

國立交通大學

電信工程學系

博士論文

內插信號處理：回音消除，預先編碼，與渦輪等化

Interpolated Signal Processing: Echo Cancellation,

Precoding, and Turbo Equalization

研究生：林壽煦

指導教授：吳文榕

中華民國九十四年六月

內插信號處理：回音消除，預先編碼，與渦輪等化

Interpolated Signal Processing: Echo Cancellation,

Precoding, and Turbo Equalization

研究生：林壽煦

Student: Shou-Sheu Lin

指導教授：吳文榕 博士

Advisor: Dr. Wen-Rong Wu

國立交通大學

電信工程學系

博士論文



Submitted to Department of Communication Engineering

College of Electrical Engineering and Computer Science

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

in

Communication Engineering

June 2005

Hsinchu, Taiwan, Republic of China

中華民國九十四年六月

# 內插信號處理：回音消除，預先編碼，與渦輪等化

研究生：林壽煦

指導教授：吳文榕

國立交通大學電信工程學系

## 摘要

在有線通訊中，回音消除、預先編碼、與等化是常用的信號處理運算。對於如數位用戶迴路(DSL)等之高速通訊系統，通道響應通常是非常的長，因此前述的信號處理計算複雜度需求也是相當的高。在本論文中，我們提出了一種基於內插濾波的高效演算法以降低複雜度運算量。內插濾波是由一個有限脈衝響應(FIR)濾波器、一個內插有限脈衝響應濾波(IFIR)與一種將上述兩個濾波器係數重疊的方法串聯實作而成。內插濾波方法可有效的降低計算複雜度同時保有傳統有限脈衝響應濾波器的數值穩定性優點與性能。

基於所提出的內插濾波器架構，我們將其廣泛運用於回音消除、決策回授等化(DFE)、與 Tomlinson-Harashima 預先編碼(THP)等信號處理功能。在論文中，我們對所提濾波器架構做了完整的理論分析，相關的理論公式如最佳解與輸出訊號雜訊比(SNR)等也提供了詳細的推導。為了適應通道變化與降低實作複雜度，我們使用最小平均平方(LMS)法作為適應性演算法。對於所提之適應性演算法，其收斂行為與理論效能，我們也做了完整的理論分析與驗證。論文所提出的適應性內插演算法，可以較低的計算複雜度達到傳統演算法相同的效能。對內插回音消除而言，模擬結果顯示在各種單迴路高速用戶迴路(SHDSL)拓撲下，可消除 73.0 分貝(dB)的回音同時可節省 57%的複雜度。對內插決策回授等化、與內插預先編碼，複雜度節省則可高達 76%。

渦輪等化是一種結合等化與錯誤更正解碼的重複等化方法，其效能遠超越傳統的分離式等化與錯誤更正解碼方法。然而，前者的複雜度卻遠超過後者。以往的研

究主要集中在無線方面的運用。因無線通道的響應長度是屬於較短或稀疏的，以格子(trellis)為基礎的演算法如軟輸出 Viterbi 演算法或 BCJR 演算法可有效的運用於無線系統。很不幸地，此類演算法的複雜度與通道的響應長度成指數成長。針對此一問題，Tüchler 等人於西元 2002 年提出了以濾波器為架構的低複雜度渦輪等化器。雖然複雜度可以大幅度的降低，但當通道的響應長度很長時，複雜度還是相當的高。由於數位用戶迴路的通道的響應長度通常長達數百，到目前為止，尚無可行的渦輪等化器可供使用。

基於內插濾波的概念，我們提出一個可應用於長通道的快速渦輪等化器。該快速等化器可將複雜度降低十倍以上。在Tüchler的渦輪等化器中，最佳濾波器係數運算與等化濾波運算的計算量相當的大。我們以理論證明，不同的最佳濾波器是可以被內插的。我們僅需事先計算好少數的最佳濾波器係數，之後便可以內插的方式來快速計算出最佳濾波器係數。因此，複雜度便可大幅度的降低。如果最佳濾波器響應或通道響應本身也是可以被內插的，則我們可運用IFIR方式再次減低複雜度。就我們所知，我們所提出的快速渦輪等化器是目前唯一可應用於SHDSL系統也是世界上複雜度最低的渦輪等化器。在數位位元錯誤率(BER)得等於  $10^{-5}$ 時，我們可用四次重複等化方式得到 8.8 分貝的效能改進，而複雜度僅需Tüchler渦輪等化器的 3.7%。就整體複雜度而言，所提出的快速渦輪等化器大約為傳統分離式等化與錯誤更正解碼方法的三倍，但這已代表所提的方法已可達到實際應用的目標。

# Interpolated Signal Processing: Echo Cancellation, Precoding, and Turbo Equalization

Student: Shou-Sheu Lin

Advisor: Dr. Wen-Rong Wu

Department of Communication Engineering  
National Chiao Tung University

## Abstract

Echo cancellation, precoding, and equalization are common signal processing operations performed in wireline communications. For high-speed systems such as digital subscriber line (DSL), the channel response is usually very long and the computational complexity requirement for those operations is very high. In this thesis, efficient algorithms based on interpolated filtering are developed to solve the problem. The idea of interpolated filtering is realized with a cascade of an FIR and an interpolated FIR (IFIR) filter, and with a tap-weight overlapping method. The interpolated filtering scheme can effectively reduce the computational complexity while inherits all the numerical stability advantages of the conventional FIR filter.

The interpolated filtering framework is then applied to echo cancellation, decision feedback equalization (DFE), and Tomlinson-Harashima precoding (THP). Performance is theoretically analyzed and close-form solutions such as optimum solutions and output signal to noise ratio (SNR) are derived also. For accommodating the channel variation and reducing implementation complexity, adaptive algorithms based on the least-mean-squared (LMS) algorithm are then considered. Convergence behavior is analyzed and the performance is theoretically evaluated. While proposed adaptive interpolated algorithms can achieve similar performance as conventional algorithms, the computational complexity is much lower. For echo cancellation, simulations with a wide variety of loop topologies show that the interpolated echo canceller can effectively cancel the echo up to 73.0 dB and achieve 57% complexity saving (in SHDSL applications). For DFE and THP, the computational saving can be as high as 76%.

It is well known that a turbo equalizer, a joint iterative equalization and decoding scheme, can significantly outperform a conventional receiver performing equalization and decoding separately. However, the complexity is much higher than the conventional receiver. Previous works mainly focus on the wireless applications in which the channel length is short or sparse. This enables the use of the trellis-based algorithms such as soft-output Viterbi algorithm (SOVA) and BCJR. Unfortunately, the computational complexity of these algorithms grows exponentially with the channel length. In 2002, Tüchler et al. proposed a low-complexity filter-based turbo equalizer reducing the complexity dramatically. Even so, the computational complexity remains tremendous if the channel length is long. So far, there is no turbo equalizer with reasonable complexity designed for a channel with hundreds of taps, which is common in DSL applications.

With the interpolated filtering approach, a fast turbo equalizer with complexity an order of magnitude less is proposed. The most computationally intensive operations in Tüchler's equalizer are the calculation of optimal filter coefficients and the filtering operation of the equalizer. The relationship between optimal filter coefficients and reliability information is exploited and a fast algorithm, which calculates the current optimal coefficients by interpolating two pre-calculated known optimal coefficients, is proposed. If the channel response has a smooth shape, the interpolated equalizer can be applied to reduce the complexity further. Closed-form expressions for interpolated optimal filter coefficients are also derived. To the best of our knowledge, the computational complexity of the proposed turbo equalizer (for SHDSL application) is the lowest in the world. The performance gain at BER  $10^{-5}$  is about 8.8 dB (with four iterations) and the complexity is only 3.7% of the conventional turbo equalization. Also, the complexity is less than three times of the conventional un-turbo equalizer scheme. This indicates that the complexity of the proposed turbo equalizer is lower enough such that real-world implementation becomes feasible.

# Acknowledgements

First of all, I would like to thank my advisor, Professor Wen-Rong Wu for his extensive guidance, valuable suggestions, and constructive discussions to make this dissertation possible. I really appreciate his time and efforts in improving my papers, thesis and writing skills. I did learn a lot from him not only in the academic works but also his attitude to everything. I consider myself to be truly fortunate to have had the opportunity to work with such a knowledgeable and friendly person.

I am extremely grateful to all the members of my PhD dissertation committee for their thoughtful comments and valuable suggestions. With their comments, the content of dissertation is further improved and more direction about future works is inspired.

I wish to thank Jeff Lee, Hua-Lung Yang and Yinman Lee for their discussion and help. I am extremely grateful to Chao-Yuan Hsu for his extensive support. His in-depth discussion and help saves me much times for documentation preparation and presentation.

I would like to special thanks to my elder brother Jun-Fu Lin for his inspiration to undertake the Ph.D. studies. Without his motivation, I will never think about pursuing a doctoral degree after graduate.

I am deeply indebted to my beloved wife, Sun-Ting Lin, for her support and understanding. I would like to thank my lovely daughter, Rui-Yu Lin, for her understanding. I express my deep gratitude to my mother-in-law, Fu-Hei Chen, and aunt, Siu-Er Dai, and her family for taking care of my daughter. I wish to thank to my grandma, Moon Hsu, for taking care of my daily

life. Just like her name, she is like the Moon and always lighting my life. I feel so lucky to be his grandson. There are so many peoples, I should thank. Finally, I would like to thank to all relatives and friends who ever encourage or help me.





# Contents

|  |  |             |
|--|--|-------------|
| <i>Abstract</i>  |  | <b>iii</b>  |
| <i>Acknowledgements</i>  |  | <b>v</b>    |
| <i>Contents</i>  |  | <b>vii</b>  |
| <i>List of Tables</i>  |  | <b>xi</b>   |
| <i>List of Figures</i>   |  | <b>xiii</b> |
| <i>Glossary</i>  |  | <b>xvii</b> |
| <b>1</b> <i>Introduction</i>                                   |  | <b>1</b>    |
| <b>2</b> <i>Digital Subscriber Loop Environment</i>            |  | <b>5</b>    |
| <b>3</b> <i>Interpolated Echo Canceller</i>                    |  | <b>11</b>   |
| 3.1 The IFIR Echo Canceller . . . . .                          |  | 14          |
| 3.2 Theoretical Analysis . . . . .                             |  | 18          |
| 3.3 Optimal Interpolation Filter Design . . . . .              |  | 21          |
| 3.4 Simulation Results . . . . .                               |  | 26          |
| 3.4.1 Loop Characteristics and Topologies . . . . .            |  | 26          |
| 3.4.2 The Cutting Point and the Interpolation Factor . . . . . |  | 29          |



|          |   |           |
|----------|---|-----------|
| 3.4.3    | Noise Environments: AWGN and NEXT . . . . .                         | 31        |
| 3.4.4    | Discussions . . . . .   | 33        |
| 3.5      | Conclusions . . . . .   | 38        |
| <b>4</b> | <b><i>Interpolated Decision Feedback Equalizer and Precoder</i></b> | <b>39</b> |
| 4.1      | System Model and Conventional DFE . . . . .                         | 42        |
| 4.2      | The Proposed Interpolated DFE . . . . .                             | 45        |
| 4.2.1    | Interpolated DFE . . . . .  | 45        |
| 4.2.2    | Adaptive IDFE . . . . .   | 51        |
|          | A) Convergence in the Mean . . . . .                                | 51        |
|          | B) Convergence in the MSE . . . . .                                 | 52        |
| 4.3      | Simulation Results . . . . .  | 55        |
| 4.3.1    | Channel CSA No. 3 . . . . .   | 55        |
| 4.3.2    | Different Loop Topologies . . . . .                                 | 56        |
| 4.3.3    | The Adaptive IDFE . . . . .   | 57        |
| 4.3.4    | Symbol Error Rate vs. SNR (for DFE and THP) . . . . .               | 58        |
| 4.3.5    | Discussions . . . . .   | 59        |
| 4.4      | Conclusions . . . . .   | 61        |
| <b>5</b> | <b><i>Fast Interpolated Turbo Equalizer</i></b>                     | <b>63</b> |
| 5.1      | System Model and Notations . . . . .                                | 67        |
| 5.2      | Summary of Previous Works . . . . .                                 | 70        |
| 5.2.1    | The BCJR Equalizer/Decoder . . . . .                                | 70        |
|          | A) Trellis Diagram Representation of an ISI Channel . . . . .       | 70        |
|          | B) Trellis Diagram Representation of a Convolutional Code . . . . . | 71        |
| 5.2.2    | The BCJR Equalizer . . . . .  | 73        |
| 5.2.3    | The BCJR Decoder . . . . .  | 75        |
| 5.2.4    | The BCJR Turbo Equalizer . . . . .                                  | 76        |



