339], and applied in various setting [341],[342]. In [341], for flat Rayleigh fading modeled by a Gauss-Markov process and the PSAM scheme, the optimal spacing between the pilot symbols is determined numerically by maximizing the mutual information with binary inputs. In [342], for the flat fading channel modeled by a bandlimited process, under some assumptions on the channel and data symbols, the optimal parameters for pilots, including pilot symbol spacing and power allocation, are determined by maximizing a lower bound on capacity. Recently, it has been shown that among all periodic placements, the RPP-1 scheme minimizes the maximum steady-state channel MMSE and uncoded BER for both BPSK and QPSK signaling, regardless of the SNR level or the rate of channel variation [340].

For (M_T, M_R) MIMO channels, the system inputs correspond to frames from M_T transmitters, and the outputs come from M_R diversity channels. Allowing pilot symbols to be placed independently for each transmit antenna and assuming that the number of pilot clusters for the mth antenna is n_m, the placement $\mathcal{P} = (\mathbf{p}, \mathbf{d})$ is defined by $\mathbf{p} = [\mathbf{p}^1, \ldots, \mathbf{p}^{M_T}]$ and $\mathbf{d} = [\mathbf{d}^1, \ldots, \mathbf{d}^{M_T}]$, where $(\mathbf{p}^m, \mathbf{d}^m)$ is the placement for the mth transmitter. In [343], for MIMO-multiuser systems, the Cramér-Rao bound (CRB) on the MSE of semiblind channel estimators was derived and minimized with respect to pilot symbols and their placement. The optimal designs and placements in MIMO channels was also described in [343], where the optimal placement involves the orthogonality design of pilot symbols among users.

In this chapter, we employ the RPP-1 scheme for tracking time-selective MIMO channels. However, we do not try any optimal designs and placements for pilot symbols.

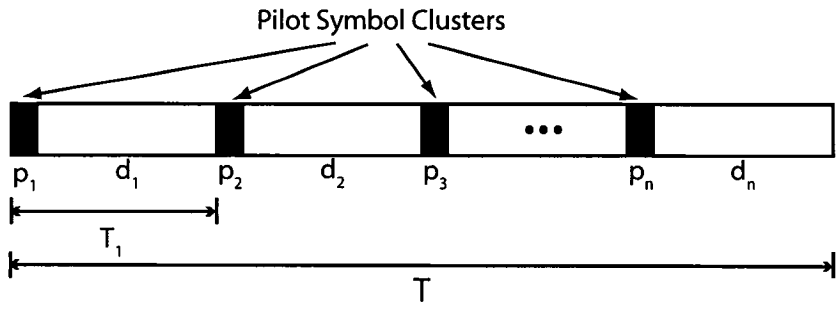


Fig. 7.4 Data stream with periodic pilot placements.

7.4.2 Particle Filter Channel Tracking Algorithm

Given the state-space model of a MIMO system in (7.37) and (7.30), the objective of channel tracking is to recursively estimate the state $\mathbf{h}[k]$ from the measurement $\mathbf{r}[k]$. We design particle filtering channel estimators to reduce the uncertainty in the channel estimate and ultimately to reduce the BER performance of the turbo-BLAST receiver. When pilot symbols are periodically inserted in each frame, the use of semiblind estimation techniques are of great interest. In the following, semiblind channel tracking is considered, where for every transmitted frame (or packet), the procedure entails training during the pilot clusters followed by tracking during the subsequent set of data blocks within the frame. In semiblind channel tracking, periodic pilot symbols are helpful to prevent the divergence of the channel estimator.

As shown in Fig. 7.3, the channel estimator cooperates with the iterative MIMO receiver and is iteratively processed. At the first iteration of the receiver processing, the channel state is estimated from the pilot symbols by a conventional estimation technique such as least-squares (LS) or MMSE estimation. Based on the channel estimate from the pilots, the SISO detector and SISO decoders are performed to produce the estimated data symbols. At the second iteration, the channel estimator uses both the pilot symbols and the estimated data symbols to attain a refined channel estimate. The refined channel estimate is used for the SISO decoder. This procedure is repeated until the receiver converges or a stopping criterion is met.

The channel estimator in Fig. 7.3 has the prediction and filtering parts that form a channel tracking procedure using particle filtering. During the training mode, the channel estimate is calculated from the pilot symbol. This estimate is then used to guide the particles. During the

decision-directed mode, the particle filter uses the estimated data symbols from the SISO decoders in order to produce a channel estimate. Because the channel tracker operates in a semiblind mode, the tracking algorithm is expected to accurately estimate the channel even when no pilot symbols are presented. The channel tracking algorithm using GPF can be summarized as follows:

1. **Initialization**

   At $k = 0$

   - For $i = 1, 2, \ldots, N$, sample particles $\mathbf{h}[0]^i$ from the prior $p(\mathbf{h}[0])$, and $w_0^i = 1/N$.

2. **Channel Tracking**

   For $k = 1, 2, \ldots$

   Step 1: Initial channel estimation and importance sampling. (Do the following steps, if pilot symbols are detected.)

   - LS channel estimation:

$$\mathbf{h}[k] = (\mathbf{s}[k]^H \mathbf{s}[k])^{-1} \mathbf{s}^H[k]\mathbf{r}[k] \tag{7.39}$$

     where $\mathbf{s}[k]$ is the pilot symbol vector at time k.

   - Importance sampling: For $i = 1, \ldots, N$, sample

$$\tilde{\mathbf{h}}^i[k] \sim p(\mathbf{h}[k]|\mathbf{h}^i[k-1]). \tag{7.40}$$

   Step 2: Prediction of the current channel state using (7.37).

   Step 3: Filtering of the current channel state.

   - Gradient MOVE-step: For each particle $\{\tilde{\mathbf{h}}^i[k]\}$, do the gradient MOVE-step via (7.25).
   - Importance weights update: Evaluate the importance weights

$$w^i[k] = w^i[k-1] \frac{p(\mathbf{r}[k]|\mathbf{h}^i[k]) p(\mathbf{h}^i[k]|\mathbf{h}^i[k-1])}{p(\tilde{\mathbf{h}}^i[k]|\mathbf{h}^i[k-1])}, \tag{7.41}$$

and normalize the importance weights via (7.11).

- Resampling: If $\hat{N}_{\text{eff}} < N_{\text{th}}$, sample N new particles $\mathbf{h}^i[k]$ from the set $\{\tilde{\mathbf{h}}^i[k]\}$ according to the normalized importance weights $\tilde{w}^i[k]$.
- Computation of the channel estimate

$$\hat{\mathbf{h}}[k] = \mathcal{E}\{\mathbf{h}[k]|\mathbf{r}[k]\} \approx \frac{1}{N}\sum_{i=1}^{N}\mathbf{h}^i[k]. \tag{7.42}$$

- Propagation of the filtered particles to the "prediction" part for time $k+1$.

Step 4: Set $k \rightarrow k+1$ and go to Step 1.

The channel tracking algorithm using the SIR filter also follows this procedure, except for the gradient MOVE-step in Step 3.

7.5 Simulation Results

Simulation experiments are performed for a MIMO link with $M_T = 2$ transmit and $M_R = 2$ receive antennas. At the transmitter, each substream of 1000 information bits is independently encoded with a rate-1/2 convolutional code generator $(7,5)_{\text{Oct}}$ and is sent with BPSK modulation. Rayleigh fading processes among all transmit and receive antennas are assumed i.i.d. and their first- and second-order statistics do not change over the entire transmission horizon. For simulations, the channel is simulated using an improved version of Jakes' fading model [344]. The performance of particle filter channel tracking algorithms is evaluated in terms of BER and channel estimation errors, defined as [334],[216]

$$\text{ANSE} = \mathcal{E}\left[\frac{\|\mathbf{h}-\hat{\mathbf{h}}\|^2}{\|\mathbf{h}\|^2}\right] \tag{7.43}$$

which is called the average normalized square error (ANSE). All results are ensemble-averaged from independent transmissions of 200 frames.

In the following, two different types of measurement noise are employed in an attempt to model the noise statistics of the urban wireless channel: the widely accepted Gaussian distribution (7.32) and the Middleton's Class A model (7.33), which is a non-Gaussian distribution.

7.5.1 Gaussian Noise Case

The time-selective frequency-flat Rayleigh fading channel has the normalized Doppler rate of $f_D T_s = 0.01$, considered here to be a fast fading channel (from [166]). Returning to (7.37) this implies $\alpha = 0.999$. In the simulations, a single pilot symbol per transmit antenna is periodically inserted at every 20 symbols, i.e. the RPP-1 with $T_1 = 20$, in order to estimate the channel and to prevent the divergence of the tracking algorithm. The performance of systems employing particle filtering for channel tracking is compared over that with perfect channel knowledge and without tracking (static case). For the static case, the channel was estimated using pilots only by LS estimation at the first iteration, and this estimated channel was used in all the following iterations. In the cases of channel tracking with two particle filters (SIR and GPF), the channel was tracked by using both the pilots and the estimated data symbols at the previous iteration. It is noted that the number of particles to use is a balance between choosing enough particles to reliably sample the state space, and using too many which will not increase performance but increase complexity. We use SIR and GPF with 200 and 50 particles, respectively, unless otherwise stated.

Fig. 7.5 shows the bit error rate (BER) performance for the tracking algorithms after the first and fifth iteration. The performance with the perfect channel knowledge can be used as a reference. At the fifth iteration the BER performance of the GPF tracking outperforms that of the static case and the SIR tracking. Among the three estimation techniques, the the static case with pilots gave the worst performance because the channel corresponds to a highly time-selective environment. This means that the channel varies from symbol to symbol within a frame. With much less particles, the GPF tracking maintained a gain of around 3 dB at the BER of 10^{-3} over the SIR tracking, while it performed approximately 4 dB worse than the case when the channel coefficients are perfectly known to the receiver. The quality of the channel estimate can be evaluated by the ANSE. Fig. 7.6 shows how the ANSE with the SIR and GPF channel tracking can attain performance gains as the iterations proceed. We observe that as the iterations increase, the accuracy of the channel estimate in ANSE values is improved due to the use of the estimated data symbols. It is the benefit of the iterative MIMO receiver employed, where the SISO decoders provide a refined symbol estimate as the iterations continue. We can see from Fig. 7.6 that compared to the SIR tracking, the improvement by the GPF tracking is outstanding. The ANSE by the GPF tracking

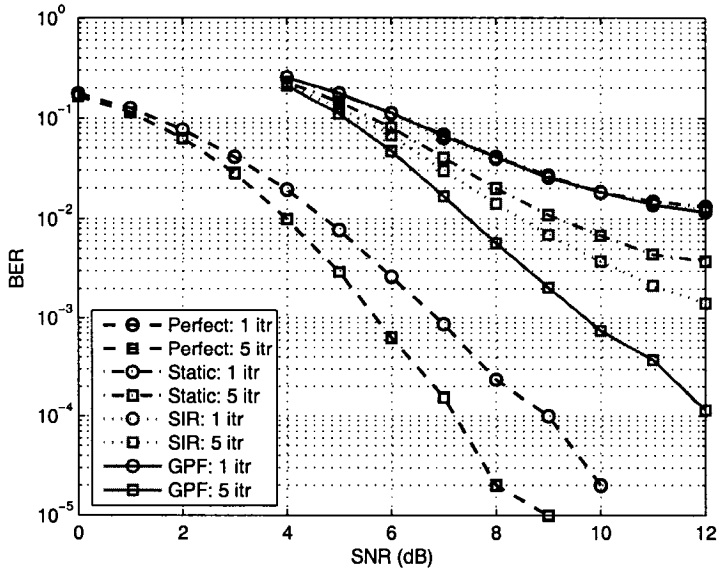


Fig. 7.5 BER performance in Gaussian noise.

drops continuously as the iterations proceed, whereas the SIR tracking does not improve the ANSE value after the third iteration. For instance, the GPF tracking earns an ANSE of 0.0615 at the SNR of 12 dB after the fifth iteration, while the static case and the SIR tracking earn 0.2541 and 0.2077, respectively.

The BER performance is directly affected by the ANSE of the channel estimation, as can be seen from Fig. 7.7. The performance with the SIR tracking approaches a BER of slightly above 10^{-3} at the SNR of 12 dB and has no more improvement after the third iteration. On the other hand, the BER performance with the GPF tracking reaches almost 10^{-4} at the SNR of 12 dB and is expected to improve with further iterations. In Fig. 7.8, we show the migration of a channel coefficient from the first transmitter to the first receiver through time, and the ability of the tracking algorithms to follow the movement of the channel coefficients. The solid lines in Fig. 7.8 depict the true channel coefficients at the SNR of 10 dB, while the dash-dotted lines are the estimated coefficient. The SIR (Fig. 7.8(a)) and the GPF (Fig. 7.8(b)) tracking are able to keep estimates close to the true values. We see that the GPF tracking offers a better channel estimate than the SIR tracking.