## CONTENTS

ix

|          |   |            |
|----------|---|------------|
| 5.2.5    | Filter-based MMSE SISO Equalizer . . . . .  | 80         |
|          | A) OSL Equalizer . . . . .  | 81         |
|          | B) LSL Equalizer . . . . .  | 88         |
|          | C) LSLN, LSLP, and LSLH Equalizers . . . . .                                      | 90         |
| 5.3      | The Proposed Fast Interpolated Turbo Equalizer . . . . .                          | 94         |
|          | 5.3.1 FSISL Equalizer . . . . .   | 94         |
|          | 5.3.2 FDISL Equalizer . . . . .   | 102        |
|          | 5.3.3 Complexity Analysis . . . . .   | 106        |
| 5.4      | Simulation Results . . . . .  | 109        |
|          | 5.4.1 Parameters of Reference Filters . . . . .                                   | 112        |
|          | 5.4.2 Parameters for IR Interpolation . . . . .                                   | 113        |
|          | 5.4.3 BER Simulations . . . . .   | 115        |
|          | A) Low-ISI Proakis A Channel . . . . .  | 115        |
|          | B) Severe-ISI Proakis C Channel . . . . .   | 116        |
|          | C) Long ISI CSA #6 Channel . . . . .  | 117        |
|          | 5.4.4 Discussions . . . . .   | 121        |
| 5.5      | Conclusions . . . . .   | 122        |
| <b>6</b> | <b><i>Conclusions and Future Works</i></b> . . . . .                              | <b>125</b> |
| <b>A</b> | <b><i>Derivation of the Correlation Matrices for DFE</i></b> . . . . .            | <b>129</b> |
| <b>B</b> | <b><i>Derivation of the Correlation Matrices for IDFE</i></b> . . . . .           | <b>133</b> |
| <b>C</b> | <b><i>Derivation of the BCJR Algorithm for Optimal Equalization</i></b> . . . . . | <b>139</b> |
|          | C.1 The BCJR Algorithm (Probabilistic Form) . . . . .                             | 144        |
|          | C.2 The BCJR Algorithm (Logarithmic Form or Log-MAP) . . . . .                    | 147        |
|          | C.3 The BCJR algorithm (Max-log-MAP) . . . . .                                    | 149        |

|          |  |            |
|----------|--|------------|
| <b>D</b> | <b><i>Derivations of Soft-input and Soft-output for BPSK and 4-PAM</i></b> | <b>151</b> |
| D.1      | 2-PAM (BPSK) . . . . .   | 151        |
| D.2      | 4-PAM . . . . .  | 152        |
| <b>E</b> | <b><i>Complexity Analysis</i></b>  | <b>155</b> |
|          | <b><i>Bibliography</i></b>   | <b>161</b> |



# List of Tables

|     |   |     |
|-----|---|-----|
| 3.1 | Computational complexity comparison for the FIR and IFIR echo cancellers . . .  | 18  |
| 3.2 | ERLE performance vs. various interpolation filters (LIN: linear, TRS: truncated sinc, HWS: Hanning-windowed sinc, CWS: Chebyshev-windowed sinc, OPS: optimal-symmetric, OPT: optimal) . . . . . | 25  |
| 3.3 | Computational complexity ratio for different interpolation factor ( $N_1 = 50, S=2$ )   | 31  |
| 3.4 | Echo return loss and path loss for CSA loops . . . . .  | 34  |
| 3.5 | SNR with or without echo interference for CSA #1 at CPE side . . . . .  | 37  |
| 4.1 | Computational complexity analysis . . . . .   | 53  |
| 5.1 | Soft-input and soft-output conversion formulae for M-PAM . . . . .  | 86  |
| 5.2 | Complexity of SISO equalizers without SISO conversion . . . . .   | 107 |
| 5.3 | Channel models for performance evaluation . . . . .   | 111 |
| D.1 | Parameters for calculating the soft-input (4-PAM) . . . . .   | 153 |
| D.2 | Parameters for calculating the extrinsic information (4-PAM) . . . . .  | 154 |
| E.1 | Complexity of FDISL equalizer . . . . .   | 156 |
| E.2 | Complexity of LSL equalizer . . . . .   | 157 |
| E.3 | Complexity of BCJR equalizer . . . . .  | 158 |
| E.4 | Complexity of BCJR decoder . . . . .  | 159 |
| E.5 | Complexity summary of the turbo equalizers . . . . .  | 159 |

E.6 Complexity of SISO equalizers without SISO conversion . . . . . 160



# List of Figures

|      |   |    |
|------|---|----|
| 2.1  | The network architecture of digital subscriber loop - the last mile. . . . .      | 6  |
| 2.2  | A typical DSL loop with different size of twisted-pair wires and bridge taps. . . | 7  |
| 2.3  | Full-duplexing transmission of a DSL channel. . . . .                             | 8  |
| 2.4  | The echo caused by the impedance mismatch of a hybrid circuit. . . . .            | 9  |
| 3.1  | A typical echo response. . . . .  | 12 |
| 3.2  | The adaptive FIR echo canceller. . . . .  | 12 |
| 3.3  | The adaptive IFIR echo canceller. . . . .   | 15 |
| 3.4  | The filter responses of the IFIR echo canceller. . . . .                          | 17 |
| 3.5  | Illustration of finding optimal interpolation filter. . . . .                     | 22 |
| 3.6  | Interpolation filters ( $M=4, S=2$ ). . . . .                                     | 26 |
| 3.7  | SHDSL echo responses for CSA loops at CO side. . . . .                            | 27 |
| 3.8  | SHDSL echo responses for CSA loops at CPE side. . . . .                           | 28 |
| 3.9  | Overlapping of FIR and IFIR filters ( $M=4, S=2, N_1=50$ ). . . . .               | 29 |
| 3.10 | ERLE performance for different CSA loops. . . . .                                 | 30 |
| 3.11 | ERLE versus the cutting point and the interpolation factor. . . . .               | 32 |
| 3.12 | Simulated ERLE performance at CO side. . . . .                                    | 33 |
| 3.13 | Simulated ERLE performance at CPE side. . . . .                                   | 34 |
| 3.14 | Total echo return loss at CPE side. . . . .                                       | 35 |
| 3.15 | The power spectral densities of echo, residual echo, and composite noises. . . .  | 36 |

|      |  |    |
|------|--|----|
| 3.16 | SNR (with echo cancellation) at CPE receiver. . . . .  | 37 |
| 4.1  | Typical channel impulse response in digital subscriber loop application. . . . .   | 41 |
| 4.2  | Conventional adaptive DFE structure. . . . .   | 43 |
| 4.3  | (a) Optimal feedback filter response of conventional DFE. (b) Interpolated feedback filter response of proposed IDFE. . . . .  | 47 |
| 4.4  | Proposed low-complexity adaptive IDFE structure. . . . .   | 48 |
| 4.5  | Complexity ratio with respect to a conventional DFE, $(N_f, N_b)=(16,180)$ . The minimum complexity ratio (C.R.) is 24% when the interpolation factor equals 8. . . . .                  | 54 |
| 4.6  | (a) Optimal feedforward filter coefficients for IDFE and DFE. (b) Optimal feedback filter coefficients for IDFE and DFE. . . . .   | 56 |
| 4.7  | (a) Performance of DFE and IDFE for eight CSA loops at received SNR=40 dB. (b) Performance of DFE and IDFE for eight CSA loops at received SNR=20 dB. . . . .                            | 57 |
| 4.8  | Learning curves of adaptive IDFE and DFE with step size 0.000638. Theoretical steady-state MSE ( $J_\infty$ ), MMSE ( $J_{\min}$ ), and misadjustment ( $\psi$ ) are also shown. . . . . | 59 |
| 4.9  | Low-complexity interpolated THP (associated with adaptive IDFE) . . . . .  | 60 |
| 4.10 | Symbol error rate comparison for adaptive IDFE plus ITHP and DFE plus THP (without error propagation), and for adaptive IDFE and DFE (with error propagation). . . . .                   | 61 |
| 5.1  | The system model of turbo equalization for coded data transmitted over ISI channel. . . . .  | 68 |
| 5.2  | A discrete-equivalent ISI channel with memory length $\mu$ . . . . .   | 71 |
| 5.3  | The state transition diagram for the ISI channel shown in Fig. 5.2, where $Q = M^\mu$ is total numbers of states for the channel. . . . .  | 72 |
| 5.4  | A half rate convolutional encoder with memory length of $\eta$ . . . . .   | 72 |



|      |   |     |
|------|---|-----|
| 5.5  | The state transition diagram of the convolutional encoder shown in Fig. 5.4, where $P = 2^n$ is total numbers of states for the code. . . . . | 73  |
| 5.6  | A generic SISO processing unit. . . . .   | 77  |
| 5.7  | The block diagram of the turbo equalizer. . . . .   | 77  |
| 5.8  | A filter-based MMSE SISO linear equalizer. . . . .  | 81  |
| 5.9  | The optimal block time-invariant filter vs. the iteration and block number. . . . .   | 95  |
| 5.10 | The average reliability function vs. the iteration and block numbers. . . . .   | 96  |
| 5.11 | Optimal block time-invariant filter vs. the reliability function (Channel: Proakis B channel with 4-PAM at SNR=18.0 dB). . . . .              | 97  |
| 5.12 | Optimal block time-invariant filter vs. the reliability function (Channel: Proakis B channel with 4-PAM at SNR=18.0 dB). . . . .              | 98  |
| 5.13 | The detailed filter structure of the LSL. . . . .   | 103 |
| 5.14 | The reference optimal filters for the SHDSL CSA #6 channel at SNR=18.0 dB. . . . .  | 105 |
| 5.15 | The IR complexity ratio vs. interpolation factor for the SHDSL CSA #6 channel. . . . .  | 109 |
| 5.16 | The computational complexity of proposed fast interpolated SISO linear equalizers (FSISL and FDISL) and LSL equalizer. . . . .                | 110 |
| 5.17 | The complexity ratio of proposed turbo equalizers (block length: 512 symbols). . . . .  | 111 |
| 5.18 | The channel response of CSA #6 channel for SHDSL application. . . . .   | 112 |
| 5.19 | The zoom-in view of the reference optimal filters ( $Z = 15, \lambda=8$ , SHDSL CSA #6 channel, and SNR=18.0 dB). . . . .                     | 114 |
| 5.20 | The NIE performance of reference optimal filters in Fig. 5.18, where the worst one is -39.7 dB and the average is -44.3 dB. . . . .           | 115 |
| 5.21 | The IR interpolation for the CSA #6 channel response and the reference optimal filters in Fig. 5.18. . . . .                                  | 116 |
| 5.22 | The interpolation filter used for IR interpolation in Fig. 5.20. . . . .  | 117 |
| 5.23 | BER comparison for the proposed FSISL (dashed-x) and LSL (solid) turbo equalizers over Proakis A channel . . . . .                            | 118 |

5.24 BER comparison for the proposed FSISL (dashed-x) and LSL (solid) turbo equalizers over Proakis C channel. . . . . 119

5.25 BER comparison for the proposed FSISL (dashed-x), FDISL (dotted-point), and LSL (solid) turbo equalizers over SHDSL CSA #6 channel . . . . . 120

5.26 BER comparison for the proposed FSISL (dashed-x), FDISL (dotted-point), and LSL (solid) turbo equalizers over SHDSL CSA #6 channel (the LLR is clipped). 122



# Glossary

**ADSL** Asymmetric Digital Subscriber Line

**ADSL2** Asymmetric Digital Subscriber Line - Second Generation

**ADSL2+** Extended Bandwidth ADSL2

**ANSI** American National Standards Institute

**APP** A Posteriori Probability

**ARMA** Autoregressive and Moving Average

**AWGN** Additive White Gaussian Noise

**BCJR** The Optimal MAP Algorithm by L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv

**BICM** Bit-Interleaved Coded Modulation

**BER** Bit-Error Rate

**BPSK** Binary Phase Shift Keying

**CDMA** Code-Division Multiple Access

**CO** Central Office

**CPE** Customer Premise Equipment



|              |   |
|--------------|---|
| <b>CSA</b>   | Carrier Service Area                            |
| <b>DFE</b>   | Decision Feedback Equalizer                     |
| <b>DMT</b>   | Discrete Multi-Tone                             |
| <b>DS</b>    | Downstream                                      |
| <b>DSL</b>   | Digital Subscriber Line                         |
| <b>ERL</b>   | Echo Return Loss                                |
| <b>ERLE</b>  | Echo Return Loss Enhancement                    |
| <b>ETSI</b>  | European Telecommunications Standards Institute |
| <b>EXIT</b>  | EXtrinsic Information Transfer                  |
| <b>FBF</b>   | Feedback Filter                                 |
| <b>FDD</b>   | Frequency Division Duplex                       |
| <b>FFF</b>   | Feedforward Filter                              |
| <b>FFT</b>   | Fast Fourier Transform                          |
| <b>FIR</b>   | Finite Impulse Response                         |
| <b>FS</b>    | Fractionally-Spaced                             |
| <b>FSISL</b> | Fast Singly Interpolated SISO Linear            |
| <b>FDISL</b> | Fast Doubly Interpolated SISO Linear            |
| <b>GSM</b>   | Global System for Mobile Communications         |
| <b>HDSL</b>  | High Bit Rate Digital Subscriber Line           |



|               |  |
|---------------|--|
| <b>HDSL2</b>  | High Bit Rate Digital Subscriber Line - 2nd Generation |
| <b>IC</b>     | Interference Cancellation                              |
| <b>IDFE</b>   | Interpolated Decision-Feedback Equalizer               |
| <b>IFIR</b>   | Interpolated Finite Impulse Response                   |
| <b>i.i.d.</b> | Independent and Identically Distributed                |
| <b>IIR</b>    | Infinite Impulse Response                              |
| <b>IR</b>     | Individual Response                                    |
| <b>ISDN</b>   | Integrated Service Digital Network                     |
| <b>ITHP</b>   | Interpolated Tomlinson-Harashima precoder              |
| <b>ITU</b>    | International Telecom Union                            |
| <b>ISI</b>    | Intersymbol Interference                               |
| <b>LDPC</b>   | Low-Density Parity-Check Codes                         |
| <b>LLR</b>    | Log-Likelihood Ratio                                   |
| <b>LMS</b>    | Least Mean Square                                      |
| <b>LS</b>     | Least-Squares  |
| <b>LSL</b>    | Low-complexity SISO Linear                             |
| <b>LSLH</b>   | Low-complexity SISO Linear, Hybrid                     |
| <b>LSLN</b>   | Low-complexity SISO Linear, No <i>a priori</i>         |
| <b>LSLP</b>   | Low-complexity SISO Linear, Perfect <i>a priori</i>    |

|             |  |
|-------------|--|
| <b>MAP</b>  | Maximum A Posteriori                       |
| <b>MFB</b>  | Matched-Filter Bound                       |
| <b>ML</b>   | Maximum Likelihood                         |
| <b>MMSE</b> | Minimum Mean Square Error                  |
| <b>MSE</b>  | Mean Square Error                          |
| <b>NIE</b>  | Normalized Interpolated Error              |
| <b>NSC</b>  | Non-Systematic Convolutional Code          |
| <b>OFDM</b> | Orthogonal Frequency Division Multiplexing |
| <b>OSL</b>  | Optimal SISO Linear                        |
| <b>PAM</b>  | Pulse Amplitude Modulation                 |
| <b>pdf</b>  | Probability Density Function               |
| <b>PL</b>   | Path Loss                                  |
| <b>PMP</b>  | Pontryagin's Maximum Principle             |
| <b>POTS</b> | Plain Old Telephone Service                |
| <b>PSD</b>  | Power Spectral Density                     |
| <b>PSK</b>  | Phase Shift Keying                         |
| <b>PSTN</b> | Public Switched Telephone Network          |
| <b>QAM</b>  | Quadrature Amplitude Modulation            |
| <b>RFI</b>  | Radio Frequency Interference               |



|               |  |
|---------------|--|
| <b>SC-FDE</b> | Single-Carrier Frequency-Domain Equalization                   |
| <b>SDSL</b>   | Symmetrical Single Pair High Bitrate Digital Subscriber Line   |
| <b>SHDSL</b>  | Single-Pair High-Speed Digital Subscriber Line                 |
| <b>SISO</b>   | Soft-Input Soft-Output   |
| <b>SNR</b>    | Signal to Noise Ratio  |
| <b>SOVA</b>   | Soft Output Viterbi Algorithm                                  |
| <b>T1</b>     | ANSI 1.544 Mbps time-division multiplex service                |
| <b>TDD</b>    | Time Division Duplex   |
| <b>telco</b>  | Telecom Company  |
| <b>TERL</b>   | Total Echo Return Loss   |
| <b>THP</b>    | Tomlinson-Harashima Precoder                                   |
| <b>US</b>     | Upstream   |
| <b>VA</b>     | Viterbi Algorithm  |
| <b>VDSL</b>   | Very-High-Bit-Rate Digital Subscriber Line                     |
| <b>VDSL2</b>  | Very-High-Bit-Rate Digital Subscriber Line - Second Generation |
| <b>WR</b>     | Whole Response   |
| <b>xDSL</b>   | Abbreviation for any kind of DSL technology                    |







# Chapter 1

## Introduction

**S**o far, the twisted-pair copper wire is known to be the most popular broadband access media in the world. Several digital subscriber line (DSL) technologies have been proposed and successfully commercialized. With these technologies, the broadband internet access era has been becoming reality. While the DSL transmission rate is greatly enhanced, it is still far from Shannon's limit. Sophisticated communication systems and advanced signal processing schemes are still continuously developed.

In the DSL environment, there exists a couple of major impairments that impose challenges for receiver signal processing. This includes echo cancellation, channel equalization, and precoding. To achieve full duplex transmission over a single pair of wire, a hybrid circuit is employed in the transmitter. However, echoes are produced due to the impedance mismatch problem. The echo will significantly interfere the receive signal. In many cases, the echo interference is so strong and the receiver becomes impossible to work properly. A common remedy to this problem is to use an echo canceller. The echo channel response is usually very long and this puts a computational complexity burden for the receiver. Originally, the twisted-pair channel is used for voice transmission and the bandwidth is narrow. It will introduce a severe intersymbol interference (ISI) for high speed data transmission. In DSL, it is not uncommon to observe a channel with hundreds taps. Channel equalization and precoding are the techniques

to combat this problem. Similar to the echo canceller, the computational complexities of these devices are very high. Though above techniques are suggested in standards and popular in the off-the-shelf DSL related products, the computational complexity due to the long channel response remain an issue, specially when the transmission rate is further increased.

In this dissertation, we will focus on the complexity reduction issue in echo cancellation, channel equalization, and precoding. We explore the DSL channel characteristics and propose effective signal processing schemes. Our goal is to develop algorithms with implementable complexity for the next generation DSL applications. There are two distinct properties in the DSL environment. The first one is that the channel and echo responses are almost static. This means that many signal processing structures and parameters can remain the same for a long period of time and this is a great design advantage. The other property is that the channel and echo responses exhibits low-pass characteristics. This indicates that even though the channel or echo response is long, we can use a low-order system to model it. Exploiting these two properties, we develop our low-complexity algorithms through a general framework of interpolation. To guarantee the stability, we use the finite impulse response (FIR) structure for the interpolated filters. With our interpolated signal processing schemes, the computational complexity can be significantly lower than the conventional approaches.

This dissertation is organized as follows. The DSL environments and impairments are briefed in Chapter 2. In Chapter 3, we investigate the echo cancellation problem and propose a low-complexity adaptive interpolated FIR (IFIR) echo canceller. Theoretical Wiener solution, minimum mean square error (MMSE) and convergence behaviors are analyzed in detail. We also propose a least-squares method to design the optimal interpolation filter. These results were verified with extensive simulations with versatile loop topologies and noise environments. Base on the similar IFIR filtering structure, we then extend the interpolation scheme to channel equalization and precoding. This is described in Chapter 4 in which we propose a low-complexity adaptive interpolated decision feedback equalizer (DFE) and a low-complexity interpolated Tomlinson-Harashima precoder. Theoretical Wiener solutions, minimum mean

square errors (MMSEs) and convergence behaviors are also analyzed in details. In Chapter 5, we study the turbo equalization algorithm. In this chapter, we propose fast interpolated turbo equalizers with complexity an order of magnitude less than the conventional approach. An actual DSL channel is used for simulations and the results show that the performance is similar to the conventional approach. Finally, we draw our conclusions in Chapter 6.





## Chapter 2

# Digital Subscriber Loop Environment

The invention of telephone is a remarkable milestone in the contemporary history of human beings. With the telephone network, people can communicate each other without any distance limitation. Lately, the modem was introduced and digital data can be transmitted/received over the network. With digital processing technology, voice, text, music, picture, and video can all be converted into their digital forms. In addition to voice, today's telephone network extends its application to multimedia communication.

As shown in Fig. 2.1, the telephone network consists of a core network and an access network. The core network, shown on the left-side of the figure, relays information from the central office (CO) and then conducts it to the different service networks such as circuit-switched public switched telephone network (PSTN) for voice communication or packet-routed data network for Internet access. The access network, shown on the right-hand side, transports information at the customer premise (CPE) side to the CO side. The distance between the CPE and the CO is usually around one mile. Thus, we also call it as the last mile. For the PSTN, any user within the telephone network needs a dedicated physical link. The media cost becomes an important issue when the access network is built. For low-cost consideration, the twisted-pair wire was chosen as the transmission media for the last mile.

The telephone network was initially designed for the voice communication only, i.e., the so-

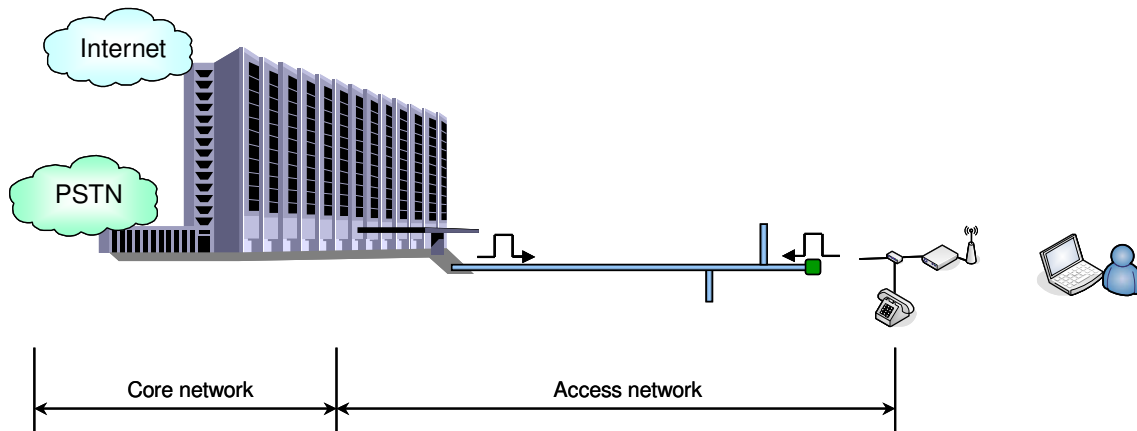


Figure 2.1: The network architecture of digital subscriber loop - the last mile.

called plain old telephone service (POTS). The voice signal is transmitted directly on a single pair of twisted-pair wire in its analog form. The bandwidth requirement for POTS is set as 4 KHz. The signal transmission between two end users during a phone call can be described as follows. A 4 KHz analog voice signal is first transmitted from a CPE to a CO. Then, it is sampled and digitized at 8 KHz rate before entering a class 5 switch (a voice switch) [1]. After the class 5 switch, the voice signal is transported and switched digitally within the core network of the PSTN and finally converted back to the original analog waveform in a remote class 5 switch near the called user. Due to the sampling operation, the class 5 switch limits the bandwidth utilization of a twisted-pair wire. As we can see, the signal transmission path can be divided into three sections. They are a twisted-pair wire section from the calling user to the first class 5 switch, a digital link section between the first and the last class 5 switch, and finally a twisted-pair wire section from the last class 5 switch to the called user. Note that the bandwidth of the twisted-pair wire is wider than 4 kHz and is capable of carrying more information. The digital section is the throughput/bandwidth bottleneck in a telephone network.