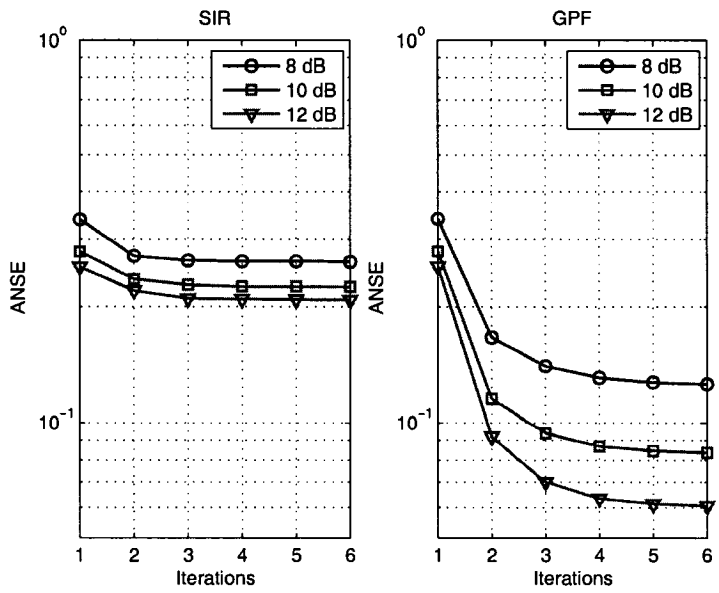
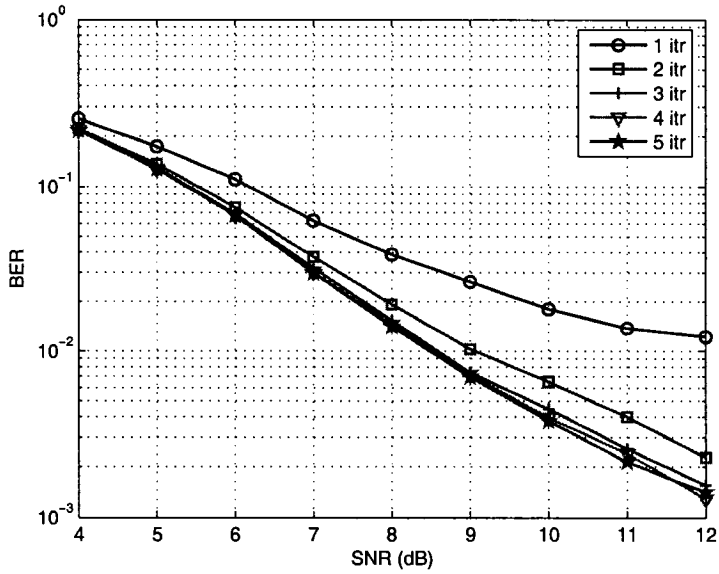


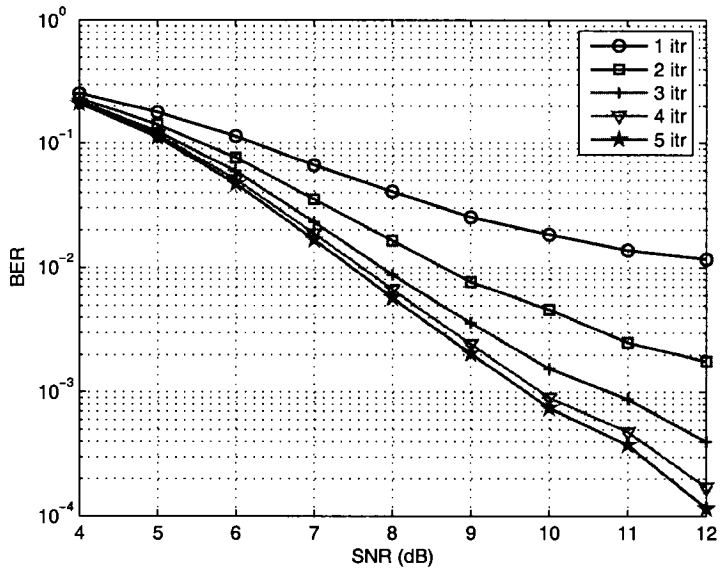
Fig. 7.6 Channel estimation errors.

In Fig. 7.9, we show the BER performance of the three receivers at the SNR of 10 dB as the period of pilot placement (or frequency) changes from $T_1 = 5$ to 30. We see that for this fast fading environment, the more frequently pilot symbols are inserted, the better the estimation and tracking are, and the more roust the receiver is, at the expense of data rates. When $T_1 \geq 15$, the performance with the GPF tracking is superior to those of both no tracking (static case) and SIR tracking. However, frequent pilot insertions ($T_1 < 10$) in the GPF tracking do not guarantee better BER performance, compared to the other two methods.

When the normalized Doppler rate is reduced to $f_D T_s = 0.001$, which corresponds to $\alpha = 1.0$ and results in a slow fading channel, the BER performance at $T_1 = 50$ is compared in Fig. 7.10. At high SNR, the BER performance by the SIR tracking is close to the optimal performance with the perfectly known channel. Interestingly, among the three methods the receiver with the GPF tracking shows the worst performance at SNR higher than 7 dB. It is shown that the channel remains approximately constant from symbol to symbol, the GPF tracking may be unsuited to track a slow fading channel.

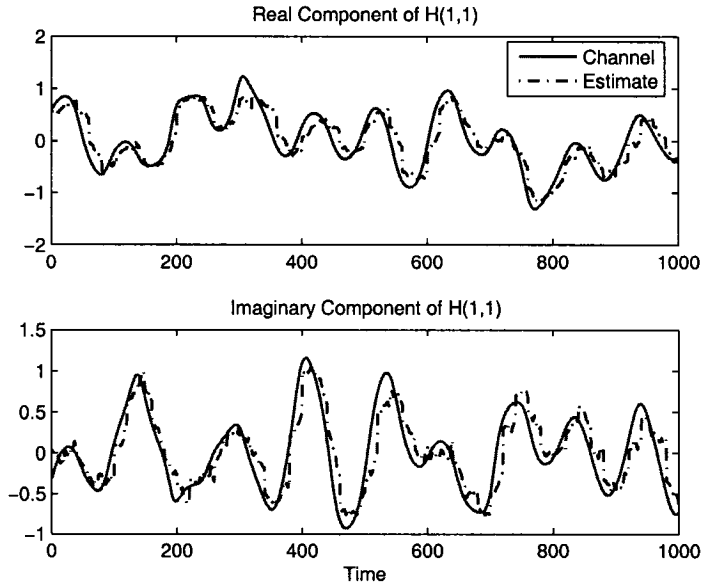


(a) SIR ($N = 200$).

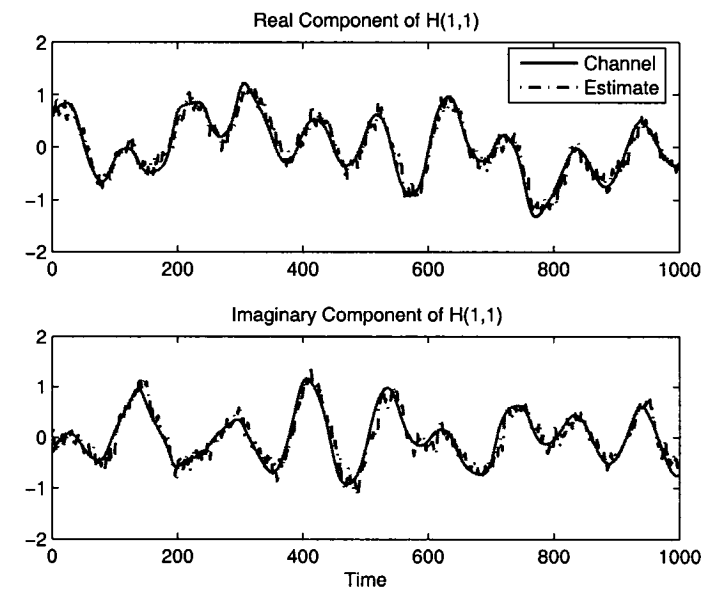


(b) GPF ($N = 50$).

Fig. 7.7 BER performances at $f_D T_s = 0.01$.



(a) SIR ($N = 200$).



(b) GPF ($N = 50$).

Fig. 7.8 Channel tracking performance of the particle filters when SNR = 10 dB and $f_D T_s = 0.01$.

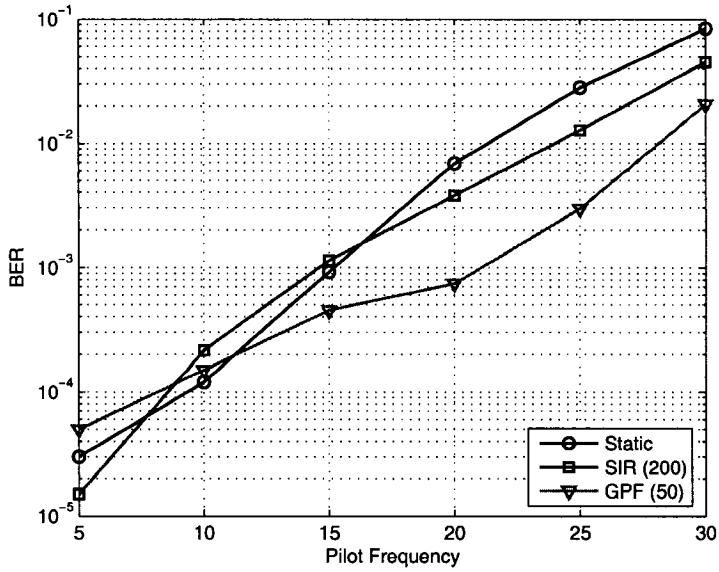


Fig. 7.9 BER performance vs. pilot frequency when SNR = 10 dB and $f_D T_s = 0.01$.

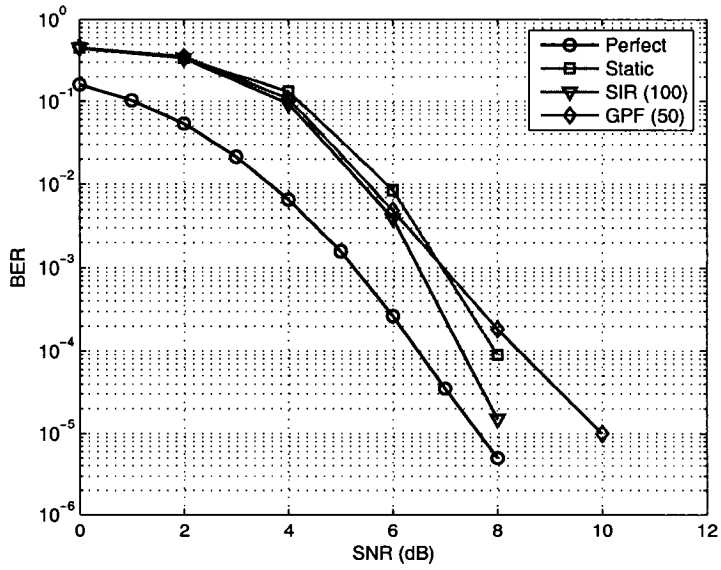


Fig. 7.10 BER performance when $f_D T_s = 0.001$ (slow fading).

7.5.2 Non-Gaussian Noise Case

In the following simulations, non-Gaussian noise is modeled by the Middleton's Class A model (7.33), with parameters $\epsilon = 0.1$ and $\kappa = 100$. We use the normalized Doppler rate of $f_D T_s = 0.01$. The number of particles for SIR and GPF are set to 200 and 100, respectively. Fig. 7.11 shows the BER performance for the tracking algorithms after the fifth iteration. Due to this high time-selectivity, it is apparent that overall BER performances of the three receivers with channel estimation are poor. However, at high SNR regime (larger than 20 dB) we observe a difference in BER performance between the GPF tracking receiver and the other two. Both the static receiver and the SIR tracking receiver approach an error floor at an SNR of 20 dB and 28 dB, respectively, whereas the GPF tracking receiver still improves the BER performance without an error floor. At a BER of 10^{-4} the GPF tracking receiver performs approximately 11 dB worse than the receiver with the perfectly known channel because of the effect of non-Gaussian noise.

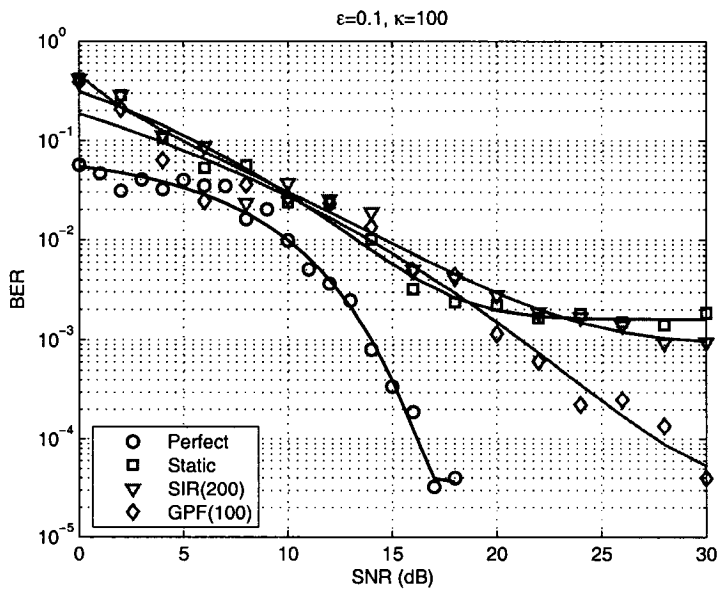


Fig. 7.11 BER performance in non-Gaussian noise ($\epsilon = 0.1$ and $\kappa = 100$) after the fifth iteration.

Just like in the case of Gaussian noise, the performance of the receivers is dependent on the quality of the channel estimate. Figs. 7.12 and 7.13 show how the ANSE and BER of the receivers change as the iterations proceed. The ANSE of the channel trackers at an SNR of 26 dB is depicted

in Fig. 7.12. As we have seen in the figure, the SIR tracker reaches a steady-state ANSE after the third iteration. On the other hand, the GPF tracker still reduces the ANSE value. Returning to Fig. 7.13, we observe that the channel tracking performance affects the BER performance of the receivers. After the third iteration, the SIR tracking receiver has no more improvement in BER performance and, in turn, reaches an asymptotic error floor at high SNR. However, the GPF tracking receiver attains a gradual BER improvement as it is iteratively performed, albeit it encounters an asymptotic error floor until the fourth iteration. Channel tracking performance shown in Fig. 7.14 indicates the difference between the two channel trackers. With half of the number of particles, the GPF channel tracker outperforms the SIR channel tracker for a fast fading channel.

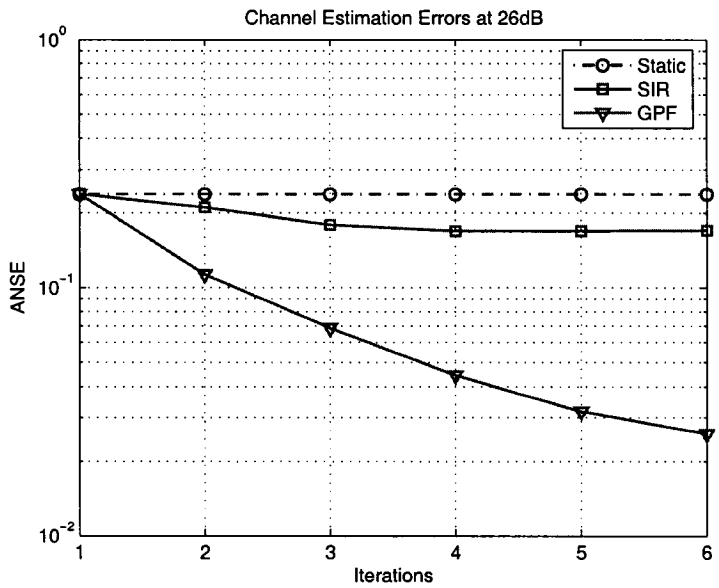
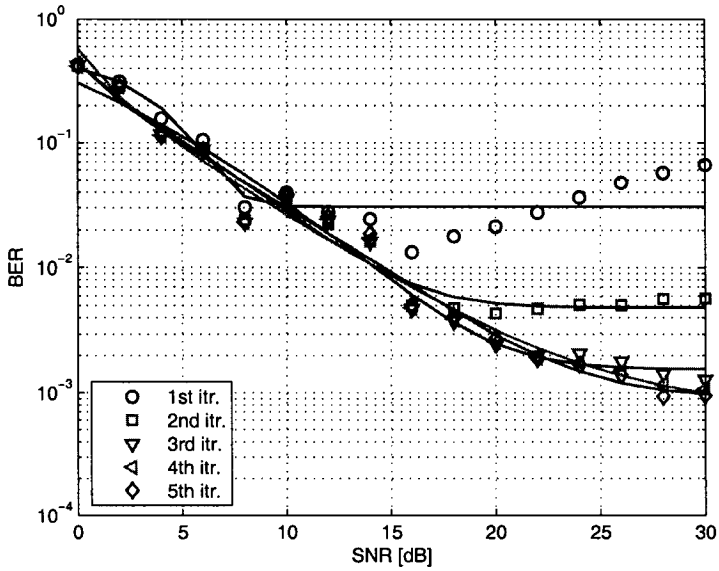
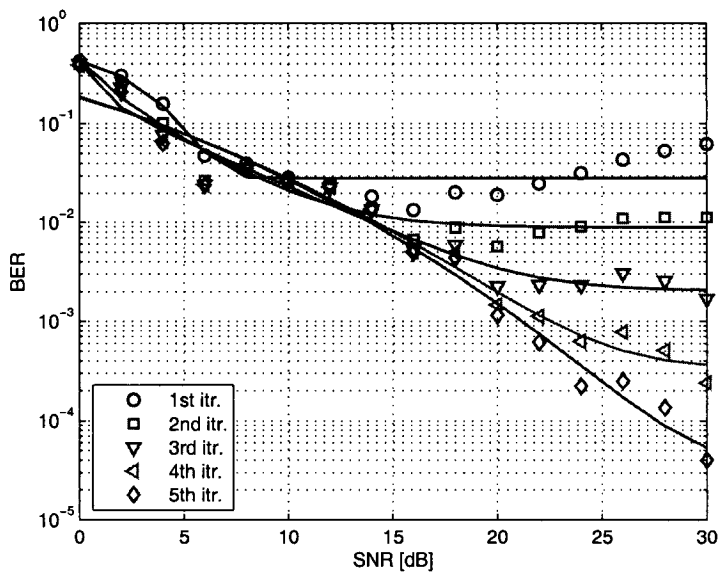


Fig. 7.12 Channel estimation errors.

For a slow fading channel where $f_D T_s = 0.001$, the BER performance at $T_1 = 50$ is compared in Fig. 7.15. Interestingly, the BER performance of both the static receiver and the SIR tracking receiver is similar to that of the receiver with the perfectly known channel, at SNR larger than 10 dB. However, the GPF tracking receiver is far away from the other receivers. As in the Gaussian noise, the GPF channel tracker may be not suitable for a slow fading environment over non-Gaussian noise.

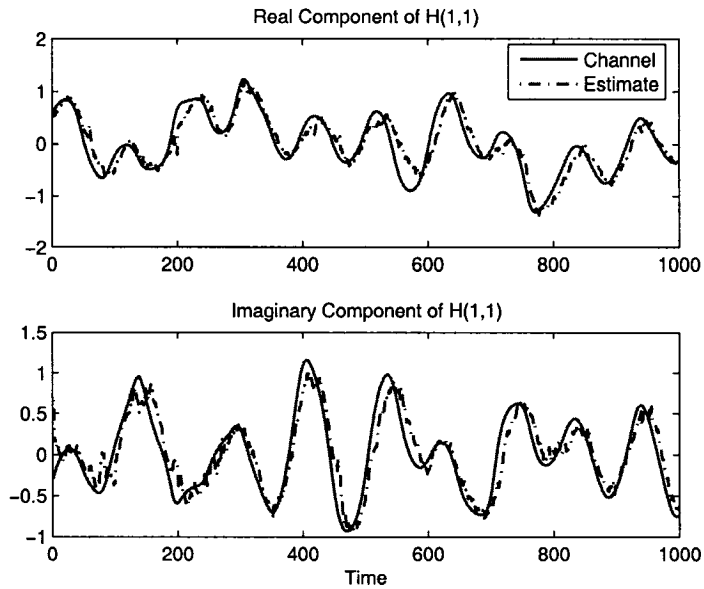


(a) SIR (200).

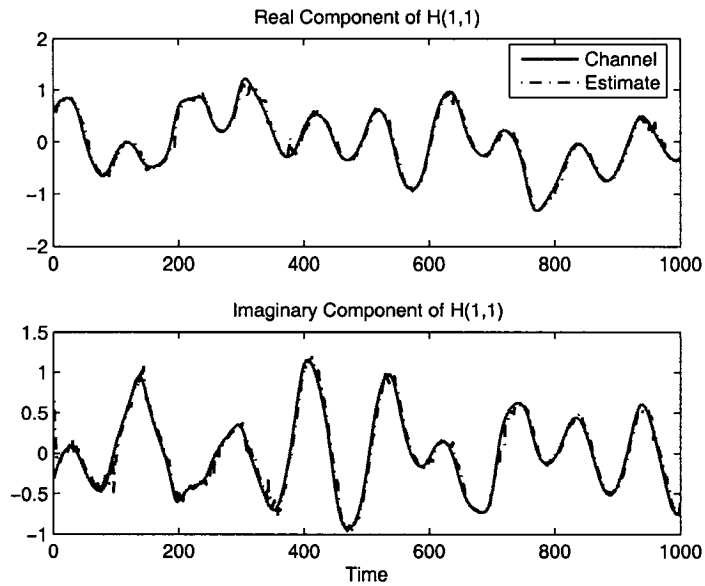


(b) GPF (100).

Fig. 7.13 Comparison of BER performances when $f_D T_s = 0.01$ ($\epsilon = 0.1$ and $\kappa = 100$).



(a) SIR ($N = 200$).



(b) GPF ($N = 100$).

Fig. 7.14 Channel tracking performance of the SIR and the GPF tracker at SNR = 26 dB.

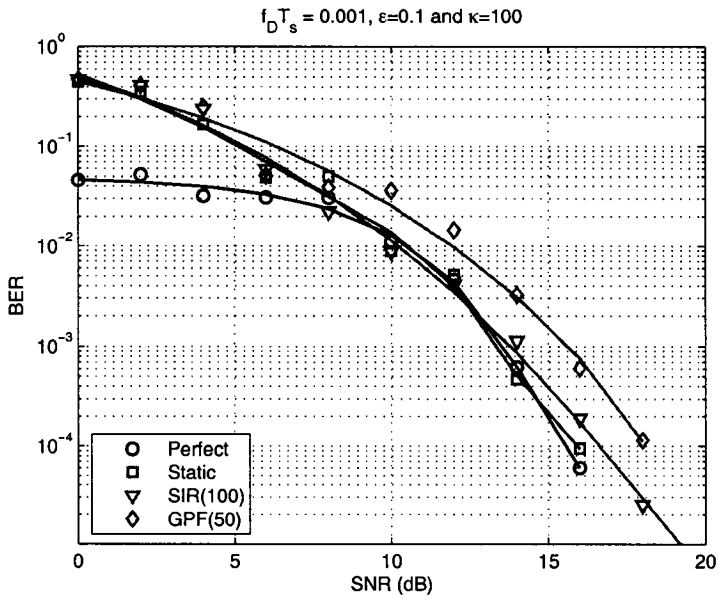


Fig. 7.15 BER performance at $f_D T_s = 0.001$ (slow fading) over non-Gaussian noise.

7.6 Concluding Remarks

This chapter has proposed the use of particle filtering to obtain a reliable channel estimate for time-selective, frequency-flat, Rayleigh fading MIMO channels. The performance of the turbo-BLAST receiver with particle filtering channel tracking has been demonstrated for both fast and slow fading channels, which are modeled by the first-order AR model. Under fast time-selective channels, the inclusion of two particle filtering channel trackers, the SIR and GPF tracker, has improved the receiver performance in terms of BER and ANSE, compared to the static receiver with no channel tracking. We have shown the feasibility of the GPF tracking receiver for fast fading channels in both Gaussian and non-Gaussian noise environments. In addition, the use of the GPF channel tracking has shown to overcome the error floor phenomenon over non-Gaussian noise.

In contrast, for a slow fading channel the SIR tracking has offered an improved BER performance compared to the GPF tracking. It has been found that the GPF tracking may not be suitable for slow fading MIMO channels over both Gaussian and non-Gaussian noise.

Chapter 8

Conclusions

This chapter summarizes the results of this thesis and outlines some directions for further research.

8.1 Summary of Results

In this thesis, we presented signal processing algorithms based on a recursive filtering framework for equalization and estimation of communication channels. This thesis consisted of three major parts. The first part was composed of two chapters (Chapter 2 and Chapter 3), providing a foundation of recursive Bayesian filtering in Chapter 2, and the literature related to channel equalization and estimation as well as non-Gaussian noise in Chapter 3. The main contributions of the thesis were the next two parts. The second part (Chapter 4 and Chapter 5) of the thesis contributed to adaptive nonlinear channel equalization for single-transmit single-receive communication links. In the third part (Chapter 6 and Chapter 7), we investigated iterative channel estimation techniques combined with an iterative space-time receiver for frequency-flat Rayleigh fading MIMO channels.

The performance of equalization over dispersive communication channels is subject to distortion and noise. The equalizers trade off performance, data rates and computational complexity to combat the distortion and noise. Traditional linear equalization schemes are simple and computationally efficient to implement, but can hardly compensate for a nonlinear distortion or non-Gaussian impulsive noise. The EKF algorithms (GEKF and DEKF), developed in Chapter 4, for a complex-valued adaptive recurrent neural equalizer, achieved fast convergence rates in the dispersive channels with a nonlinear distortion and a Gaussian noise environment. This property leads to high data rate transmission since the EKF algorithms require only a small number of training sym-

bols at the cost of some computational complexity. In addition to the convergence rates, they have attained better performance gains in bit error probability than the conventional RTRL algorithm. We showed that the DEKF requires less computational complexity and storage requirements in implementation than the GEKF, without any performance degradation. The adaptive equalizer trained with the EKF algorithms is helpful to combat intersymbol interference and nonlinear distortions in the Gaussian noise environment.

For non-Gaussian channels where impulsive noise is modeled as a symmetric α-stable random process, an adaptive nonlinear equalizer using a recurrent multilayer perceptron and its training algorithm using the UKF algorithm was presented in Chapter 5. The results indicated that in the non-Gaussian noise case the UKF algorithm converges faster than the EKF for both linear or nonlinear distortions and time-invariant or time-varying channels. The UKF, therefore, attained not only better BER performance but also led to the reduction of the number of training symbols. On the other hand, the normalized least mean p-norm (NLMP) algorithm, which is known as a computationally efficient algorithm and is suitable to impulsive noise, showed good performance for the linear time-invariant channels in impulsive noise. However, it required a long training sequence. We recognized that compared to the UKF and the EKF, the NLMP algorithm trades off performance, data rates and complexity in some channel conditions, although it may not suit the time-varying channel with nonlinear distortion.

MIMO wireless channels with spatial multiplexing offer a tremendous capacity improvement which increases linearly with $min(M_T, M_R)$ for no additional bandwidth cost or power (only the transceiver complexity is increased). This spatial multiplexing gain is realized by the transceiver where the individual antennas transmit independent data signals and the receiver separates the different streams under conducive channel conditions, such as rich scattering. An iterative space-time MIMO transceiver, referred to as the turbo-BLAST which is a practical solution to high data-rate transmission for wireless communications, requires accurate channel state information at the receiver to fully exploit the advantages of MIMO channels. Iterative channel estimation techniques to be used with the turbo-BLAST over quasi-static Rayleigh fading channels were explored in Chapter 6. The iterative strategy with adaptive filtering is a computationally efficient solution to channel estimation. The strategy is that initial channel estimation is performed with a snapshot method (LS or MMSE) with the training symbols and for further iterations an adaptive

filter (LMS, RLS or KF) is employed to achieve an improvement of the channel estimate with the entire data block. The iterative channel estimation techniques using both the training symbols and the data symbol estimates provided by the decoder achieved significant improvement of the channel estimate, allowing the use of shorter training sequences. Particularly, we found that the tentative hard decisions from the decoder offer more reliable information in iterative channel estimation than the soft decisions.

In Chapter 7, we provided convincing evidence for the benefit of incorporating particle filtering techniques for use in tracking fast fading MIMO wireless channels. The GPF channel tracking obtained a reliable channel estimate and resulted in an improved BER performance over both Gaussian and non-Gaussian noise environments. Moreover, the turbo-BLAST receiver with the GPF tracking can overcome the error floor phenomenon for non-Gaussian noise, whereas both the static receiver and the SIR tracking receiver can not. By using particle filtering for channel tracking, non-Gaussian noise can also be taken into account without any added structural changes or computational complexity.

8.2 Further Research

There are a variety of fruitful areas for further research on channel equalization and estimation. Some suggestions concerning possible extensions of the work presented in this thesis are as follows.

Adaptive Recurrent Neural Equalization:

- Recurrent Neural Architecture for Equalization: In Chapters 4 and 5, two architectures of recurrent neural networks were used for decision feedback equalization. We may have a question about what types of network architecture is optimal to construct a recurrent neural equalizer in a given transmission environment, including the channel type (minimum or nonminimum phase), distortion (linear or nonlinear), noise (Gaussian or non-Gaussian), and so on. Further study on constructing or customizing the architecture of recurrent neural equalizers can serve as a blueprint for further design, analysis and application.

- Scaling Parameters for the UKF: The use of the UKF algorithm was explored in Chapter

5. Notwithstanding the advantages of the UKF over the EKF, a confusing aspect of implementing the UKF is uncertainty in the setting of the scaling parameters (θ, β, and κ), which affects the performance and convergence of the UKF. A further study on the optimal selection is in demand. The values of the parameters could be dependent on data and be tuned online considering the severeness of nonlinearities, variances of noise, and so on.