Though the original purpose of the twisted-pair wire is for voice communication, it's possible to extend its use for other types of application. For example, fax uses the digital modulation technology such as quadrature-amplitude-modulation (QAM) to transmit digital data. However,

due to the sampling constraint in the class 5 switch, the bandwidth utilization of a twisted-pair wire for digital communication is still limited to 4 KHz. For higher speed communication requirement such as Internet access, the bandwidth is not sufficient. To solve the problem, the class 5 switch is bypassed and transmission characteristic of the twisted-pair wire above 4 KHz was exploited. This results in the development of new systems such as xDSLs (ISDN, HDSL, SHDSL, ADSL and VDSL) pumping more throughput with wider bandwidth utilization. Note that a new type of switch is required at CO side to handle the new applications. Using this approach, the bottleneck in the digital section is resolved.

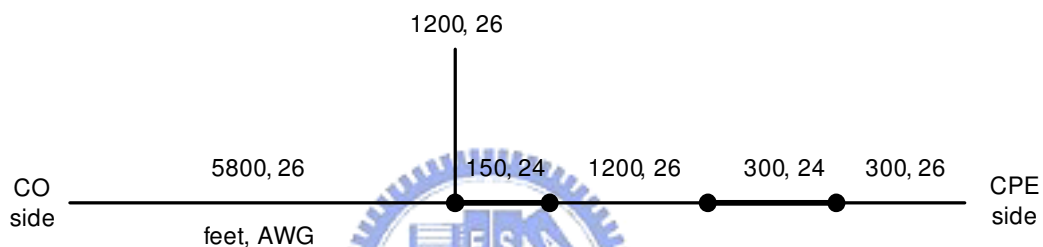


Figure 2.2: A typical DSL loop with different size of twisted-pair wires and bridge taps.

A DSL loop may consist of twisted-wire pairs with different sizes. A typical DSL loop is shown in Fig. 2.2, in which each segment of the twisted-pair wire is labeled with a length in feet and a diameter in gauge number. A smaller gauge number stands for a wire with larger diameter. Since the telephone company built the network before user subscription, it doesn't know who will connect to a particular DSL loop in advance. The bridge taps provide access points for potential users in that service area. Whenever the wire gauge is changed or a bridge tap is placed, impedance mismatch will occur and this will result in signal reflection. When the signal is transmitted from one side to the other, it will be self-interfered with the reflection. The destructive reflection may cause spectral nulls at some frequencies. For a given segment of a twisted-pair wire, it is well known that the transfer function of twisted-pair wire is non-flat and the attenuation is proportional to the square root of frequency [1]. Thus, the overall transfer function of a typical DSL loop in Fig. 2.2 is frequency selective. The corresponding impulse

response will have a large dispersion and this result in a severe ISI in time domain. Thus, a short rectangular pulse transmitted from the CO side may result in a wide-spread distorted pulse at the CPE side. Thus, an effective equalizer is needed to compensate the ISI effect at receive side.

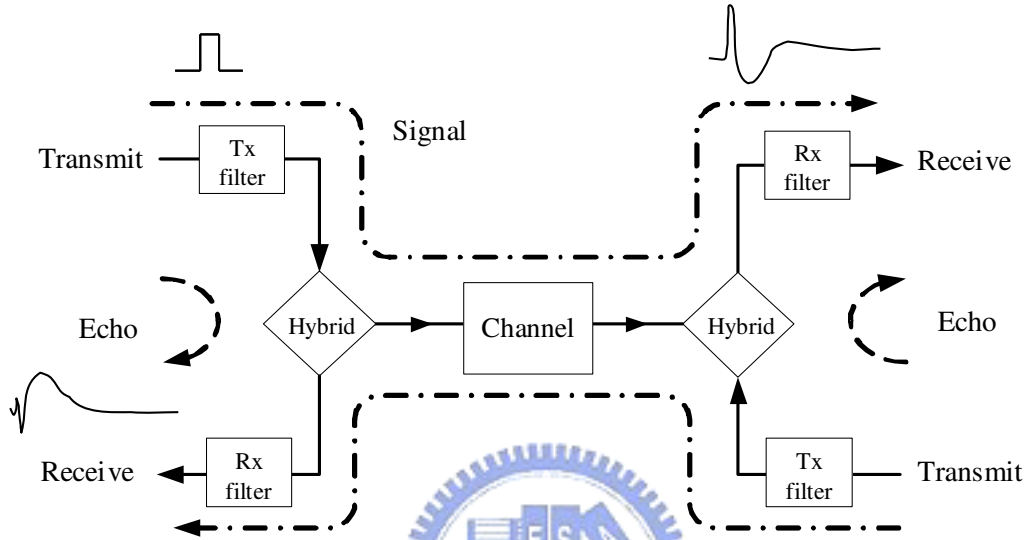


Figure 2.3: Full-duplexing transmission of a DSL channel.

In the telephone network, a special design was used to achieve full-duplex transmission over a single twisted pair of wire. Fig. 2.3 shows the architecture. In the figure, we can see a device called hybrid performing the duplexing operation. A simplified hybrid circuit [1] is shown in Fig. 2.4. The circuitry was developed according to the Wheatstone bridge principle. In fact, a hybrid circuit is an impedance bridge consisting of two voltage dividers. When the bridge is balanced, i.e.,

$$\frac{R_1}{Z_i} = \frac{R_2}{Z_b}, \quad (2.1)$$

where  $Z_i$  defined as the impedance measured at the primary side of the coupling transformer. Thus, a voltage applied at transmit terminals causes zero voltage difference between receive terminals. In the hybrid circuit, a coupling transformer shown on the bottom-left side is usually required to couple the signal between the transceiver and the twisted-pair wire and block the undesired DC signal (due to the unequal ground level between the CO and CPE sides). The



loop impedance represents the impedance of the cascaded twisted-pair wire and the loading in the receiver side. Obviously,  $Z_i$  will depend on the loop to which the hybrid connects. The balance condition in (3.1) is then difficult to hold accurately. Thus, the transmit signal will partially return to the receive terminals and this is called echo. The echo signal will interfere with the receive signal seriously. Thus, a short rectangular pulse transmitted from CO side will return a dispersed received pulse (echo) at the same side. An echo canceller is the commonly used device to eliminate the unwanted echo signal. Except for ISI and echo, there are other kinds of impairments in a telephone network such as crosstalk noise, impulsive noise, radio frequency interference (RFI) noise, and thermal noise.

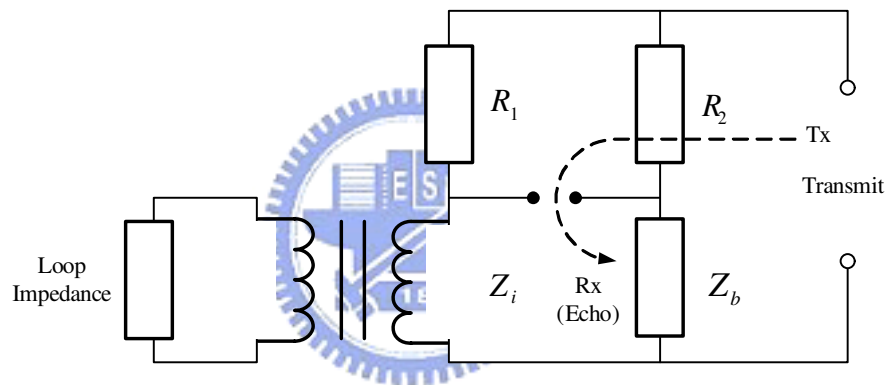


Figure 2.4: The echo caused by the impedance mismatch of a hybrid circuit.

A transceiver must be designed to cope with the impairments mentioned above so that information can be transmitted and received properly. A transceiver usually consists of an analog front-end and a digital processing unit. The functionality of a typical signal processing unit (for receiver) includes error-correcting, echo cancellation, equalization, and detection. The analog front-end is used to transform the digital signal into proper analog waveform for signal transmission, and the operation is reversed for signal reception. The analog front-end includes a transmit filter, a receive filter, an analog to digital signal conversion circuit (ADC), and digital to analog signal conversion circuit (DAC). The transmit filter is designed to shaping the transmit pulse in time domain for transmission or the power spectral density of the transmit signal

in frequency domain controlling the interference to other systems. The receive filter is used for anti-aliasing and filtering noises outside the signal band. ADC and DAC perform the analog and digital signal conversion, respectively.

To increase the throughput, a wider transmission bandwidth is required and a higher clock rate for a transceiver is necessary, too. Like the proverb said - “No pains, no gains”. A higher clock rate will result in longer (more severe) echo and ISI responses. For low speed transmission such as analog modem and ISDN, the response length of these impairments is not too long. The computational complexity for the echo canceller and the equalizer is moderate. For high speed transmission such as ADSL, SHDSL, and next generation xDSL, however, the response length will be much longer. The computational complexity will become the main obstacle in the transceiver design.



## Chapter 3

# Interpolated Echo Canceller

In a DSL environment, full duplex transmission via a single twisted pair can be achieved using a hybrid circuit. Due to the impedance mismatch problem, the hybrid circuit will introduce echoes. In an analog telephone, a hybrid circuit with 10-20 dB echo return loss is good enough for voice conversation. For high-speed digital transmission, the required echo return loss, however, is much higher (50-70 dB). In versatile subscriber loop environments, it is difficult to design a hybrid circuit achieving a high echo return loss, even when using an adaptive hybrid circuit. Thus, an additional adaptive digital echo canceller [2, 3] is required.

A subscriber loop is composed of telephone wires with different gauges and lengths [1], and has many possible topologies. A typical echo response, shown in Fig. 3.1, usually consists of a short and rapidly changing head echo, and a long and slowly decaying tail echo. Since the subscriber loop between a central office and a customer premise is fixed, its characteristics will not vary with time (except for temperature variation which is very slow). As a result, the echo response is usually considered as time-invariant. Conventionally, an adaptive transversal FIR filter, shown in Fig. 3.2, is used to synthesize and cancel the echo. For a lower speed application such as ISDN, the echo canceller has about 50 taps, and the computational complexity of the transversal filter structure is moderate. However, in higher speed applications such as HDSL [4–6], HDSL2 [7], and SHDSL [8–10], the echo response is usually much longer. The

echo canceller may require hundreds of tap weights, and the computational complexity of the transversal FIR filter approach becomes very high. In order to reduce the computational com-

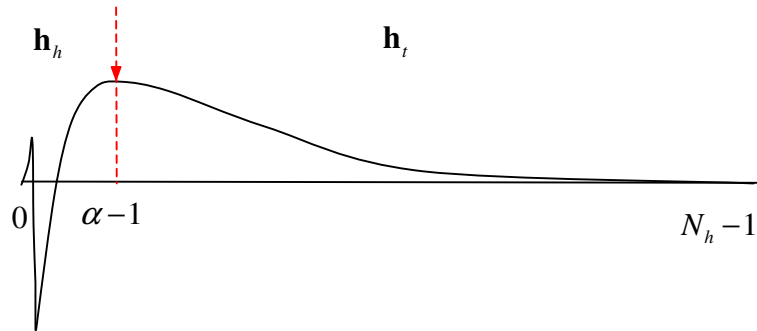


Figure 3.1: A typical echo response.