MIMO Channel Estimation:

- Analysis of Channel Estimation Error: In an iterative MIMO channel estimation two types of information fed back from the SISO decoders are available: *hard* and *soft* decision feedback information. In Chapter 6, the experimental results showed that the hard decision feedback is more reliable in terms of channel estimation accuracy than the soft decision feedback in the iterative estimation process, when a whole data set is used to channel estimation without using any thresholding technique. Further theoretical analysis of channel estimation error using the two information will be required to support the results.

- Channel Estimation for Frequency-Selective MIMO Channels: In the third part of the thesis our channel estimation strategy is restricted to only frequency-flat MIMO channels. When the time variation of frequency-selective MIMO channels within a packet is significant, channel tracking is needed for equalization. In addition to this, an iterative MIMO receiver requires turbo equalization to cancel out interferences. The particle filtering combined with turbo equalization in the design of iterative MIMO receivers will be an interesting topic in order to overcome the fast time variations of the channel and non-Gaussian noise.

References

[1] K. Abend and B. D. Fritchman, "Statistical detection for communication channels with intersymbol interference," *Proceedings of The IEEE*, vol. 58, pp. 779–785, May 1970.

[2] J. G. D. Forney, "Maximum likelihood sequence estimation of digital sequences in the presence of intersymbol interference," *IEEE Transactions on Information Theory*, vol. IT-18, pp. 636–678, May 1972.

[3] D. Middleton, "Non-Gaussian noise models in signal processing for telecommunications: New methods and results for Class A and Class B noise models," *IEEE Transactions on Information Theory*, vol. 45, pp. 1129–1149, May 1999.

[4] E. Telatar, "Capacity of multi-antenna Gaussian channels," *AT &T Bell Laboratories, Technical Memorandum*, October 1995 (Published in *European Transactions on Telecommunications*, Vol. 10, No. 6, pp. 585-595, Nov/Dec 1999).

[5] G. Foschini, "Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas," *Bell Labs Technical Journal*, pp. 41–59, Autumn 1996.

[6] G. Foschini and M. Gans, "On limits of wireless communications in a fading environment when using multiple antennas," *Wireless Personal Communications*, vol. 6, pp. 311–335, 1999.

[7] H. Bölcskei, D. Gesbert, and A. J. Paulraj, "On the capacity of OFDM-based spatial multiplexing systems," *IEEE Transactions on Communications*, vol. 50, pp. 225–234, Feb. 2002.

[8] D. Gesbert, M. Shafi, D. Shiu, P. Smith, and A. Naguib, "From theory to practice: An overview of MIMO space-time coded wireless systems," *IEEE Journal on Selected Areas in Communications*, vol. 21, pp. 281–302, April 2003.

[9] G. Golden, G. Foschini, R. Valenzuela, and P. Wolniansky, "Detection algorithm and initial laboratory results using V-BLAST space-time communication architecture," *Electronics Letters*, vol. 35, pp. 14–16, January 1999.

[10] V. Tarokh, N. Seshadri, and A. R. Calderbank, "Space-time codes for high data rate wireless communication: Performance criterion and code construction," *IEEE Transactions on Information Theory*, vol. 44, pp. 744–756, March 1998.

[11] S. Alamouti, "Space block coding: A simple transmitter diversity technique for wireless communications," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 1451–1458, october 1998.

[12] S. Haykin, *Adaptive Filter Theory, 4th Ed.* Upper Saddle River, NJ: Prentice Hall, 2002.

[13] S. J. Julier and J. K. Uhlmann, "A new extension of the Kalman filter to nonlinear systems," in *Proceedings of AeroSence: The 11th International Symposium on Aerospace/Defence Sensing, Simulation and Controls*, 1997 (Available at http://citeseer.ist.psu.edu/julier97new.html).

[14] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Transactions on Automatic Control*, vol. 45, pp. 477–482, March 2000.

[15] E. A. Wan and R. van der Merwe, *"The unscented Kalman filter," in Kalman Filtering and Neural Networks, Edited by S. Haykin.* John Wiley and Sons, Inc., 2001.

[16] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice.* New York: Springer-Verlag, 2001.

[17] S. Haykin, K. Huber, and Z. Chen, "Bayesian sequential state estimation for MIMO wireless communications," *Proceedings of The IEEE*, vol. 92, pp. 439–454, March 2004.

[18] H. Sorenson, *"Recursive Estimation for Nonlinear Dynamic Systems," in Bayesian Analysis of Time Series and Dynamic Models, Edited by J. C. Spall.* Dekker, 1988.

[19] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[20] M. Nørgaard, N. K. Poulsen, and O. Ravn, "New development in state estimation for nonlinear systems," *Automatica*, vol. 36, pp. 1627–1638, 2000.

[21] T. Lefebvre, H. Bruyninckx, and J. De Schutter, "Comment on "a new method for the nonlinear transformation of means and covariances in filters and estimators"," *IEEE Transactions on Automatic Control*, vol. 47, pp. 1406–1408, August 2002.

[22] E. A. Wan and R. van der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications and Control Symposium (AS-SPCC)*, pp. 153–158, Lake Louise, Alberta, Canada, October 2000.

[23] N. Gordon, *Bayesian Methods for Tracking*. PhD thesis, University of London, U.K., 1993.

[24] G. Kitagawa, "A Monte Carlo filtering and smoothing method for non-gaussian nonlinear state space models.," in *Proceedings of the 2nd US-Japan Joint Seminar on Statistical Time Series Analysis*, pp. 110–131, Honolulu, Hawaii, USA, 1993.

[25] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, pp. 197–208, 2000.

[26] A. Kong, J. Liu, and W. Wong, "Sequential imputations and Bayesian missing data problems," *Journal of the American Statistical Association*, vol. 89, pp. 278–288, 1994.

[27] A. Doucet, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Technical Report, Univ. Cambridge, UK*, 1998.

[28] N. J. Gordon, D. J. Gordon, and A. F. M. Smith, "Novel approache to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceedings - F*, vol. 140, no. 2, pp. 107–113, 1993.

[29] M. K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, pp. 174–188, June 1999.

[30] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, *Markov Chain Monte Carlo Methods in Practice*. London: Chapman and Hall, 1996.

[31] R. van der Merwe, J. F. G. de Freitas, D. Doucet, and E. A. Wan, "The unscented particle filter," Technical Report CUED/F-INFENG/TR 380, Cambridge University Engineering Department, August 2000.

[32] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, pp. 174–188, February 2002.

[33] M. Ibnkahla, *Signal Processing for Mobile Communications: Handbook*. New York: CRC Press, 2005.

[34] J. R. Treichler, I. Fijalkow, and C. R. Johnson, Jr., "Fractionally spaced equalizers: How long should they really be?," *IEEE Signal Proceesing Mag.*, vol. 13, pp. 65–81, July 1996.

[35] W. C. Jakes, *Microwave Mobile Communications*. NY: John Wiley and Sons, 1974.

[36] M. Yao and B. Rao, "Performance of an array receiver with a Kalman channel predictor for fast Rayleigh flat fading environments," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 6, pp. 1164–1172, 2001.

[37] H. Wang and P. Chang, "On verifying the first-order Markovian assumption for a Rayleigh fading channel model," *IEEE Transactions on Vehicular Technology*, vol. 45, pp. 353–357, 1996.

[38] C. Cowan and S. Semnani, "Time-variant equalization using a novel non-linear adaptive structure," *International Journal of Adaptive Control and Signal Processing*, vol. 12, no. 2, pp. 195–206, 1998.

[39] G. B. Giannakis and C. Tepedelenlioğlu, "Basis expansion models and diversity techniques for blind identification and equalization of time-varying channels," *Proceedings of The IEEE*, vol. 86, pp. 1969–1986, October 1998.

[40] A. Paulraj, R. Nabar, and D. Gore, *Introduction to Space-Time Wireless Communications*. Cambridge, UK: Cambridge University Precess, 2003.

[41] T. C. Clapp, *Statistical Methods for the Processing of Communications Data*. PhD thesis, University of Cambridge, U.K., 2000.

[42] R. W. Lucky, "Automatic equalization for digital communications," *Bell Systems Technical Journal*, vol. 44, pp. 547–588, 1965.

[43] A. Goldsmith, *Wireless Communications*. Cambridge: Cambridge University Press, 2005.

[44] J. G. Proakis, *Digital Communications, 4th Ed.* McGraw-Hill, 2001.

[45] S. Qureshi, "Adaptive equalization," *Proceedings of the IEEE*, vol. 73, pp. 1349–1387, September 1985.

[46] S. Qureshi, "Adaptive equalization," *IEEE Communications Magazine*, vol. 20, pp. 9–16, March 1982.

[47] M. K. Tugnait, L. Tong, and Z. Ding, "Single-user channel estimation and equalization," *IEEE Signal Processing Magazine*, vol. 17, pp. 17–28, May 2000.

[48] C. R. Johnson, Jr., "Direct adaptive equalization: Trained and blind." Technical Note, Cornell University, 2001.

[49] C. R. Johnson, Jr., P. Schniter, T. J. Endres, J. D. Behm, D. R. Brown, and R. A. Casas, "Blind equalization using the constant modulus criterion: A review," *Proceedings of the IEEE*, vol. 86, pp. 1927–1950, October 1998.

[50] J.-J. Werner, J. Yang, D. D. Harman, and G. A. Dumont, "Blind equalization for broadband access," *IEEE Communications Magazine*, vol. 37, pp. 87–93, April 1999.

[51] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification, 2nd Ed.* New York: Wiley Interscience, 2000.

[52] J. F. Hayes, T. M. Cover, and J. B. Riera, "Optimal sequence detection and optimal symbol-by-symbol detection: Similar algorithms," *IEEE Transactions on Communications*, vol. 30, pp. 152–157, January 1982.

[53] E. H. Satorius and S. T. Alexander, "Channel equalization using adaptive lattice algorithms," *IEEE Transactions on Communications*, vol. 27, pp. 899–905, June 1979.

[54] E. H. Satorius and J. D. Pack, "Application of least squares lattice algorithms to adaptive equalization," *IEEE Transactions on Communications*, vol. 29, pp. 136–142, February 1981.

[55] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals, and Systems*, vol. 2, pp. 303–314, 1989.

[56] K. M. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, pp. 359–366, 1989.

[57] S. Haykin, "Adaptive digital communication receivers," *IEEE Communications Magazine*, vol. 38, pp. 106–114, December 2000.

[58] B. Mulgrew, "Applying radial basis function networks," *IEEE Signal Processing Magazine*, vol. 13, pp. 50–65, March 1996.

[59] Q. Liang and J. M. Mendel, "Equalization of nonlinear time-varying channels using type-2 fuzzy adaptive filters," *IEEE Transactions on Fuzzy Systems*, vol. 8, pp. 551–563, October 2000.

[60] L.-X. Wang and J. M. Mendel, "Fuzzy adaptive filters with application to nonlinear channel equalization," *IEEE Transactions on Fuzzy Systems*, vol. 1, pp. 161–170, August 1993.

[61] S. Benedetto and E. Biglieri, "Non linear equalization of satellite channels," *IEEE Journal on Selected Areas in Communications*, vol. 1, pp. 57–62, January 1983.

[62] R. Bernardini, "A fast algorithm for general Volterra filtering," *IEEE Transactions on Communications*, vol. 48, pp. 1853–1864, November 2000.

[63] G. L. Sicuranza, "Quadratic filters for signal processing," *Proceedings of the IEEE*, vol. 80, pp. 1262–1285, August 1992.

[64] J. M. Cioffi, G. P. Dudevoir, V. Eyuboglu, and G. D. Forney, Jr., "MMSE decision-feedback equalizers and coding I: Equalization results," *IEEE Transactions on Communications*, vol. 43, pp. 2582–2594, October 1995.

[65] N. Al-Dhahir and J. M. Cioffi, "MMSE decision-feedback equalizers: Finite-length results," *IEEE Transactions on Information Theory*, vol. 41, pp. 961–975, July 1995.

[66] E. Biglieri, J. Proakis, and S. Shamai, "Fading channels: Information-theoretic and communications aspects," *IEEE Transactions on Information Theory*, vol. 44, pp. 2619–2691, October 1998.

[67] F. Ling and J. G. Proakis, "Adaptive lattice decision-feedback equalizers–Their performance and application to time-variant multipath channels," *IEEE Transactions on Communications*, vol. 33, pp. 348–356, April 1985.

[68] J. M. Cioffi, G. P. Dudevoir, V. Eyuboglu, and G. D. Forney, Jr., "MMSE decision-feedback equalizers and coding II: Coding results," *IEEE Transactions on Communications*, vol. 43, pp. 2595–2604, October 1995.

[69] B. Bjerke, J. G. Proakis, M. K. Lee, and Z. Zvonar, "A comparison of decision feedback equalization and data directed estimation techniques for the GSM system," in *Proc. of IEEE 6th International Conference on Universal Personal Communications*, vol. 1, pp. 84–88, 1997.

[70] S. Siu, G. J. Gibson, and C. F. N. Cowan, "Decision feedback equalisation using neural network structures and performance comparision with standard architecture," *IEE Proceedings: Part I*, vol. 137, no. 4, pp. 221–225, 1990.

[71] S. K. Patra and B. Mulgrew, "Efficient architecture for Bayesian equalization using fuzzy filters," *IEEE Transactions on Circuits and Systems - II: Analog and Digital Signal Processing*, vol. 45, pp. 812–820, July 1998.

[72] S. Chen, B. Mulgrew, and S. McLaughlin, "Adaptive Bayesian equalizer with decision feedback," *IEEE Transactions on Signal Processing*, vol. 41, pp. 2918–2927, September 1993.

[73] S. K. Patra, *Development of Fuzzy System Based Channel Equalisers*. PhD thesis, The University of Edinburgh, U.K., 1998.

[74] S. Chen and B. Mulgrew, "Overcoming co-channel interference using an adaptive radial basis function equaliser," *Signal Processing*, vol. 28, pp. 91–107, 1992.

[75] S. Chen, B. Mulgrew, and P. M. Grant, "A clustering technique for digital communications channel equalization using radial basis function networks," *IEEE Transactions on Neural Networks*, vol. 4, pp. 570–590, July 1993.