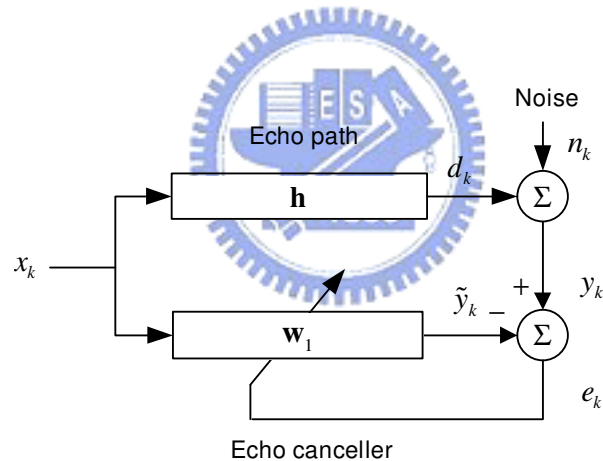


Figure 3.2: The adaptive FIR echo canceller.

plexity, some researchers tried to use an adaptive IIR filter to cancel the tail echo. However, the adaptive IIR filtering suffers from the local minima and stability problems. Since an IIR filter usually consists of a feedforward and a feedback filter, a compromising approach is to let the feedforward filter be adaptive only. In [11], August et al. collected some echo responses for the European subscriber loops, and used a criterion to determine the feedback filter optimally. In [12], Gordon et al. considered echo cancellation as a series expansion problem. They used a

set of IIR orthonormal functions to expand the echo response and let the expanding coefficients be adaptive. The orthonormal responses were obtained using a set of predetermined cascaded feedback filters. If only a small number of loops are considered, good performance can be obtained using these methods. However, the existing loop responses are versatile, it will be difficult to find a feedback filter that will always yields the optimal performance.

To retain the FIR structure of the echo canceller, and to reduce the complexity, an interesting echo canceller structure was proposed in [13, 14]. The canceller is cascaded of an adaptive FIR head echo canceller and an adaptive interpolated FIR (IFIR) tail echo canceller. Since the tail echo always decays smoothly, an IFIR filter with a small number of coefficients can effectively cancel the echo. Unfortunately, the IFIR filter proposed in [13, 14] has an uncontrollable transient response, and the direct cascade of an FIR and an IFIR filter will leave a certain period of the echo response uncanceled. Although this problem is critical, it was overlooked in [14]. Recently, a new interpolated FIR echo canceller was proposed [15] to solve the problem. In this work, the FIR and the IFIR filters are overlapped instead of being directly cascaded. Due to this overlapping operation, some echo responses will be simultaneously cancelled by the FIR and IFIR filters. Although this will not affect the final performance, it will slow down the convergence. In order to solve the problem, some of tap-weights used in the FIR filter coefficients are nulled. This results in a high performance yet low-complexity echo canceller.

An IFIR filter consists of an interpolation filter and an upsampled FIR filter. It is known that the interpolation filter for an IFIR filter has great impact on the interpolated result. Many interpolation filters have been proposed in [16]. These filters are general in the sense that they are independent to the echoes being interpolated. Since they are general, they cannot yield best performance for all types of echoes. If we know the characteristics of the signal which we will interpolate, we can design a better interpolation filter. The purpose of this thesis is to enhance the performance of the IFIR echo canceller in [15] via the interpolation filter design. We propose a least-squares method to obtain the optimal interpolation filter for a single or multiple DSL loops. Simulations show that the optimal interpolation filter can have much better performance

than the conventional interpolation filters [14, 16].

This chapter is organized as follows. In Section 3.1, we describe the IFIR echo canceller structure in [15] and review its properties. In Section 3.3, we discuss the proposed least-squares interpolation filter design. In Section 3.4, we show the simulation results and discuss the complexity reduction issues. Finally, we draw our conclusions in Section 3.5.

### § 3.1 The IFIR Echo Canceller

Let the length of an echo response be  $N_h$  and its response be  $\mathbf{h} = [h_0 \ h_1 \ \dots \ h_{N_h-1}]^T$ . The received echo signal can be described as follows:

$$y_k = \mathbf{h}^T \mathbf{x}_k + n_k \quad (3.1)$$

where  $\mathbf{x}_k = [x_k \ x_{k-1} \ \dots \ x_{k-N_h+1}]^T$  is the transmitted signal,  $n_k$  is a zero-mean additive noise which may be the additive white Gaussian noise (AWGN), the near-end crosstalk (NEXT) noise, or both combined.

From Fig. 3.1, we can clearly see that a typical echo response has a fast changing head echo and a slowly varying tail echo. Thus, we can select a cutting point to segment these two portions. Let  $\mathbf{h}_h = [h_0 \ \dots \ h_{\alpha-1}]^T$ ,  $\mathbf{x}_h = [x_k \ \dots \ x_{k-\alpha+1}]^T$ ,  $\mathbf{h}_t = [h_\alpha \ \dots \ h_{N_h-1}]^T$ , and  $\mathbf{x}_t = [x_{k-\alpha} \ \dots \ x_{k-N_h+1}]^T$ . Then, the echo response can be re-expressed as:

$$y_k = \mathbf{h}_h^T \mathbf{x}_h + \mathbf{h}_t^T \mathbf{x}_t + n_k \quad (3.2)$$

In absence of noise,  $y_k$  can be synthesized and cancelled by an  $N_h$ -tap FIR filter. This filter, having  $\mathbf{h}$  as its response, can be decomposed to an  $\alpha$ -tap and an  $(N_h - \alpha)$ -tap FIR filter; one cancels the head echo and the other cancels the tail echo. In general,  $(N_h - \alpha)$  is much larger than  $\alpha$ . As a result, the tail echo canceller will dominate the overall computational complexity. Since the tail echo is slowly varying, we can use a filter with a lower complexity to approximate  $\mathbf{h}_t$ . The idea is to use an IFIR filter, which is an interpolation filter cascaded by a filter with an upsampled response [17, 18]. The detailed structure of the IFIR echo canceller is shown in Fig.

3.3. The FIR filter  $\mathbf{w}_1$  is used to model the head echo response  $\mathbf{h}_h$  and the IFIR filter  $\mathbf{g} * \mathbf{w}_2^U$ , where ‘ $*$ ’ denotes the convolution operation, is used to model  $\mathbf{h}_t$ . Note that  $\mathbf{w}_2^U$  is an upsampled version of a filter  $\mathbf{w}_2$ . Basically,  $\mathbf{w}_2$  tries to model the downsampled version of  $\mathbf{h}_t$ , and  $\mathbf{g}$  is a FIR filter that interpolates  $\mathbf{w}_2$ . Let the downsampling factor be  $M$ , and the FIR filter and the IFIR filter be overlapped for  $N_o = N_g - M$  taps. The head echo canceller length is extended to  $N_1 = \alpha + N_o$  instead of just  $\alpha$ . As Fig. 3.3 shows,  $x_k$  is the input to  $\mathbf{w}_1$  and  $\tilde{x}_k$  is that to  $\mathbf{w}_2^U$ . The output of the IFIR echo canceller in Fig. 3.3 can be expressed as:

$$\tilde{y}_k = \mathbf{w}_1^T \mathbf{x}_{1,k} + \mathbf{w}_2^T \tilde{\mathbf{x}}_{2,k} \quad (3.3)$$

where  $\mathbf{w}_1 = [\mathbf{w}_{1,0} \ \mathbf{w}_{1,1} \ \cdots \ \mathbf{w}_{1,N_1-1}]^T$  is the  $N_1$ -tap head echo canceller,  $\mathbf{x}_{1,k} = [x_k \ x_{k-1} \ \cdots \ x_{k-N_1+1}]^T$  is its input vector,  $\mathbf{w}_2 = [\mathbf{w}_{2,0} \ \mathbf{w}_{2,1} \ \cdots \ \mathbf{w}_{2,N_2-1}]^T$  is the  $N_2$ -tap tail echo canceller, and  $\tilde{\mathbf{x}}_{2,k} = [\tilde{x}_{k-\alpha} \ \tilde{x}_{k-\alpha-M} \ \cdots \ \tilde{x}_{k-\alpha-(N_2-1)M}]^T$  is its input vector. In terms of  $z$ -transform representation, we have  $\mathbf{w}_2^U(z) = \mathbf{w}_2(z^{-M})$ . Rewriting (3.3), we have

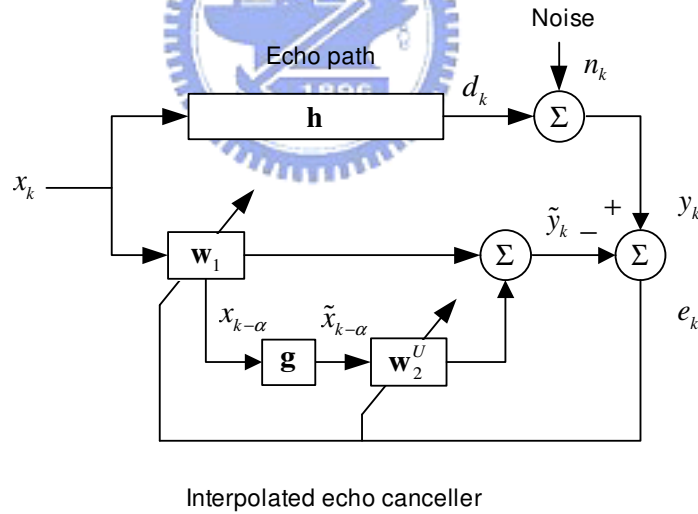


Figure 3.3: The adaptive IFIR echo canceller.

$$\begin{aligned} \tilde{y}_k &= \begin{bmatrix} \mathbf{w}_1^T & \mathbf{w}_2^T \end{bmatrix} \begin{bmatrix} \mathbf{x}_{1,k} \\ \tilde{\mathbf{x}}_{2,k} \end{bmatrix} \\ &= \mathbf{w}^T \tilde{\mathbf{x}}_k. \end{aligned} \quad (3.4)$$

Let  $\mathbf{g} = [g_0 \ g_1 \ \cdots \ g_{N_g-1}]^T$  be an  $N_g$ -tap interpolation filter, and its length equals  $2SM - 1$ , where  $S$  be the number of  $\mathbf{w}_2$  tap-weights involved in calculating an interpolated value for a single side span. Then, the interpolator output can be expressed as follows:

$$\tilde{x}_k = \sum_{i=0}^{N_g-1} g_i x_{k-i}. \quad (3.5)$$

Generally, the impulse response of the interpolation filter  $\mathbf{g}$  is peaking at the center, slowly decaying to its two sides and is symmetric around the center. The simplest response of  $\mathbf{g}$  is a triangular window function with  $2M-1$  taps, which gives a linear interpolation result. The design of  $\mathbf{g}$  is important to the IFIR filter and will be discussed later. Since the impulse response of IFIR filter is the convolution of  $\mathbf{g}$  and  $\mathbf{w}_2^U$ , it exhibits two transient responses; each one decaying to zero (each with  $N_o$  samples), one in the front end of  $\mathbf{g} * \mathbf{w}_2^U$  and the other in the tail end. Since the tail end of  $\mathbf{h}_t$  always decays to zero, there is no problem with that transient response in the tail. However, the head portion of  $\mathbf{h}_t$  has an abrupt rising edge. As a consequence, the front end transient response of  $\mathbf{g} * \mathbf{w}_2^U$  cannot model that of  $\mathbf{h}_t$ . A simple way to solve this problem is to increase the length of  $\mathbf{w}_1$ , and overlap  $\mathbf{w}_1$  with the front end transient response of  $\mathbf{g} * \mathbf{w}_2^U$ . The IFIR echo canceller overlap the last  $N_o$  taps of  $\mathbf{w}_1$  with the first  $N_o$  taps of  $\mathbf{g} * \mathbf{w}_2^U$  to cover the full front-end transient response. Fig. 3.4 shows how the FIR and IFIR responses are overlapped. Note that in the structure, there are  $(S-1)$  echo samples being cancelled simultaneously by two filters;  $\mathbf{w}_1$  and  $\mathbf{g} * \mathbf{w}_2^U$ . As shown in [15], these taps are redundant and they will slow down the convergence rate of the IFIR echo canceller. An easy and efficient way to overcome this problem is to null  $\mathbf{w}_1$ ; we can let the coefficients  $[w_{1,N_1-(S-1)M} \ \cdots \ w_{1,N_1-2M} \ w_{1,N_1-M}]$  all be zeros. This nulling scheme removes the redundant taps and accelerates the convergence rate. To obtain the tap weights of  $\mathbf{w}_1$  and  $\mathbf{w}_2$ , an adaptive algorithm is applied. From (3.4), we can see that the IFIR echo cancellation filter, similar to a conventional FIR filter, has a linear structure. As a result, adaptive algorithms developed for the conventional FIR filter can be directly applied here. For the complexity consideration, the simplest adaptive algorithm, namely