[76] J. Larsen, *Design of Neural Network Filters*. PhD thesis, Technical University of Denmark, Denmark, 1993.

[77] M. Ibnkahla, "Applications for neural networks to digital communications - a survey," *Signal Processing*, vol. 80, pp. 1185–1215, 2000.

[78] A. Zerguine, A. Shafi, and M. Bettayeb, "Multilayer perceptron-based DFE with lattice structure," *IEEE Transactions on Neural Networks*, vol. 12, pp. 532–545, May 2001.

[79] S. Haykin, *Neural Networks: A Comprehensive Foundation, 2nd Ed.* Upper Saddle River, NJ: Prentice Hall, 1999.

[80] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Magazine*, vol. 4, pp. 4–22, April 1987.

[81] S. Chen, G. J. Gibson, B. Mulgrew, and S. McLaughlin, "Adaptive equalization of finite nonlilear channels using multilayer perceptrons," *Signal Processing*, vol. 20, pp. 107–119, 1990.

[82] Z. Xiang, G. Bi, and T. Le-Ngoc, "Polynomial perceptrons and their applications to fading channel equalization and co-channel interference suppression," *IEEE Transactions on Signal Processing*, vol. 42, pp. 2470–2480, September 1994.

[83] P.-R. Chang and B.-C. Wang, "Adaptive decision feedback equalization for digital satellite channels using multilayer neural networks," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 316–324, February 1995.

[84] P.-R. Chang, B. F. Yeg, and C. C. Chang, "Adaptive packet equalization for indoor radio channels using multilayer neural networks," *IEEE Transactions on Vehicular Technology*, vol. 43, pp. 773–780, August 1994.

[85] S. K. Nair and J. Moon, "A theoretical study of linear and nonlinear equalization in nonlinear magnetic storage channels," *IEEE Transactions on Neural Networks*, vol. 8, pp. 1106–1110, September 1997.

[86] A. Shafi, A. Zerguine, and M. Bettayeb, "Neural network-based decision feedback equalizer with lattice structure," *Electronics Letters*, vol. 35, pp. 1705–1707, September 1999.

[87] Y. Iiguni, H. Sakai, and H. Tokumaru, "A real-time learning algorithm for a multilayered neural network based on the extended Kalman filter," *IEEE Transactions on Signal Processing*, vol. 40, no. 4, pp. 959–966, 1992.

[88] R. Parisi, E. D. Di Claudio, G. Orlandi, and B. D. Rao, "A generalized learning paradigm exploiting the structure of neural nets," *IEEE Transactions on Neural Networks*, vol. 7, pp. 1450–1460, November 1996.

[89] C. Sweatman, B. Mulgrew, and G. J. Gibson, "Two algorithms for neural network design and training with application to channel equalization," *IEEE Transactions on Neural Networks*, vol. 9, pp. 533–543, May 1998.

[90] M. J. D. Powell, "Radial basis functions for multivariable interpolation: A review," in Algorithms for Approximation: J.C. Mason and M. G. COx (Eds.), pp. 143–167, Clarendon Press, Oxford, 1987.

[91] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," *Complex Systems*, vol. 2, pp. 321–355, 1988.

[92] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithms for radial basis function networks," *IEEE Transactions on Neural Networks*, vol. 2, pp. 302–309, January 1991.

[93] L. Tarassenko and S. Roberts, "Supervised and unsupervised learning in radial basis function classifiers," *IEE Proceedings - Vision, Image and Signal Processing*, vol. 141, pp. 210–216, August 1994.

[94] S. Chen, S. McLaughlin, and B. Mulgrew, "Complex-valued radial basis function networks, Part I: Network architecture and learning algorithms," *Signal Processing*, vol. 35, pp. 19–31, January 1994.

[95] S. Chen, S. McLaughlin, and B. Mulgrew, "Complex-valued radial basis function networks, Part II: Application to digital communication channel equalization," *Signal Processing*, vol. 36, pp. 175–188, January 1994.

[96] I. Cha and S. A. Kassam, "Channel equalization using adaptive complex radial basis function networks," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 122–131, January 1995.

[97] J. Cid-Sueiro, A. Artes-Rodriguez, and A. R. Figueiras-Vidal, "Recurrent radial basis function networks for optimal symbol-by-symbol equalization," *Signal Processing*, vol. 40, pp. 53–63, 1994.

[98] D. Jianping, N. Sundararajan, and P. Saratchandran, "Nonlinear magnetic storage channel equalization using minimal resource allocation network (MRAN)," *IEEE Transactions on Neural Networks*, vol. 12, pp. 171–174, January 2001.

[99] D. Jianping, N. Sundararajan, and P. Saratchandran, "Communication channel equalization using complex-valued minimal radial basis function neural networks," *IEEE Transactions on Neural Networks*, vol. 13, pp. 687–696, May 2002.

[100] L. Yingwei, N. Sundararajan, and P. Saratchandran, "A sequential learning scheme for function approximation using minimal radial basis function networks," *Neural Computation*, vol. 9, pp. 461–478, February 1997.

[101] J. Choi, A. C. de C. Lima, and S. Haykin, "Unscented Kalman filter-trained recurrent neural equalizer for time-varying channel," in *Proceedings of IEEE International Conference on Communications*, pp. 3241–3245, Anchorage, Alaska, USA, May 2003.

[102] J. Choi, A. C. de C. Lima, and S. Haykin, "Kalman filter-trained recurrent neural equalizers for time-varying channels," *IEEE Transactions on Communications*, vol. 53, pp. 472–480, March 2005.

[103] J. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, pp. 179–211, 1990.

[104] R. Parisi, E. D. Di Claudio, G. Orlandi, and B. D. Rao, "Fast adaptive digital equalization by recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2731–2739, November 1997.

[105] G. Kechriotis, E. Zervas, and E. S. Manolakos, "Using recurrent neural networks for adaptive communication channel equalizations," *IEEE Transactions on Neural Networks*, vol. 5, pp. 267–278, March 1994.

[106] S. Ong, C. You, S. Choi, and D. Hong, "A decision feedback recurrent neural equalizer as an infinite impulse response filter," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2851–2858, November 1997.

[107] K. Hacioglu, "Comments on "A decision feedback recurrent neural equalizer as an infinite impulse response filter"," *IEEE Transactions on Signal Processing*, vol. 48, pp. 1206–1207, April 2000.

[108] D.-C. Park and T.-K. Jeong, "Complex-bilinear recurrent neural network for equalization of a digital satellite channel," *IEEE Transactions on Neural Networks*, vol. 13, pp. 711–725, May 2002.

[109] M. J. Bradley and P. Mars, "Application of recurrent neural networks to communication channel equalization," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 3399 –3402, 1995.

[110] F. M. Hsu, "Square-root Kalman filtering for high speed data received over fading dispersive HF channels," *IEEE Transactions on Information Theory*, vol. IT-28, pp. 753–763, September 1982.

[111] J. M. Cioffi and T. Kailath, "Fast recursive least-squares transversal filters for adaptive filtering," *IEEE Trans. on Acoust., Speech Signal Processing*, vol. ASSP-32, pp. 304–337, April 1984.

[112] D. T. M. Slock and T. Kailath, "Numerically fast transversal filters for recursive least-squares adaptive filtering," *IEEE Transactions on Signal Processing*, vol. 39, pp. 92–114, January 1991.

[113] A. H. Sayed and T. Kailath, "A state-space approach to adaptive RLS filtering," *IEEE Signal Processing Magazine*, vol. 11, pp. 18–60, July 1994.

[114] M. R. Aaron and D. Tufts, "Intersymbol interference and error probability," *IEEE Transactions on Information Theory*, vol. 12, pp. 26–34, January 1966.

[115] E. Shamash and K. Yao, "On the structure and performance of a linear decision feedback equalizer based on the minimum error probability criterion," in *Proc. Int. Conf. Communications*, pp. 25F1–25F4, Minneapolis, MN, 1974.

[116] C. Yeh and J. Barry, "Adaptive minimum bit-error rate qqualization for binary signaling," *IEEE Transactions on Communications*, vol. 48, pp. 1226–1235, July 2000.

[117] I. Santamaria, D. Erdogmus, and J. C. Principe, "Entropy minimization for supervised digital communications channel equalization," *IEEE Transactions on Signal Processing*, vol. 50, pp. 1184–1192, May 2002.

[118] S. Choi and T.-W. Lee, "A negentropy minimization approach to adaptive equalization for digital communication systems," *IEEE Transactions on Neural Networks*, vol. 15, pp. 928–936, May 2004.

[119] C. Douillard, M. Jezequel, C. Berrou, A. Picart, P. Didier, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo equalization," *European Trans. Telecomm.*, vol. 6, pp. 507–511, Sept.-Oct. 1995.

[120] R. Koetter, A. Singer, and M. Tüchler, "Turbo equalization," *IEEE Signal Processing Magazine*, vol. 21, pp. 67–80, January 2004.

[121] G. Bauch, A. Naguib, and N. Seshadri, "MAP equalization of space-time coded signals over frequency selective channels," in *Proceedings of IEEE Wireless Communications and Networking Conference*, pp. 261–265, New Orleance, USA, 21-24 September 1999.

[122] J. Hagenauer, E. Offer, C. Méasson, and M. Mörz, "Decoding and equalization with analog non-linear networks," *European Transactions on Telecommunications*, vol. 10, pp. 107–128, Nov.-Dec. 1999.

[123] C. Laot, A. Glavieux, and J. Labat, "Turbo equalization: Adaptive equalization and channel decoding jointly optimized," *IEEE Journal on Selected Areas in Communications*, vol. 19, pp. 1744–1752, September 2001.

[124] M. Tüchler, A. Singer, and R. Koetter, "Minimum mean squared error equalization using *a priori* information," *IEEE Transactions on Signal Processing*, vol. 50, pp. 673–683, March 2002.

[125] M. Tüchler, R. Koetter, and A. Singer, "Turbo equalization: principles and new results," *IEEE Transactions on Communications*, vol. 50, pp. 754–767, May 2002.

[126] A. Glavieux, C. Laot, and J. Labat, "Turbo equalization over a frequency selective channel," in *The 1st International Symposium on Turbo Codes and Related Topics (ISTC)*, pp. 96–102, Brest, France, September 1997.

[127] M. Tüchler and J. Hagenauer, "Frequency domain turbo equalization," in *Proc. Winterschool on Coding and Information Theory*, Ulm, Germany, December 2000.

[128] D. Raphaeli and T. Kaitz, "A reduced-complexity algorithm for combined equalization and decoding," *IEEE Transactions on Communications*, vol. 48, pp. 1940–1943, November 2000.

[129] M. S. Yee, T. H. Liew, and L. Hanzo, "Burst-by-burst adaptive turbo-coded Radial Basis Function-assisted decision feedback equalization," *IEEE Transactions on Communications*, vol. 49, pp. 1935–1945, November 2001.

[130] M. S. Yee, S. X. Ng, and L. Hanzo, "Radial basis function assisted reduced complexity in-phase/quadrature-phase turbo equalisation of coded modulation schemes," in *Proceedings of IEEE Semiannual Vehicular Technology Conference*, pp. 2367–2371, April 2003.

[131] G. Bauch and N. Al-Dhahir, "Reduced-complexity space time turbo-equalization for frequency-selective MIMO channels," *IEEE Transactions on Wireless Communications*, vol. 1, pp. 819–928, October 2002.

[132] H.-N. Lee and V. Gulati, "Iterative equalization/decoding of LDPC code transmitted over MIMO ISI fading channels," in *Proceedings of IEEE International Symposium on Personal, Indoor ad Mobile Radio Communications*, pp. 1330–1336, Lisbon, Portugal, September 2002.

[133] T. Abe and T. Matsumoto, "A MIMO equalizer for frequency-selective MIMO channels with unkwon interference," *IEEE Transactions on Vehicular Technology*, vol. 52, pp. 476–482, May 2003.

[134] T. Abe and T. Matsumoto, "Space-time turbo equalization in frequency-selective MIMO channels," *IEEE Transactions on Vehicular Technology*, vol. 52, pp. 469–475, May 2003.

[135] Y. Sun, M. Yee, and M. Sandell, "Iterative channel estimation with MIMO MMSE-turbo equalization," in *Proceedings of IEEE Semiannual Vehicular Technology Conference*, pp. 1278–1282, Orlando, FL, USA, October 2003.

[136] E. de Carvalho and D. T. C. Slock, "Blind and semi-blind FIR multichannel estimation: (global) identifiability conditions," *IEEE Transactions on Signal Processing*, vol. 52, pp. 1053–1064, April 2004.

[137] L. Tong, B. M. Sadler, and M. Ding, "Pilot-assisted wireless transmissions: General model, design criteria, and signal processing," *IEEE Signal Processing Magazine*, vol. 21, pp. 12–25, November 2004.

[138] A. Furuskär, S. Mazur, F. Müller, and H. Olofsson, "EDGE: Enhanced data rates gsm and TDMA/136 evolution," *IEEE Personal Communications*, vol. 6, pp. 56–66, June 1999.

[139] D. J. Goodman, "Second generation wireless information networks," *IEEE Transactions on Vehicular Technology*, vol. 40, pp. 366–374, May 1991.

[140] M. Austin, A. Buckley, C. Coursey, P. Hartman, R. Kobylinski, M. Majmundar, K. Raith, and J. P. Seymour, "Service and system enhancements for TDMA digital cellular systems," *IEEE Personal Communications*, vol. 6, pp. 20–33, June 1999.

[141] H.-N. Lee, *Adaptive Diversity Combining, Equalization and Sequence Decoding for Time-Varying Dispersive Channels*. PhD thesis, University of California, Los Angeles, USA, 1999.

[142] H. Arslan and G. E. Bottomley, "Channel estimation in narrowband wireless communication systems," *Wireless Communications and Mobile Computing*, vol. 1, pp. 201–219, 2001.

[143] H.-K. Song, "A channel estimation using sliding window approach and tuning algorithm for mlse," *IEEE Transactions on Communications Letters*, vol. 3, pp. 211–213, July 1999.

[144] G. Benelli, A. Garzelli, and F. Salve, "Simplified viterbi processors for the GSM pan-european cellular communication system," *IEEE Transactions on Vehicular Technology*, vol. 43, pp. 870–878, November 1994.

[145] J. I. Park, S. B. Wicker, and H. L. Owen, "Trellis-based soft-output adaptive equalization techniques for TDMA cellular systems," *IEEE Transactions on Vehicular Technology*, vol. 49, pp. 83–94, January 2000.