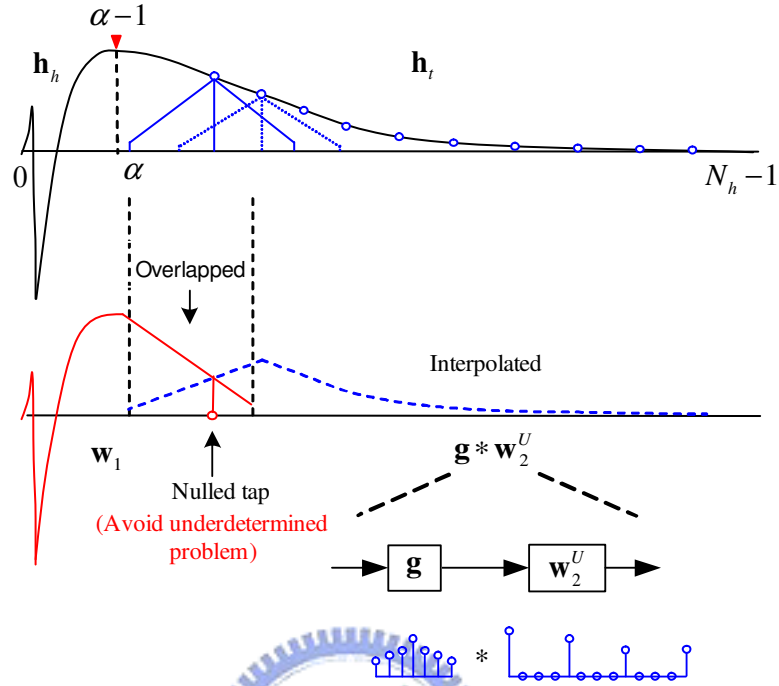


Figure 3.4: The filter responses of the IFIR echo canceller.

the least mean square (LMS), is employed. The LMS algorithm is given by [19]

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \mu e_k \tilde{\mathbf{x}}_k \quad (3.6)$$

where  $\mu$  is the step size controlling the convergence rate, and  $e_k = y_k - \tilde{y}_k$  is the error signal. The convergence behavior of the adaptive IFIR echo canceller and the upper bound of  $\mu$  for convergence will be discussed later.

The computational complexity of the adaptive IFIR echo canceller can be easily evaluated. Table 3.1 summarizes the numbers of additions and multiplications required in the echo cancellation, for an IFIR and a conventional FIR echo canceller. As we can see, the complexity reduction for the IFIR echo canceller comes from the IFIR filter. The computational complexity of  $w_2$  is only one  $M$ -th of that of the corresponding FIR filter.

Table 3.1: Computational complexity comparison for the FIR and IFIR echo cancellers

| Operation | Echo emulation        |                   | Taps weight update |             |
|-----------|-----------------------|-------------------|--------------------|-------------|
|           | +                     | $\times$          | +                  | $\times$    |
| FIR       | $N_h - 1$             | $N_h$             | $N_h$              | $N_h$       |
| IFIR      | $N_1 + N_2 + N_g - 2$ | $N_1 + N_2 + N_g$ | $N_1 + N_2$        | $N_1 + N_2$ |

## § 3.2 Theoretical Analysis

The Wiener solution, minimum mean squared error (MMSE), and error return loss enhancement (ERLE) of the IFIR echo canceller have been derived in [15]. Here, we only give the final results. These formulas will be used for performance evaluation. Let  $\mathbf{R} = E[\tilde{\mathbf{x}}_k \tilde{\mathbf{x}}_k^T]$  be the input correlation matrix (without nulling), and  $\mathbf{p} = E[\tilde{\mathbf{x}}_k y_k]$  a cross correlation vector. The Wiener solution with coefficient nulling is

$$\hat{\mathbf{w}}_o = \hat{\mathbf{R}}^{-1} \hat{\mathbf{p}} \quad (3.7)$$

where  $\hat{\mathbf{R}}$  and  $\hat{\mathbf{p}}$  is the correlation matrix and vector for the nulled filter, respectively.  $\hat{\mathbf{R}}$  and  $\hat{\mathbf{p}}$  are obtained by eliminating the  $i$ -th row and  $i$ -th column of  $\mathbf{R}$ , and the  $i$ -th row of  $\mathbf{p}$ , where  $i \in \{(N_1 - (S-1)M), \dots, (N_1 - 2M), (N_1 - M)\}$ , respectively. By doing so, the corresponding weights in  $\mathbf{w}_1$  will be all zeros, i.e.,  $[w_{1, N_1 - (S-1)M} \cdots w_{1, N_1 - 2M} w_{1, N_1 - M}] = \mathbf{0}_{1 \times (S-1)}$ .

From definition, the correlation matrix without nulling is

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_{\mathbf{x}_1 \mathbf{x}_1} & \mathbf{R}_{\mathbf{x}_1 \tilde{\mathbf{x}}_2} \\ \mathbf{R}_{\mathbf{x}_1 \tilde{\mathbf{x}}_2}^T & \mathbf{R}_{\tilde{\mathbf{x}}_2 \tilde{\mathbf{x}}_2} \end{bmatrix}. \quad (3.8)$$

Assume that the transmitted signal  $x_k$  is white. The correlation matrix of  $\mathbf{x}_{1,k}$  is

$$\mathbf{R}_{\mathbf{x}_1 \mathbf{x}_1} = \sigma_x^2 \mathbf{I}_{N_1 \times N_1} \quad (3.9)$$

where  $\sigma_x^2$  is the transmitted signal variance. The correlation matrix of  $\tilde{\mathbf{x}}_{2,k}$  is

$$\mathbf{R}_{\tilde{\mathbf{x}}_2 \tilde{\mathbf{x}}_2} = \sigma_x^2 \mathbf{M} \mathbf{M}^T \quad (3.10)$$

where

$$\mathbf{M} = \begin{bmatrix} \mathbf{g}^T, \underbrace{0, \dots, 0}_{(N_2-1)M} \\ \underbrace{0, \dots, 0}_M, \mathbf{g}^T, \underbrace{0, \dots, 0}_{(N_2-2)M} \\ \vdots \\ \underbrace{0, \dots, 0}_{(N_2-1)M}, \mathbf{g}^T \end{bmatrix} \quad (3.11)$$

is an  $N_2$ -by- $(N_h - \alpha)$  interpolation matrix. The cross correlation matrix of  $\mathbf{x}_{1,k}$  and  $\tilde{\mathbf{x}}_{2,k}$  is given by

$$\mathbf{R}_{\mathbf{x}_1 \tilde{\mathbf{x}}_2} = \sigma_x^2 \begin{bmatrix} \mathbf{0}_{\alpha \times N_o} & \mathbf{0}_{\alpha \times (N_h - N_1)} \\ \mathbf{I}_{N_o \times N_o} & \mathbf{0}_{N_o \times (N_h - N_1)} \end{bmatrix} \mathbf{M}^T. \quad (3.12)$$

If we assume that noise  $n_k$  is independent of the transmitted signal  $x_k$ , then, the cross correlation vector is given by

$$\mathbf{p} = \sigma_x^2 \begin{bmatrix} \mathbf{h}(0 : N_1 - 1) \\ \mathbf{M}\mathbf{h}(\alpha : N_h - 1) \end{bmatrix} \quad (3.13)$$

where the notation  $\mathbf{h}(i : j)$  denote a vector whose elements consisting of the  $i$ -th to the  $j$ -th component of  $\mathbf{h}$ .

The residual echo response is

$$\Delta \mathbf{h} = \mathbf{h} - (\hat{\mathbf{w}}_{o,1} + \mathbf{g} * \hat{\mathbf{w}}_{o,2}^U) \quad (3.14)$$

where  $\hat{\mathbf{w}}_{o,1}$  and  $\hat{\mathbf{w}}_{o,2}$  are the optimal weights for  $\mathbf{w}_1$  and  $\mathbf{w}_2$ , respectively, and  $\hat{\mathbf{w}}_{o,2}^U$  is an upsampled version of  $\hat{\mathbf{w}}_{o,2}$ . The MMSE is then equal to the summation of the residual echo power and the noise variance.

$$\text{MMSE} = (\Delta \mathbf{h}^T \Delta \mathbf{h}) \sigma_x^2 + \sigma_n^2 \quad (3.15)$$

where  $\sigma_n^2$  is the noise variance. The theoretical ERLE then equals

$$\text{ERLE} = 10 \cdot \log_{10} \frac{\mathbf{h}^T \mathbf{h}}{\Delta \mathbf{h}^T \Delta \mathbf{h}}. \quad (3.16)$$

We now analyze the convergence behavior of the adaptive IFIR filter. From (3.6), we see that the update equation for the adaptive IFIR filter is identical to that of a standard adaptive FIR filter, except for the input vectors. Thus, the existing results for the adaptive FIR filter can be applied. Using the independence theory [19], we assume that  $\{\tilde{\mathbf{x}}_k\}$  be a sequence of statistically independent vectors. Defining the weight-error vector as  $\varepsilon_k = \hat{\mathbf{w}}_k - \hat{\mathbf{w}}_o$ , we then have [19]

$$E[\varepsilon_{k+1}] = (\mathbf{I} - \mu \hat{\mathbf{R}}) E[\varepsilon_k]. \quad (3.17)$$

Thus, if the step size satisfies the following condition, the mean of  $\varepsilon_k$  will converge to zero as  $k$  approaches infinity:

$$0 < \mu < \frac{2}{\lambda_{\max}} \quad (3.18)$$

where  $\lambda_{\max}$  is the largest eigenvalue of the correlation matrix  $\hat{\mathbf{R}}$ .

Let the MSE of the adaptive IFIR filter be denoted as  $J_k = E[e_k^2]$ . The MSE,  $J_k$ , will converge to a steady-state value equal to  $J_\infty$  if, and only if, the step-size  $\mu$  satisfies the following two conditions [19]:

$$0 < \mu < \frac{2}{\lambda_{\max}}, \quad \sum_{n=1}^{N_1+N_2-S+1} \mu \lambda_n / 2 (1 - \mu \lambda_n) < 1 \quad (3.19)$$

where  $\lambda_n$  is the  $n$ -th eigenvalue of the correlation matrix  $\hat{\mathbf{R}}$ , and  $N_1 + N_2 - S + 1$  is the number of adjustable tap weights in the proposed IFIR echo canceller. The MSE value in the steady state is given by

$$J_\infty = \frac{J_{\min}}{1 - \sum_{n=1}^{N_1+N_2-S+1} \mu \lambda_n / 2 (1 - \mu \lambda_n)} \quad (3.20)$$

where  $J_{\min}$  is the MMSE shown in (3.15). The misadjustment  $\psi$  can then be expressed as

$$\psi = \frac{\sum_{n=1}^{N_1+N_2-S+1} \mu \lambda_n / 2 (1 - \mu \lambda_n)}{1 - \sum_{n=1}^{N_1+N_2-S+1} \mu \lambda_n / 2 (1 - \mu \lambda_n)}. \quad (3.21)$$

From simulations we found that the convergence speed of the proposed adaptive IFIR filter is somewhat slower than that of the conventional LMS FIR filter. This is due to the fact that the input correlation matrix for the IFIR filter is not truly diagonal.

### § 3.3 Optimal Interpolation Filter Design

Given a value for the interpolation factor  $M$ , the optimal interpolation filter is the ideal lowpass filter with bandwidth  $\pi/M$ ; however, the corresponding impulse response is an unrealizable sinc function. Therefore, many suboptimal interpolation filters have been proposed [16]. These filters are general in the sense that they are independent to the signal being interpolated. If we know the characteristics of the signals for which we will be interpolating, we can design a better interpolation filter. In this section, we propose a least-squares method to obtain the optimal interpolation filter for a set of given tail echo responses.