[146] M. Pukkila, *Iterative Receivers And Multichannel Equalisation For Time Division Multiple Access Systems*. PhD thesis, Helsinki University of Technology, Finland, 2003.

[147] R. Trepkowski, "Channel estimation strategies for coded mimo systems," Master's thesis, Virginia Polytechnic Institute and State University, USA, 2004.

[148] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Upper Saddle River, NJ: Prentice Hall, 1998.

[149] H.-N. Lee and G. J. Pottie, "Fast adaptive equalization/diversity combining for time-varying dispersive channels," *IEEE Transactions on Communications*, vol. 46, pp. 1146–1162, September 1998.

[150] L. M. Davis, I. B. Collings, and P. Hoeher, "Joint MAP equalization and channel estimation for frequency-selective and frequency-flat fast-fading channels," *IEEE Transactions on Communications*, vol. 49, pp. 2106–2114, December 2001.

[151] D. Schafhuber, G. Matz, F. Hlawatsch, and P. Loubaton, "MMSE estimation of time-varying channels for DVB-T systems with strong co-channel interference," in *Proceedings of European Signal Processing Conference*, pp. 24–28, Toulouse, France, September 2002.

[152] B. C. Ng, M. Cedervall, and A. Paulraj, "A structured channel estimator for maximum-likelihood sequence detection," *IEEE Transactions on Communications Letters*, vol. 1, pp. 52–55, March 1997.

[153] S. Sampei, "Computation reduction of decision feedback equalizer using interpolation for land mobile communications," in *Proceedings of IEEE Global Telecommunications Conference*, pp. 521–525, Pheonix, AZ, December 1991.

[154] G. W. Davidson, D. D. Falconer, and A. U. H. Sheikh, "An investigation of block adaptive decision feedback equalization for frequency-selective fading channes," in *Proceedings of IEEE International Conference on Communications*, pp. 360–365, Atlanta, GA, 1998.

[155] S. A. Fechtel and H. Meyer, "Optimal parametric feedforwared estimation of frequency-selective fading radio channels," *IEEE Transactions on Communications*, vol. 42, pp. 1639–1650, Feb/Mar/Apl 1994.

[156] H. Meyer, M. Moeneclaey, and S. A. Fechtel, *Digital Communication Receivers: Synchronization, Channel Estimation, and Signal Processing*. New York: Jone Wiley and Sons, 1998.

[157] E. de Carvalho and D. T. C. Slock, "Asymptotic performance of ML methods for semi-blind channel estimation," in *Proceedings of Asilomar Conference on Signals, Systems and Computers*, pp. 1624–1628, Pacific Grove, CA, November 1997.

[158] E. de Carvalho and D. T. C. Slock, "Maximum-likelihood FIR multi-channel estimaiton with Gaussian prior for the symbols," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 3593–3596, Munich, Germany, April 1997.

[159] E. de Carvalho and D. T. C. Slock, "Cramér-Rao bounds for semi-blind, blind and training sequence based channel estimation," in *Proceedings of IEEE Workshop on Signal Processing Advances in Wireless Communications*, pp. 353–356, Paris, France, April 1997.

[160] H. A. Cirpan and M. K. Tsdatsanis, "Stochastic maximum likelihood methods for semi-blind channel estimation," *IEEE Signal Processing Letters*, vol. 5, pp. 21–24, January 1998.

[161] G. K. Kaleh and R. Vallet, "Joint parameter estimation and symbol detection for linear or nonlinear unknown channels," *IEEE Transactions on Communications*, vol. 42, pp. 2406–2413, July 1994.

[162] J. Laurila, K. Kopsa, and E. Bonek, "Semi-blind signal estimation for smart antennas using subspace tracking," in *Proceedings of IEEE Workshop on Signal Processing Advances in Wireless Communications*, pp. 271–274, Annapolis, MD, May 1999.

[163] A. Gorokhov and P. Loubaton, "Semi-blind second order identification of convolutive channels," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 3905–3908, Munich, Germany, April 1997.

[164] S. Lasaulce, P. Loubaton, and E. Moulines, "A semi-blind channel estimation technique based on second-order blind method for CDMA systems," *IEEE Transactions on Signal Processing*, vol. 51, pp. 1894–1904, July 2003.

[165] B. Farhang-Boroujeny, *Adaptive Filters: Theory and Applications*. Chichester, UK: John Wiley and Sons, 1998.

[166] D. Schafhuber, *Wireless OFDM Systems: Channel Prediction and System Capacity*. PhD thesis, Technische Universität Wien, Austria, 2004.

[167] A. Duel-Hallen, S. Hu, and H. Hallen, "Long-range prediction of fading signals," *IEEE Signal Processing Magazine*, vol. 17, pp. 62–75, May 2000.

[168] D. K. Borah and B. D. Hart, "Frequency-selective fading channel estimation with a polynomial time-varying channel model," *IEEE Transactions on Communications*, vol. 47, pp. 862–873, June 1999.

[169] A. Duel-Hallen, S. Hu, and H. Hallen, "Adaptation and tracking in system identification A survey," *Automatica*, vol. 26, pp. 7–21, January 1990.

[170] L. M. Davis, I. B. Collings, and R. J. Evans, "Coupled estimators for equalization of fast-fading mobile channels," *IEEE Transactions on Communications*, vol. 46, pp. 1262–1265, October 1998.

[171] L. Lindbom, "Simplified Kalman estimation of fading mobile radio channels: High performance at LMS computational load," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 352–355, Minneapolis, MN, April 1993.

[172] R. A. Ziegler and J. M. Cioffi, "Estimation of time-varying digital radio channels," *IEEE Transactions on Vehicular Technology*, vol. 41, pp. 134–151, May 1992.

[173] F. R. Magee and J. G. Proakis, "Adaptive maximum-likelihood sequence estimation for digital signaling in the presence of intersymbol interference," *IEEE Transactions on Information Theory*, vol. 19, pp. 120–124, January 1973.

[174] A. H. Sayed, *Fundamentals of Adaptive Filtering*. John Wiley and Sons, 2003.

[175] N. R. Yousef and A. H. Sayed, "Ability of adaptive filters to track carrier offsets and channel nonstationarities," *IEEE Transactions on Signal Processing*, vol. 50, pp. 1533–1544, July 2002.

[176] J. F. Galdino, E. L. Pinto, and M. S. Alencar, "Analytical performance of the LMS algorithm on the estimation of wide sense stationary channels," *IEEE Transactions on Communications*, vol. 52, pp. 982–991, June 2004.

[177] Y. Xue and X. Zhu, "Wireless channel tracking based on self-tuning second-order LMS algorithm," *IEE Proc.-Commun.*, vol. 150, pp. 115–120, April 2003.

[178] E. Eleftheriou and D. D. Falconer, "Tracking properties and steady-state performance of rls adaptive filter algorithms," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, pp. 1097–1110, October 1986.

[179] J. Lin, J. G. Proakis, F. Ling, and H. Lev-Ari, "Optimal tracking of time-varying channels: A frequency domain approach for known and new algorithms," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 141–154, January 1995.

[180] J. Wu and A. H. Aghvami, "A new adaptive equalizer with channel estimator for mobile radio communications," *IEEE Transactions on Vehicular Technology*, vol. 45, pp. 467–414, August 1996.

[181] E. Baccarelli and R. Cusani, "Combined channel estimation and data detection using soft statistics for frequency-selective fast-fading digital links," *IEEE Transactions on Communications*, vol. 46, pp. 424–427, April 1998.

[182] R. Chen, X. Wang, and J. S. Liu, "Adaptive joint detection and decoding in flat-fading channels via mixture kalman filtering," *IEEE Transactions on Information Theory*, vol. 46, pp. 2079–2094, September 2000.

[183] Z. Cheng and D. Dahlhaus, "Time versus frequency domain channel tracking using Kalman filters for OFDM systems with antenna arrays," in *Proceedings of IEEE Semiannual Vehicular Technology Conference*, vol. 1, pp. 651–655, Jeju, Korea, April 2003.

[184] K. J. Kim, J. Yue, R. A. Iltis, and J. D. Gibson, "A QRD-M/Kalman filter-based detection and channel estimation algorithm for MIMO-OFDM systems," *IEEE Transactions on Wireless Communications*, vol. 4, pp. 710–721, March 2005.

[185] C. Komninakis, C. Fragouli, A. H. Sayed, and R. D. Wesel, "Multi-input multi-output fading channel tracking and equalization using Kalman estimation," *IEEE Transactions on Signal Processing*, vol. 50, pp. 1065–1076, May 2002.

[186] Z. Liu, X. Ma, and G. Giannakis, "Space-time coding and Kalman filtering for time-selective fading channels," *IEEE Transactions on Communications*, vol. 50, pp. 183–186, February 2002.

[187] S. Song, A. C. Singer, and K.-M. Sung, "Turbo equalization with an unknown channel," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 2805–2808, Orlando, FL, USA, May 2002.

[188] G. E. Bottomley and K. J. Molnar, "Adaptive channel estimation for multichannel MLSE receivers," *IEEE Transactions on Communications Letters*, vol. 3, pp. 40–42, February 1999.

[189] Q. Dai and E. Shwedyk, "Detection of bandlimited signals over frequency selective rayleigh fading channels," *IEEE Transactions on Communications*, vol. 42, pp. 941–950, Feb/Mar/Apr 1994.

[190] H. Zamiri-Jafarian and S. Pasupathy, "Adaptive mlsd receiver with identification of flat fading channels," in *Proceedings of IEEE Semiannual Vehicular Technology Conference*, pp. 695–699, Phoenix, AZ, USA, May 1997.

[191] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding," in *Proceedings of IEEE International Conference on Communications*, pp. 1064–1070, Geneva, Switzerland, May 1993.

[192] M. Sandell, C. Luschi, P. Strauch, and R. Yan, "Iterative channel estimation using soft decision feedback," in *Proceedings of IEEE Global Telecommunications Conference*, pp. 3728–3733, Sydney, Australia, November 1998.

[193] M. Valenti and B. D. Woerner, "Iterative channel estimaiton and decoding of pilot symbol assisted turbo codes over flat-fading channels," *IEEE Journal on Selected Areas in Communications*, vol. 19, pp. 1697–1705, September 2001.

[194] R. R. Lopes and J. R. Barry, "The extended-window channel estimator for iterative channel-and-symbol estimation," *EURASIP Journal onWireless Communications and Networking*, no. 2, pp. 92–99, 2005.

[195] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, pp. 284–287, February 1974.

[196] A. O. Berthet, B. S. Ünal, and R. Visoz, "Iterative decoding of convolutionally encoded signals over multipath Rayleigh fading channels," *IEEE Journal on Selected Areas in Communications*, vol. 19, pp. 1729–1743, September 2001.

[197] N. Nefedov, M. Pukkila, R. Visoz, and A. O. Bethet, "Iterative receiver concept for TDMA packet data systems," *European Transactions on Telecommunications*, vol. 14, pp. 457–469, September/October 2003.

[198] K.-D. Kammeyer, V. Kühn, and T. Petermann, "Blind and nonblind turbo estimation for fast fading GSM channels," *IEEE Journal on Selected Areas in Communications*, vol. 19, pp. 1718–1728, September 2001.

[199] B. L. Yeap, C. H. Wong, and L. Hanzo, "Reduced complexity In-phase/Quadrature-phase M-QAM turbo equalization using iterative channel estimation," *IEEE Transactions on Wireless Communications*, vol. 2, pp. 2–10, January 2003.

[200] J. Wehinger, C. F. Mecklenbräuker, R. R. Müller, T. Zemen, and M. Lončar, "On channel estimators for iterative CDMA multiuser receivers in flat Rayleigh fading," in *Proceedings of IEEE International Conference on Communications*, vol. 5, pp. 2497–2501, Paris, France, June 2004.

[201] M. Tüchler, R. Otnes, and R. Schmidbauer, "Performance of soft iterative channel estimation in turbo equalization," in *Proceedings of IEEE International Conference on Communications*, vol. 3, pp. 1858–1862, New York, USA, April/May 2002.

[202] R. Otnes and M. Tüchler, "Soft iterative channel estimation for turbo equalization: Comparison of channel estimation algorithms," in *The 8th International Conference on Communication Systems*, vol. 1, pp. 72–76, Singapore, November 2002.

[203] R. Kimura, D. Garg, and F. Adachi, "Comparative study of iterative channel estimation schemes for turbo decoding with antenna diversity reception in rayleigh fading," *IEICE Transactions on Communcations*, vol. E86-B, pp. 1149–1152, March 2003.

[204] R. Otnes and M. Tüchler, "Iterative channel estimation for turbo equalization of time-varying frequency-selective channels," *IEEE Transactions on Wireless Communications*, vol. 4, pp. 1918–1923, November 2004.

[205] T. Marzetta, "BLAST training: Estimating channel characteristics for high-capacity space-time wireless," in *Proceedings of Allerton Conference on Communication, Control, and Computing*, pp. 958–966, Monticello, IL, September 1999.

[206] H. Bölcskei, R. W. Heath, Jr., and A. J. Paulraj, "Blind channel identification and equalization in OFDM-based multiantenna systems," *IEEE Transactions on Signal Processing*, vol. 50, pp. 96–109, January 2002.

[207] B. Hassibi and B. M. Hochwald, "How much training is needed in multiple-antenna wireless links?," *IEEE Transactions on Information Theory*, vol. 49, pp. 951–963, April 2003.

[208] C. Budianu and L. Tong, "Channel estimation for space-time orthogonal block codes," *IEEE Transactions on Signal Processing*, vol. 50, pp. 2515–2528, October 2002.

[209] L.-U. Choi and R. D. Murch, "A pre-BLAST-DFE technique for the downlink of frequency-selective fading MIMO channels," *IEEE Transactions on Communications*, vol. 52, pp. 737–743, May 2004.

[210] A. Grant, "Joint decoding and channel estimation for linear MIMO channels," in *Proceedings of IEEE Wireless Communications and Networking Conference*, pp. 1009–1012, Chicago, IL, September 2000.

[211] H. Zhu, B. Farhang-Boroujeny, and C. Schlegel, "Pilot embedding for joint channel estimation and data detection in MIMO communication systems," *IEEE Transactions on Communications Letters*, vol. 7, pp. 30–32, January 2003.

[212] X. Deng, A. M. Haimovich, and J. Garcia-Frias, "Decision directed iterative channel estimation for mimo systems," in *Proceedings of IEEE International Conference on Communications*, pp. 2326–2329, Ancharage, AK, USA, May 2003.

[213] M. Lončar, R. R. Müller, T. Abe, J. Wehinger, and C. F. Mecklenbräuker, "Iterative equalization using soft-decoder feedback for MIMO systems in frequency-selective fading," in *Proc. of URSI General Assembly*, Maastricht, Holland, August 2002.

[214] T. Abe, S. Tosisato, and T. Matsumoto, "Performance evaluation of a space-time turbo equalizer in frequency selective MIMO channels using field measurement data," in *Proc. of IEE Seminar on MIMO: Communications Systems from Concepts to Implementations*, pp. 21.1–21.5, London, UK, December 2001.

[215] M. Lončar, R. R. Müller, J. Wehinger, and T. Abe, "Iterative joint detection, decoding, and channel estimation for dual antenna arrays in frequency selective fading," in *Proc. of the 5th International Symposium on Wireless Personal Multimedia Communications*, Honolulu, Hawaii, October 2002.

[216] M. Lončar, R. R. Müller, J. Wehinger, C. F. Mecklenbräuker, and T. Abe, "Iterative channel estimation and data detection in frequency-selective fading MIMO channels," *European Transactions on Telecommunications*, vol. 15, pp. 459–470, September 2004.

[217] J. Míguez and L. Castedo, "Space-time channel estimaiton and soft detection in time-varying multiaccess channels," *Signal Processing*, vol. 83, pp. 389–411, 2003.

[218] H. Mai and A. G. Burr, "Iterative channel estimation for turbo equalization over MIMO double selective fading channels," in *Proceedings of IEEE Semiannual Vehicular Technology Conference*, pp. 1478–1482, Los Angeles, CA, USA, September 2004.

[219] H. A. Çirpan, E. Panayirci, and H. Doğan, "Iterative channel estimation approach for space-time/frequency coded OFDM systems with transmitter diversity," *European Transactions on Telecommunications*, vol. 15, pp. 235–248, May 2004.

[220] Y. Sun and M. S. Yee, "MIMO iterative array processing with LMMSE turbo equalization," in *Proceedings of IEEE Semiannual Vehicular Technology Conference*, pp. 1585–1589, Los Angeles, CA, USA, September 2004.

[221] D. Schafhuber, G. Matz, and F. Hlawatsch, "Kalman tracking of time-varying channels in wireless MIMO-OFDM systems," in *Proceedings of Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 1261–1265, Pacific Grove, CA, USA, November 2003.

[222] P. M. Djurić, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Míguez, "Particle filtering," *IEEE Signal Processing Magazine*, pp. 19–38, September 2003.

[223] E. Punskaya, C. Andrieu, A. Doucet, and W. J. Fitzgerald, "Particle filtering for demodulation in fading channels with non-Gaussian additive noise," *IEEE Transactions on Communications*, vol. 49, pp. 579–582, April 2001.

[224] C. Andrieu, A. Doucet, and A. Touzni, "Adaptive MAP muti-user detection for fading CDMA channels," in *Proc. IEEE Workshop SSAP*, pp. 6–9, Pocono Manor, PA, 2000.

[225] E. Punskaya, C. Andrieu, A. Doucet, and W. J. Fitzgerald, "Particle filtering for mutiuser detection for fading CDMA channels," in *Proc. 11th IEEE Workshop SSP*, pp. 38–41, Singapore, 2001.

[226] W. H. Chin, D. B. Ward, and A. G. Constantinides, "Semi-blind channel tracking using auxiliary particle filtering," in *Proceedings of IEEE Global Telecommunications Conference*, pp. 322–325, November 2002.

[227] W. H. Chin, D. B. Ward, and A. G. Constantinides, "Channel tracking for space-time block coded systems using particle filtering," in *Proc. 14th International Conference on Digital Sigal Processing*, pp. 671–674, July 2002.

[228] C. Huber and S. Haykin, "Application of particle filters to MIMO wireless communications," in *Proceedings of IEEE International Conference on Communications*, pp. 2311–2315, Anchorage, Alaska, USA, May 2003.

[229] K. L. Blackard, T. S. Rappaport, and C. W. Bostian, "Measurements and models of radio frequency impulsive noise for indoor wireless communications," *IEEE Journal on Selected Areas in Communications*, vol. 11, pp. 991–1001, September 1993.

[230] X. Wang and H. V. Poor, "Robust multiuser detection in non-Gaussian channels," *IEEE Transactions on Signal Processing*, vol. 47, pp. 289–305, February 1999.

[231] W. H. Chin, D. B. Ward, and A. G. Constantinides, "A block refinement scheme for tracked MIMO channel estimates," in *Proc. London Communications Symposium*, pp. 357–360, September 2003.

[232] S. Haykin and K. Huber, "Improved bayesian MIMO channel tracking for wireless communications: Incorporating a dynamical channel," *submitted for publication*, 2004.

[233] D. Chizhik, J. Ling, P. W. Wolniansky, R. A. Valenzuela, N. Costa, and K. Huber, "Multiple-input multiple-output measurements and modelling in manhattan," *IEEE Journal on Selected Areas in Communications*, vol. 21, pp. 321–331, April 2003.

[234] S. Hong, M. Bolić, and P. M. Djurić, "An efficient fixed-point implementation of residual resampling scheme for high-speed particle filters," *IEEE Signal Processing Letters*, vol. 11, pp. 482–485, May 2004.

[235] M. Bolić, P. M. Djurić, and S. Hong, "Resampling algorithms and architectures for distributed particle filters," *accepted for publication in IEEE Transactions on Signal Processing*, 2004.

[236] T. K. Blankenship, D. M. Krizman, and T. S. Rappaport, "Measurements and simulation of radio frequency impulsive noise in hospitals and clinics," in *Proceedings of IEEE Semiannual Vehicular Technology Conference*, pp. 1942–1946, 1997.

[237] P. L. Brockett, M. Hinich, and G. R. Wilson, "Nonlinear and non-Gaussian ocean noise," *J. Acoust. Soc. Am.*, vol. 82, pp. 1286–1399, 1987.

[238] S. A. Kassam, G. Moustakides, and J. G. Shin, "Robust detection of known signals in asymmetric noise," *IEEE Transactions on Information Theory*, vol. 28, pp. 84–91, January 1982.

[239] D. Middleton, "Channel modeling and threshold signal processing in underwater acoustics: An analytical overview," *IEEE Journal of Ocean Engineering*, vol. 12, pp. 4–28, 1987.

[240] S. M. Zabin and H. V. Poor, "Efficient estimation of the class A parameters via the EM algorithm," *IEEE Transactions on Information Theory*, vol. 37, pp. 60–72, January 1991.

[241] J. Häring and A. J. H. Vinck, "Performance bounds for optimum and suboptimum reception under Class-A impulsive noide," *IEEE Transactions on Communications*, vol. 50, pp. 1130–1136, July 2002.

[242] A. D. Spaulding and D. Middleton, "Optimum reception in an impulsive interference environments – Part I: Coherent detection," *IEEE Transactions on Communications*, vol. 25, pp. 910–923, September 1977.

[243] B. B. Madan, S. Prasad, and A. K. Mahalanbis, "On optimum receivers for analog communication in impulsive noise," *IEEE Transactions on Communications*, vol. 26, pp. 1446–1454, October 1978.

[244] X. Wang and R. Chen, "Adaptive bayesian multiuser detection for synchronous CDMA with Gaussian and impulsive noise," *IEEE Transactions on Signal Processing*, vol. 47, pp. 2013–2028, July 2000.

[245] P. Spasojević and X. Wang, "Imporved robust multiuser deteciton in non-Gaussian channels," *IEEE Signal Processing Letters*, vol. 8, pp. 83–86, March 2001.

[246] H. V. Poor and M. Tanda, "Multiuser deteciton in flat fading non-Gaussian channels," *IEEE Transactions on Communications*, vol. 50, pp. 1769–1777, November 2002.

[247] V. D. Phan and X. Wang, "Bayesian turbo multiuser detection for nonlinearly modulated CDMA," *Signal Processing*, vol. 82, pp. 43–68, 2002.

[248] X. Wang and H. V. Poor, *Wireless Communication Systems: Advanced Techniques for Signal Reception*. Upper Saddle River, NJ: Prentice Hall, 2004.

[249] R. Fa, C. C. Tsimenidis, and B. S. Sharif, "Robust MMSE receiver employing Hampel-type nonlinear preprocessing for DS-CDMA communication in impulsive noise," *Electronics Letters*, vol. 40, pp. 489–491, 15th April 2004.

[250] B. Sayadi and S. Marcos, "Robust equalization based on a network of Kalman filters in impulsive noise environments," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. V, pp. 37–40, Hong Kong, April 2003.

[251] C. Luschi and B. Mulgrew, "Nonparametric trellis equalization in the presence of non-Gaussian interference," *IEEE Transactions on Communications*, vol. 51, pp. 229–239, February 2003.

[252] R. S. Blum, R. J. Kozick, and B. M. Sadler, "An adaptive spatial diversity receiver for non-Gaussian interference and noide," *IEEE Transactions on Signal Processing*, vol. 47, pp. 2100–2111, August 1999.

[253] Y. Yap and H. V. Poor, "Blind detection of synchronous CDMA in non-Gaussian channels," *IEEE Transactions on Signal Processing*, vol. 52, pp. 271–279, January 2004.

[254] R. Schober and L. Lampe, "Sequence detection and adaptive channel estimation for ISI channels under class-A impulsive noise," *IEEE Transactions on Communications*, vol. 52, pp. 1523–1531, September 2004.

[255] S. N. Batalama, M. J. Medley, and I. N. Psaromiligkos, "Adaptive robust spread-spectrum receivers," *IEEE Transactions on Communications*, vol. 47, pp. 905–917, June 1999.

[256] S. N. Batalama, M. J. Medley, and I. N. Psaromiligkos, "Robust adaptive recovery of spread-spectrum signals with short data records," *IEEE Transactions on Communications*, vol. 48, pp. 1725–1731, October 2000.

[257] D. A. Pados, M. J. Medley, and S. N. Batalama, "Adaptive antenna array receivers for spread-spectrum signals in non-Gaussian noide," *Digital Signal Processing*, vol. 11, pp. 144–158, 2001.

[258] J. G. Gonzalez, *Robust Techniques for Wireless Communications in Non-Gaussian Environments*. PhD thesis, University of Delaware, USA, 1997.

[259] A. G. Dabak and D. H. Johnson, "Signal constellations for non-Gaussian communication problems," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. III, pp. 33–36, 27-30 April 1993.

[260] J.-O. Gustavsson, S. Nordebo, and P. O. Börjesson, "Simultaneous channel and symbol maximum likelihood estimation in Laplacian noise," in *Proceedings of the Fourth International Conference on Signal Processing (ICSP)*, vol. I, pp. 81–84, Beijing, China, October 1998.

[261] J. Bas and A. N. Neira, "Fuzzy median filter for CDMA detection in non-Gaussian channels," in *Proceedings of IEEE Semiannual Vehicular Technology Conference*, vol. II, pp. 1210–1214, Atlantic City, NJ, October 2001.

[262] B. A. Jones, "Rapid prototyping of wireless communications systems," Master's thesis, Rice University, May 2002.

[263] M. N. Desai and . S. Mangoubi, "Robust Gaussian and non-Gaussian matched subspace detection," *IEEE Transactions on Signal Processing*, vol. 51, pp. 3115–3127, December 2003.

[264] J. M. Leiva-Murillo, R. Satiago-Mozos, S. S. Sanz, and A. Artés-Rodríguez, "Symbol decision via genetic optimization of mutial information," in *The 6th Baiona Workshop on Signal Processing in Communications*, Baiona, Spain, September 2003 (Available at http://www.baionaworkshop.org/Proceedings/leiva.pdf).

[265] X. Huang and N. Phamdo, "Turbo decoders which adapt to noise distribution mismatch," *IEEE Transactions on Communications Letters*, vol. 2, pp. 321–323, December 1998.

[266] S. Verdú, "Spectral efficiency in the wideband regime," *IEEE Transactions on Information Theory*, vol. 48, pp. 1319–1343, June 2002.

[267] S. Verdú, "Second-order asymptotics of mutual information," *IEEE Transactions on Information Theory*, vol. 50, pp. 1567–1580, August 2004.

[268] Z. He and S. K. Mitra, "A unified rate-distortion analysis framework for transform coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, pp. 1221–1236, December 2001.

[269] J. H. Miller and J. B. Thomas, "Detectors for discrete-time signals in non-Gaussian noise," *IEEE Transactions on Information Theory*, vol. 18, pp. 241–250, March 1972.

[270] C. L. Nikias and M. Shao, *Signal Processing with alpha-Stable Distributions and Applications*. New York: John Wiley and Sons, 1995.

[271] A. Janicki and A. Weron, *Simulation and Chaotic Behavior of α-Stable Stochastic Processes*. New York: Marcel Dekker, Inc., 1994.

[272] M. Shao and C. L. Nikias, "Signal processing with fractional lower order moments: Stable processes and their applications," *Proceedings of The IEEE*, vol. 81, pp. 986–1149, July 1993.

[273] A. T. Georgiadis, *Adaptive Equalisation for Impulsive Noise Environments*. PhD thesis, The University of Edinburgh, UK, 2000.

[274] J. M. Chambers, C. L. Mallows, and B. W. Stuck, "A method for simulating stable random variables," *Journal of the American Statistical Association*, vol. 71, pp. 340–344, June 1976.

[275] C. W. J. Granger and D. Orr, ""infinite variance" and research strategy in time series analysis," *Journal of the American Statistical Association*, vol. 67, pp. 275–285, June 1972.

[276] A. T. Georgiadis and B. Mulgrew, "Adaptive Bayesian decision feedback equaliser for alpha-stable noise environments," *Signal Processing*, vol. 81, pp. 1603–1623, 2001.

[277] T. Chuah, B. Sharif, and O. Hinton, "Robust decorrelating decision-deedback multiuser detection in non-Gaussian channels," *Signal Processing*, vol. 81, pp. 1997–2004, 2001.

[278] T. Chuah, B. Sharif, and O. Hinton, "Nonlinear decorrelator for multiuser detection in non-Gaussian impulsive environments," *Electronics Letters*, vol. 36, pp. 920–922, May 2000.