We first consider the interpolation filter design for a single tail echo response. Let

$$\begin{aligned} \mathbf{f} &= [f_0 f_1 \cdots f_{(N_2-1)M}]^T \\ &= [h_\alpha h_{\alpha+1} \cdots h_{\alpha+(N_2-1)M}]^T \end{aligned} \quad (3.22)$$

be a tail echo response. Here, we take  $((N_2 - 1)M + 1)$ -sample from the original tail echo. If the original tail echo is not long enough, we can pad zeros. Let

$$\begin{aligned} \hat{\mathbf{f}} &= [\hat{f}_0 \hat{f}_1 \cdots \hat{f}_{(N_2-1)M}]^T \\ &= [f_0 \underbrace{0, \cdots, 0}_{M-1} f_M \underbrace{0, \cdots, 0}_{M-1} \cdots f_{(N_2-1)M}]^T \end{aligned} \quad (3.23)$$

be the corresponding downsampled and then upsampled response used in interpolation,  $\mathbf{g}$  be the interpolation filter response, and

$$\tilde{\mathbf{f}} = [\tilde{f}_0 \tilde{f}_1 \cdots \tilde{f}_{(N_2-1)M}]^T \quad (3.24)$$

be the interpolated tail response. Note that

$$\tilde{f}_k = \sum_{i=0}^{N_g-1} g_i \hat{f}_{k-i}. \quad (3.25)$$

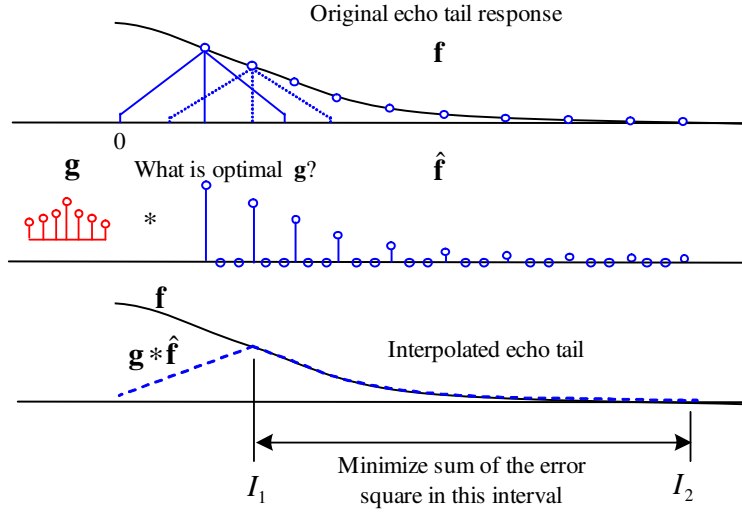


Figure 3.5: Illustration of finding optimal interpolation filter.

Let the interpolation error response be  $v_i = f_i - \tilde{f}_i$ . We can then define a cost function using the error response as

$$\xi(\mathbf{g}) = \sum_{i=I_1}^{I_2} v_i^2 \quad (3.26)$$

where  $I_1 = N_g - 1$  and  $I_2 = (N_2 - 1)M$ . Note that the cost function does not include the interpolation errors for the first  $N_g - 1$  and last  $N_g - 1$  transient responses of  $\tilde{f}_k$ , as shown in Fig. 3.5. Obviously, the optimization problem is a classic least-squares problem. Define an error vector as  $\mathbf{v} = [v_{I_1} \ v_{I_1+1} \ \cdots \ v_{I_2}]^T$ . We can rewrite the cost function in (3.26) using a vector form.

$$\begin{aligned} \xi(\mathbf{g}) &= \mathbf{v}^T \mathbf{v} \\ &= (\mathbf{f}_s - \tilde{\mathbf{f}}_s)^T (\mathbf{f}_s - \tilde{\mathbf{f}}_s) \end{aligned} \quad (3.27)$$

where  $\mathbf{f}_s = \mathbf{f}(I_1 : I_2)$  is the reduced-length tail response,  $\tilde{\mathbf{f}}_s = \tilde{\mathbf{f}}(I_1 : I_2)$  is the corresponding interpolated response. If we expressed  $\tilde{\mathbf{f}}_s$  in the following matrix form,

$$\tilde{\mathbf{f}}_s = \mathbf{F}\mathbf{g} \quad (3.28)$$

where

$$\mathbf{F} = \begin{bmatrix} \hat{f}_{I_1} & \hat{f}_{I_1-1} & \cdots & \hat{f}_{I_1-(N_g-1)} \\ \hat{f}_{I_1+1} & \hat{f}_{I_1} & \cdots & \hat{f}_{I_1-(N_g-2)} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{f}_{I_2} & \hat{f}_{I_2-1} & \cdots & \hat{f}_{I_2-(N_g-1)} \end{bmatrix} \quad (3.29)$$

is a  $(I_2 - I_1 + 1)$ -by- $N_g$  Toeplitz matrix, then the cost function in (3.27) can be re-expressed as:

$$\xi(\mathbf{g}) = (\mathbf{f}_s - \mathbf{F}\mathbf{g})^T (\mathbf{f}_s - \mathbf{F}\mathbf{g}). \quad (3.30)$$

Taking the derivative with respect to  $\mathbf{g}$  in (3.30) and set the result to zero, we can obtain the least-squares solution as [20]

$$\hat{\mathbf{g}} = (\mathbf{F}^T \mathbf{F})^{-1} \mathbf{F}^T \mathbf{f}_s. \quad (3.31)$$

Extending the idea developed above, we can find a single optimal interpolation filter for a set of echo loops. Let  $\mathbf{f}_{s,i}$  be the reduced-length tail response for the  $i$ -th loop,  $N$  be the number of loops considered, and  $\tilde{\mathbf{f}}_{s,i}$  the corresponding interpolated response. Now we augment the interpolated response in (3.28) as follows:

$$\begin{bmatrix} \tilde{\mathbf{f}}_{s,1} \\ \tilde{\mathbf{f}}_{s,2} \\ \vdots \\ \tilde{\mathbf{f}}_{s,N} \end{bmatrix} = \begin{bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \\ \vdots \\ \mathbf{F}_N \end{bmatrix} \mathbf{g}. \quad (3.32)$$

where  $\mathbf{F}_i$  is the matrix in (3.29) for the  $i$ -th loop. Let  $\bar{\mathbf{f}}_s = \begin{bmatrix} \tilde{\mathbf{f}}_{s,1} & \tilde{\mathbf{f}}_{s,2} & \cdots & \tilde{\mathbf{f}}_{s,N} \end{bmatrix}^T$ , and  $\bar{\mathbf{F}} = \begin{bmatrix} \mathbf{F}_1 & \mathbf{F}_2 & \cdots & \mathbf{F}_N \end{bmatrix}^T$ . We can then define a cost function similar to (3.30) as

$$\bar{\xi}(\mathbf{g}) = (\bar{\mathbf{f}}_s - \bar{\mathbf{F}}\mathbf{g})^T (\bar{\mathbf{f}}_s - \bar{\mathbf{F}}\mathbf{g}) \quad (3.33)$$

The cost function in (3.33) is equivalent to

$$\begin{aligned} \bar{\xi}(\mathbf{g}) &= \sum_{i=1}^N \xi_i(\mathbf{g}) \\ &= \sum_{i=1}^N \sum_{j=I_1}^{I_2} |v_{ij}|^2. \end{aligned} \quad (3.34)$$

Using the least-squares method, we can obtain the optimal interpolation filter for multiple loops as

$$\tilde{\mathbf{g}} = (\bar{\mathbf{F}}^T \bar{\mathbf{F}})^{-1} \bar{\mathbf{F}}^T \bar{\mathbf{f}}_s. \quad (3.35)$$

and the sum of averaged squared errors as

$$\begin{aligned} \bar{\xi}_{\min} &= \|\bar{\mathbf{f}}_s - \bar{\mathbf{F}} \tilde{\mathbf{g}}\|^2 \\ &= \bar{\mathbf{f}}_s^T \left( \mathbf{I} - \bar{\mathbf{F}} (\bar{\mathbf{F}}^T \bar{\mathbf{F}})^{-1} \bar{\mathbf{F}}^T \right) \bar{\mathbf{f}}_s. \end{aligned} \quad (3.36)$$

Assuming that the head echo, including the overlapped portion is cancelled perfectly, we have the theoretical ERLE for loop  $i$  as

$$\text{ERLE} = 10 \cdot \log_{10} \frac{\|\mathbf{h}_i\|^2}{\xi_{\min,i}} \quad (3.37)$$

where  $\mathbf{h}_i$  is the  $i$ -th echo response vector, and  $\xi_{\min,i} = \|\mathbf{f}_{s,i} - \mathbf{F}_i \tilde{\mathbf{g}}\|^2$  is the sum of the minimum squared errors for that loop. Note that the optimal interpolation filter is usually not symmetric. The computational complexity for the interpolation operations, using the optimal filter, is higher than that when using the conventional symmetric ones. We can solve the problem by constraining the filter response to be symmetric, i.e.,  $g_i = g_{N_g-1-i}$ ,  $i = 0, 1, \dots, (SM - 2)$ . The optimal solution for a single echo loop is identical to that in (3.31) except for the matrix  $\mathbf{F}$ . The matrix now becomes:

$$\mathbf{F} = \begin{bmatrix} \hat{f}_{I_1} + \hat{f}_{I_1-(N_g-1)} & \cdots & \hat{f}_{I_1-(SM-1)} \\ \hat{f}_{I_1+1} + \hat{f}_{I_1-(N_g-2)} & \cdots & \hat{f}_{I_1-SM} \\ \vdots & \ddots & \vdots \\ \hat{f}_{I_2} + \hat{f}_{I_2-(N_g-1)} & \cdots & \hat{f}_{I_2-(SM-1)} \end{bmatrix}. \quad (3.38)$$

Compared to (3.29), the column dimension of (3.38) is reduced by half. The  $(N_g - 1 - i)$ -th column of  $\mathbf{F}$  in (3.29) is added to the  $i$ -th column,  $i = 0, 1, \dots, (SM - 2)$  except for the middle column corresponding to the central tap of  $\mathbf{g}$ . That means the middle column of  $\mathbf{F}$  is unchanged and its result is the most right column in (3.38). The dimension of  $\mathbf{g}$  is also reduced by half.



Using a similar method, we can find the optimal symmetric interpolation filter for a single or for multiple loops via (3.31) or (3.35).

In this paragraph, we compare the performance of some existing interpolation filters with proposed ones. Those we considered include the linear, the truncated-sinc, the Hanning-windowed sinc, and the Chebyshev-windowed sinc filters. The responses of the Chebyshev-windowed sinc, the optimal, and the optimal symmetric interpolation filters are shown in Fig. 3.6 ( $S = 2$  and  $M=4$ ), and the theoretical ERLEs for the CSA loop #1 [7] are shown in Table 3.2. Note that the optimal interpolation filters are optimized for eight CSA loops [7] then apply to CSA loop #1. As we can see, for the case of  $S=1$  only the optimal interpolation filter has ERLE higher than 70-dB. For the case of  $S=2$ , the proposed interpolation filters outperform other filters by an amount of more than 10 dB. For the case of  $S=3$ , the proposed filter still has the best performance, however, the performance difference is reduced. Specifically the performance of the Chebyshev-windowed sinc filter is close to that of the proposed ones. It seems that the performance bound has been reached, and that there is no need to consider a span of more than three. We conclude that the proposed method provides a systematic rather than a heuristic or a trial-and-error way to find the interpolation filter. From the results shown in Table 3.2, we can see that the two-span symmetric interpolator seems a good choice for the IFIR echo canceller.