[279] T. Chuah, B. Sharif, and O. Hinton, "Performance of C-filter based adaptive spread spectrum recieverin non-Gaussian channels," *IEE Proc.-Commun.*, vol. 148, pp. 321–326, October 2001.

[280] T. Chuah, B. Sharif, and O. Hinton, "Robust adaptive spread-spectrum receiver with neural-net preprocessing in non-Gaussian noise," *IEEE Transactions on Neural Networks*, vol. 12, pp. 546–558, May 2001.

[281] T. Chuah, B. Sharif, and O. Hinton, "Robust CDMA multiuser detection using a neural-network approach," *IEEE Transactions on Neural Networks*, vol. 13, pp. 1532–1539, November 2002.

[282] S. Yoon, I. Song, and S. Y. Kim, "Code acquisition for DS/SS communications in non-Gaussian impulsive channels," *IEEE Transactions on Communications*, vol. 52, pp. 187–190, February 2004.

[283] M. Rupi, P. Tsakalides, E. D. Re, and C. L. Nikias, "Constant modulos blind equalization based on fractional lower-order statistics," *Signal Processing*, vol. 84, pp. 881–894, 2004.

[284] Y. Xue and X. Zhu, "The minimum error entropy based robust wireless channel tracking in impulsive noise," *IEEE Transactions on Communications Letters*, vol. 6, pp. 228–230, June 2002.

[285] T. Chuah, B. Sharif, and O. Hinton, "Turbo decoding for robust multiuser detection in impulsive noise channels," *Electronics Letters*, vol. 38, pp. 459–461, May 2002.

[286] M. Rupi, P. Tsakalides, E. D. Re, and C. L. Nikias, "Robust spatial filtering of coherent sources for wireless communications," *Signal Processing*, vol. 80, pp. 381–396, 2000.

[287] O. Arikan, A. E. Çetin, and E. Erzin, "Adaptive filtering for non-Gaussian stable processes," *IEEE Signal Processing Letters*, vol. 1, pp. 163–165, November 1994.

[288] G. Aydín, O. Arıkan, and E. Çetin, "Robust adaptive filtering algorithm for α-stable random processes," *IEEE Tr. on Circuits and Systems–II: Analog and Digital Signal Processing*, vol. 46, pp. 198–202, February 1999.

[289] M. H. Kahaei, B. Boashash, and M. Deriche, "Performance analysis of normalized least mean p-norm lattice algorithm for alpha-stable processes," in *Proceedings of 5th Interational Symposium on Signal Processing and Its Applications*, pp. 387–390, Brisbane, Australia, 22-25 August 1999.

[290] J. S. Bodenschatz and C. L. Nikias, "Symmetric alpha-stable filter theory," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2301–2306, September 1997.

[291] M. Belge and E. L. Miller, "A sliding window RLS-like adaptive algorithm for filtering alpha-stable noise," *IEEE Signal Processing Letters*, vol. 7, pp. 86–89, April 2000.

[292] E. Masry, "Alpha-stable signals and adaptive filtering," *IEEE Transactions on Signal Processing*, vol. 48, pp. 3011–3016, November 2000.

[293] S. Kalluri and G. R. Arce, "A general class of nonlinear normalized adaptive filtering algorithms," *IEEE Transactions on Signal Processing*, vol. 47, pp. 2262–2272, August 1999.

[294] S. Kalluri and G. R. Arce, "Adaptive weighted myriad filter algorithms for robust signal processing in α-stable noise environments," *IEEE Transactions on Signal Processing*, vol. 46, pp. 322–334, February 1998.

[295] J. G. Gonzalez and G. R. Arce, "Optimality of the myriad filter in practical impulsive-noise environments," *IEEE Transactions on Signal Processing*, vol. 49, pp. 438–441, February 2001.

[296] R. J. Kozick and B. M. Sadler, "Maximum-likelihood array processing in non-Gaussian noise with Gaussian mixtures," *IEEE Transactions on Signal Processing*, vol. 48, pp. 3520–3535, December 2000.

[297] M. J. Coates and E. E. Kuruoğlu, "Time–frequency-based detection in impulsive noise environments using α-stable noise models," *Signal Processing*, vol. 82, pp. 1917–1925, 2002.

[298] M. Sahmoudi, K. Abed-Meraim, and M. Benidir, "Blind separation of impulsive alpha-stable sources using minimum dispersion criterion," *IEEE Signal Processing Letters*, vol. 12, pp. 281–285, April 2005.

[299] L. M. S. Jose-Revuelta and J. Cid-Sueiro, "A neuro-evolutionary framework for Bayesian blind equalization in digital communications," *Signal Processing*, vol. 83, pp. 325–338, 2003.

[300] K. Hacioglu, "An improved recurrent neural network for M-PAM symbol detection," *IEEE Transactions on Neural Networks*, vol. 8, pp. 779–783, May 1997.

[301] G. Kechriotis and E. S. Manolakos, "Training fully recurrent neural networks with complex weights," *IEEE Transactions on Circuits and Systems - II: Analog and Digital Signal Processing*, vol. 41, pp. 235–238, March 1994.

[302] S. Ong, S. Choi, C. You, and D. Hong, "A complex version of a decision feedback recurrent neural equalizer as an infinite impulse response filter," in *Proceedings of the GLOBECOM '97*, pp. 57–61, 3-8 November 1997.

[303] H. R. Jiang and K. S. Kwak, "On modified complex recurrent neural network adaptive equalizer," *Journal of Circuits, Systems, and Computers*, vol. 11, no. 1, pp. 93–101, 2002.

[304] X. Wang, H. Lin, J. Lu, and T. Yahagi, "Combining recurrent neural networks with self-organizing map for channel equalization," *IEICE Transactions on Communications*, vol. E85-B, pp. 2227–2235, October 2002.

[305] P. H. G. Coelho, "A new state space model for a complex RTRL neural network," in *Proceedings of International Joint Conference on Neural Networks (IJCNN)*, pp. 1756–1761, 15-19 July 2001.

[306] S. Singhal and L. A. Wu, "Training multilayer perceptrons with the extended Kalman algorithm," *Advances in Neural Information Processing Systems 1*, pp. 133–140, San Mateo, CA: Morgan Kaufmann 1989.

[307] P. H. G. Coelho, "A complex EKF-RTRL neural network," in *Proceedings of International Joint Conference on Neural Networks (IJCNN)*, pp. 120–125, 15-19 July 2001.

[308] G. V. Puskorius and L. A. Feldkamp, "Neurocontrol of nonlinear dynamical systems with Kalman filter trained recurrent networks," *IEEE Transactions on Neural Networks*, vol. 5, pp. 279–297, March 1994.

[309] L. A. Feldkamp and G. V. Puskorius, "A signal processing framework based on dynamic neural networks with application to problems in adaptation, filtering and classification," *Proceedings of the IEEE*, vol. 86, pp. 2259–2277, November 1998.

[310] M. Solazzi, A. Uncini, E. D. Di Claudio, and R. Parisi, "Complex discriminative learning Bayesian neural equalizer," *Signal Processing*, vol. 81, pp. 2493–2502, 2001.

[311] T. Kim and T. Adali, "Fully complex multi-layer perceptron network for nonlinear signal processing," *Journal of VLSI Signal Processing*, vol. 32, pp. 29–43, 2002.

[312] E. D. Di Claudio, R. Parisi, and G. Orlandi, "Performance comparison among neural decision feedback equalizers," in *Proc. of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN 2000)*, pp. 361–365, Como, Italy, 24-27 July 2000.

[313] X. Ma and C. L. Nikias, "Parameter estimation and blind channel estimation in impulsive signal environments," *IEEE Transactions on Signal Processing*, vol. 43, pp. 2884–2897, December 1995.

[314] J. Choi, T. H. Yeap, and M. Bouchard, "Online state-space modeling using recurrent multilayer perceptrons with unscented Kalman filter," *Neural Processing Letters*, 2005 (to appear).

[315] R. P. Lippmann and P. Beckman, "Adaptive neural net preprocessing for signal detection in non-Gaussian noise," *Advances in Neural Information Processing Systems 1*, pp. 124–132, San Mateo, CA: Morgan Kaufmann 1989.

[316] P. P. Gandhi and V. Ramamurti, "Neural networks for signal detection in non-Gaussian noise," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2846–2851, November 1997.

[317] G. A. Tsihrintzis and C. L. Nikias, "Performance of optimum and suboptimum receivers in the presence of impulsive noise modeled as an alpha-stable process," *IEEE Transactions on Communications*, vol. 43, pp. 904–914, Feb./Mar./Apr. 1995.

[318] J. Choi, M. Bouchard, and T. H. Yeap, "Decision feedback recurrent neural equalization with fast convergence rate," *IEEE Transactions on Neural Networks*, vol. 16, pp. 699–708, May 2005.

[319] A. Sitz, U. Schwarz, J. Kurths, and H. U. Voss, "Estimation of parameters and unobserved components for nonlinear systems from noisy time series," *Physical Review E*, vol. 66, p. 016210, 2002.

[320] S. Julier and J. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of The IEEE*, vol. 92, pp. 401–422, March 2004.

[321] J. Moon, "The role of SP in data-storage systems," *IEEE Signal Proceesing Mag.*, vol. 15, pp. 54–72, July 1998.

[322] M. Sellathurai and S. Haykin, "Turbo-BLAST for wireless communications: Theory and experiments," *IEEE Transactions on Signal Processing*, vol. 50, pp. 2538–2546, October 2002.

[323] M. Sellathurai and S. Haykin, "Turbo-BLAST: Performance evaluation in correlated Rayleigh-fading environment," *IEEE Journal on Selected Areas in Communications*, vol. 21, pp. 340–349, April 2003.

[324] A. J. Paulraj, D. A. Gore, R. U. Nabar, and H. Bölcskei, "An overview of MIMO communications – A key to gigabit wireless," *Proceedings of The IEEE*, vol. 92, pp. 198–218, February 2004.

[325] S. N. Diggavi, N. Al-Dhahir, A. Stamoulis, and A. R. Calderbank, "Greate expectations: The value of spatial diversity in wireless networks," *Proceedings of The IEEE*, vol. 92, pp. 219–270, February 2004.

[326] A. Goldsmith, S. A. Jafar, N. Jindal, and S. Vishwanath, "Capacity limits of MIMO channels," *IEEE Journal on Selected Areas in Communications*, vol. 21, pp. 684–702, June 2003.

[327] M. Médard, "The effect upon channel capacity in wireless communications of perfect and imperfect knowledge of the channel," *IEEE Transactions on Information Theory*, vol. 46, pp. 933–946, May 2000.

[328] N. B. Mehta, F. F. Digham, A. F. Molisch, and J. Zhang, "Rate of MIMO systems with CSI at transmitter and receiver from pilot-aided estimation," in *Proceedings of IEEE Semiannual Vehicular Technology Conference*, vol. 1, pp. 81–85, Los Angeles, CA, USA, September 2004.

[329] G. Primolevo, O. Simeone, and U. Spagnolini, "Effects of imperfect channel state information on the capacity of broadcast OSDMA-MIMO systems," in *Proceedings of IEEE Workshop on Signal Processing Advances in Wireless Communications*, pp. 546–550, Lisbon, Portugal, July 2004.

[330] T. Yoo and A. J. Goldsmith, "Capacity of fading MIMO channels with channel estimation error," in *Proceedings of Allerton Conference on Communication, Control, and Computing*, October 2003 (Available at http://wsl.stanford.edu/).

[331] T. Yoo and A. J. Goldsmith, "Capacity of fading MIMO channels with channel estimation error," in *Proceedings of IEEE International Conference on Communications*, vol. 2, pp. 808–815, Paris, France, June 2004.

[332] T. Yoo and A. J. Goldsmith, "Capacity and optimal power allocation for fading MIMO channels with channel estimation error," *IEEE Transactions on Information Theory*, Submitted, 2004 (Available at http://wsl.stanford.edu/).

[333] A. Lozano, A. T. Tulino, and S. Verdú, "Multiple-antenna capacity in the low-power regime," *IEEE Transactions on Information Theory*, vol. 49, pp. 2527–2544, October 2003.

[334] J. Choi, M. Bouchard, and T. H. Yeap, "Adaptive filtering-based iterative channel estimation for MIMO wireless communications," in *Proceedings of IEEE International Symposium on Circuits and Systems*, pp. 4951–4954, Kobe, Japan, May 2005.

[335] G. Kitagawa, "Non-Gaussian state-space modeling of nonstationary time sereis," *Journal of the American Statistical Association*, vol. 82, pp. 1032–1063, 1987.

[336] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *Journal of Comput. Graph. Statist.*, vol. 5, no. 1, pp. 1–25, 1996.

[337] W. R. Gilks and C. Berzuini, "Following a moving targetMonte Carlo inference for dynamic Bayesian models," *J. Roy. Statist. Soc., Ser. B*, vol. 63, pp. 127–146, 2001.

[338] J. F. G. de Freitas, M. Niranjan, A. H. Gee, and D. Doucet, "Sequential Monte Carlo methods to train nerual network models," *Neural Computation*, vol. 12, pp. 955–993, 2000.

[339] J. K. Cavers, "An analysis of pilit symbol assisted modulation for Rayleigh fading channels," *IEEE Transactions on Vehicular Technology*, vol. 40, no. 4, pp. 686–693, 1991.

[340] M. Dong, L. Tong, and B. M. Sadler, "Optimal insertion of pilot symbols for transmissions over time-varying flat fading channels," *IEEE Transactions on Signal Processing*, vol. 52, pp. 1403–1418, May 2004.

[341] M. Medard, I. Abu-Faycal, and U. Madhow, "Adaptive coding with pilot signals," in *Proceedings of Allerton Conference on Communication, Control, and Computing*, October 2000.

[342] S. Ohno and G. B. Giannakis, "Average-rate optimal PSAM transmissions over time-selective fading channels," *IEEE Transactions on Wireless Communications*, vol. 1, pp. 712–720, October 2002.

[343] M. Dong and L. Tong, "Optimal design and placement of pilot symbols for channel estimation," *IEEE Transactions on Signal Processing*, vol. 50, pp. 1403–1418, December 2002.

[344] P. Dent, G. E. Bottomley, and T. Croft, "Jakes fading model revisited," *IEE Electronics Letters*, vol. 29, pp. 1162–1163, June 1993.