Table 3.2: ERLE performance vs. various interpolation filters (LIN: linear, TRS: truncated sinc, HWS: Hanning-windowed sinc, CWS: Chebyshev-windowed sinc, OPS: optimal-symmetric, OPT: optimal)

| Filter | LIN  | TRS  | HWS  | CWS  | OPS  | OPT  |
|--------|------|------|------|------|------|------|
| $S=1$  | 62.4 | 32.3 | 31.5 | 28.5 | 68.9 | 70.1 |
| $S=2$  | -    | 39.1 | 55.1 | 57.8 | 73.2 | 73.3 |
| $S=3$  | -    | 43.4 | 66.4 | 75.0 | 75.9 | 76.3 |

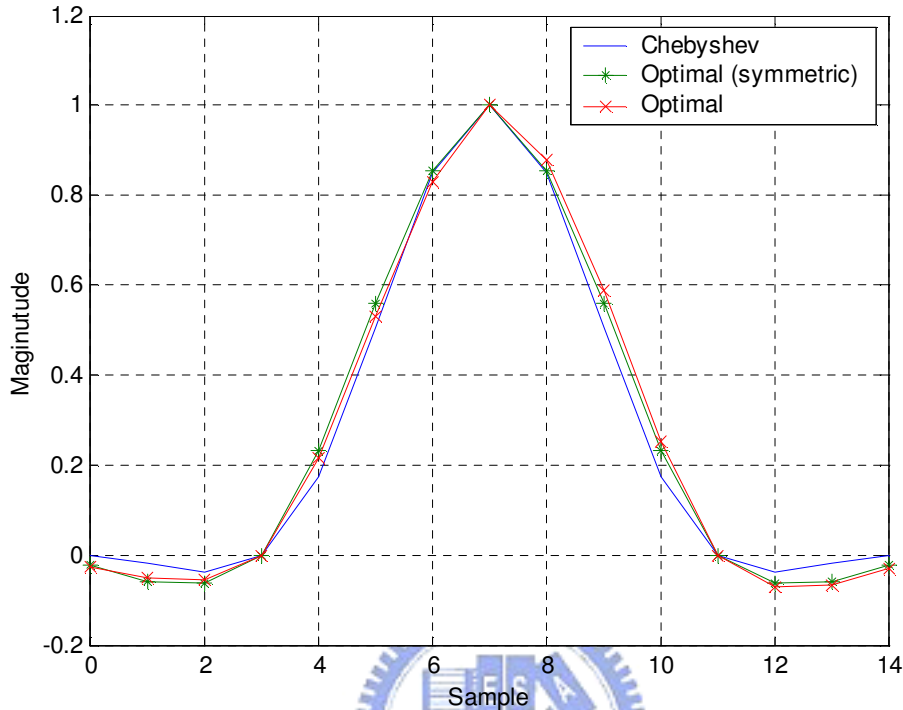


Figure 3.6: Interpolation filters ( $M=4$ ,  $S=2$ ).

## § 3.4 Simulation Results

In this section, we report some computer simulation results to demonstrate the effectiveness of the optimal IFIR echo canceller. Specifically, we have taken SHDSL as the application example and evaluated the echo cancellation performance under scenarios with different loop topologies, echo cutting points, interpolation factors, and noise environments.

### § 3.4.1 Loop Characteristics and Topologies

To test the robustness of the optimal IFIR echo canceller, we used eight CSA loops in [7] for simulations. The method to model echo responses was described in [1, 21]. The echo path contained a transmit shaping filter, a transmit differential hybrid circuit (with a  $135\Omega$  termination

impedance), a CSA loop, a receive differential hybrid circuit, and a receive filter. The transmit/receive filter was modelled as a 6-th order Butterworth lowpass filter with a 3-dB cutoff frequency at 775 KHz. The primary inductance of the transformer in the hybrid circuits was 3 mH. For the SHDSL application, the sampling rate was as high as 775 KHz. The simulated echo responses at the central office side (CO) and the customer premise side (CPE) are shown in Fig. 3.7 and Fig. 3.8, respectively. The responses exhibited a short and rapidly changing head portion and a long and slowly decaying tail portion, as was expected. The line code of

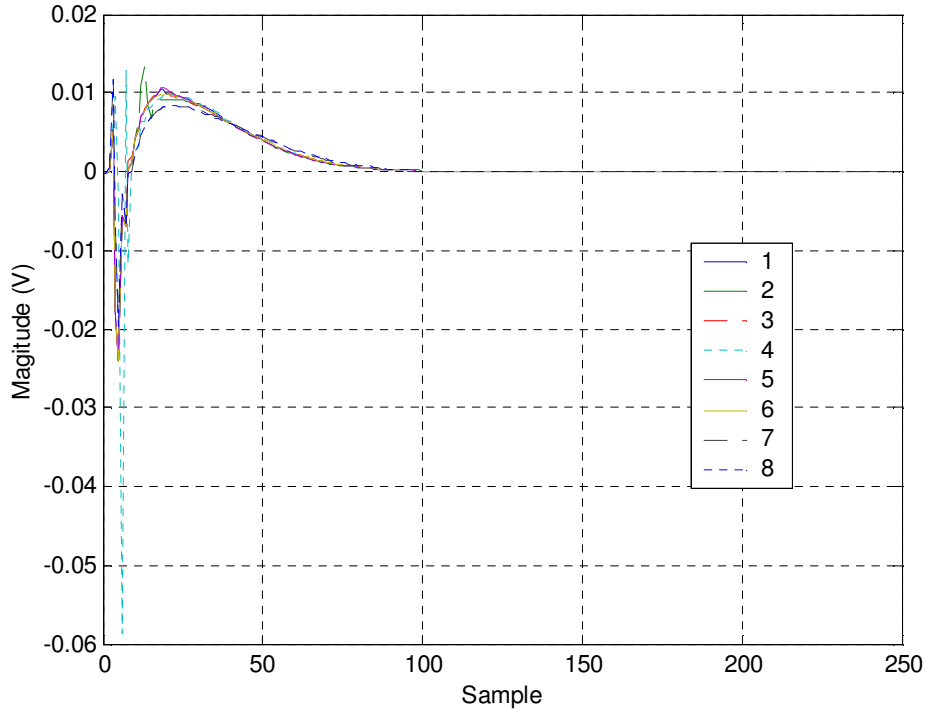


Figure 3.7: SHDSL echo responses for CSA loops at CO side.

the transmit signal was 16-PAM. Here, AWGN with -140 dBm/Hz was used to contaminate the received signal. The cutting point  $\alpha$  was set as 39 and the interpolation factor  $M$  was set as 4. The optimal symmetric interpolation filter ( $S=2$ ) obtained by using the eight CSA loops, was applied. We then have  $N_1 = 50$ . During the training period, the far-end transmit signal was

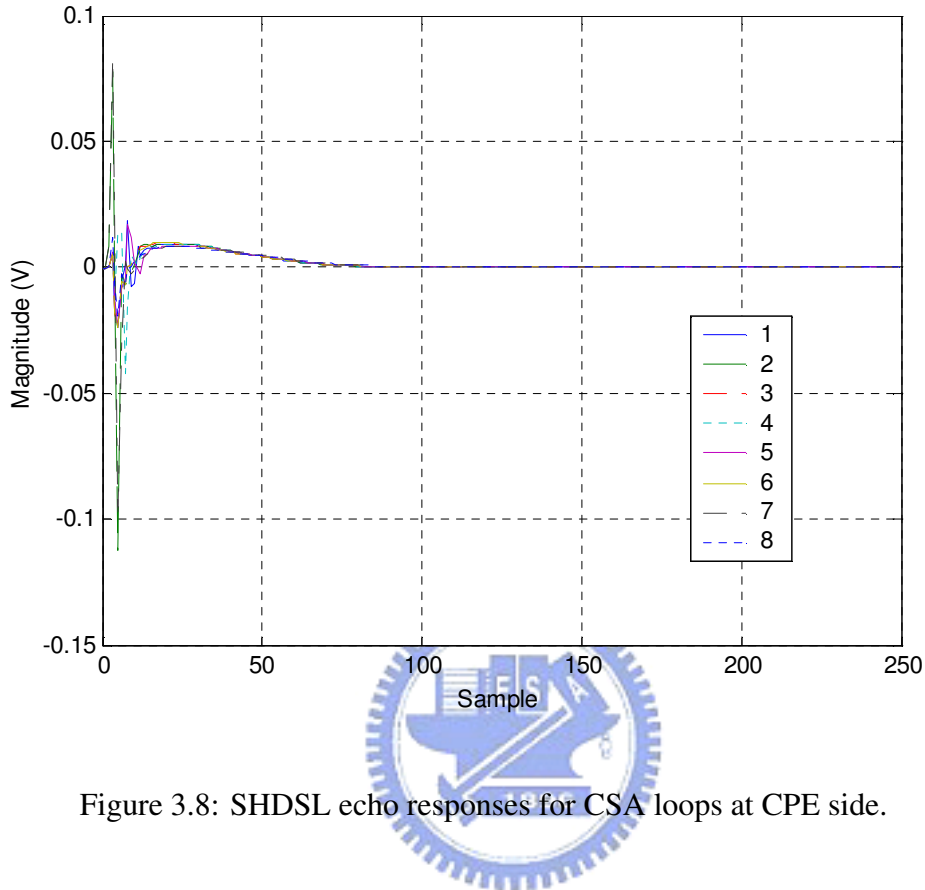


Figure 3.8: SHDSL echo responses for CSA loops at CPE side.

turned off. After that, the transceiver was operated in a full duplex data transmission mode. For a faster convergence, the step size was varied using the following scheme. The training period was divided into five stages and the overall period was 12,000 samples. In each stage, the step size was simply reduced by a factor of two. The step size was initialized as  $1/N_h = 0.004$ . The emulated echo response, which was an overlapped combination of the FIR and IFIR responses, is shown in Fig. 3.9. Note that there was one nulled tap (zero weight) located in the tail end of  $w_1$ . As we can see, the tail response was modelled accurately using the IFIR filter except for the transient response in the beginning, however,  $w_1$  compensated for that effectively. All the eight CSA test loops both at the CO and the CPE side were simulated. The resultant ERLE performances are shown in Fig. 3.10. As the figure shows, the ERLE was between 73.0 and 77.1 dB. The averaged ERLE was around 74.0 dB at the CO side, and 74.5 dB at the CPE side.

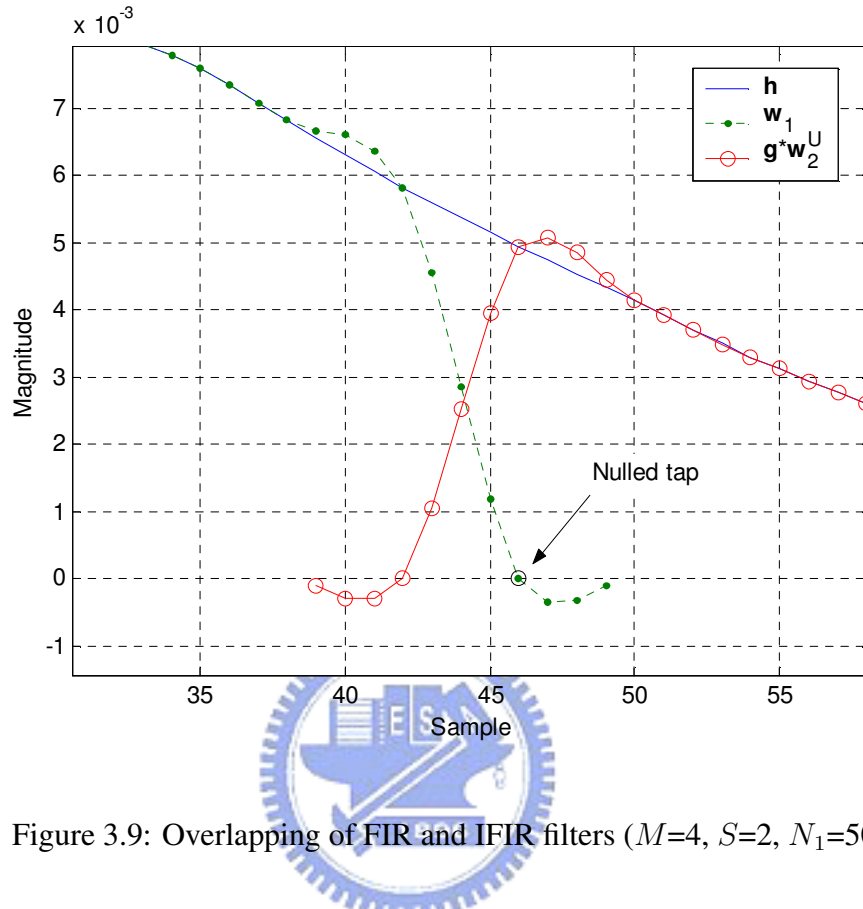


Figure 3.9: Overlapping of FIR and IFIR filters ( $M=4$ ,  $S=2$ ,  $N_1=50$ ).

The low sensitivity of the optimal IFIR echo canceller to different topologies and loop characteristics exhibited its feasibility to real-world applications. Generally speaking, theoretical ERLE predictions, which are also shown in Fig. 3.10, were accurate. The higher the ERLE, the larger the difference between the theoretical and empirical ERLEs. This was because if the ERLE was higher, a smaller step size was required to hold the independence theory. However, we used the same step size for all cases.

### § 3.4.2 The Cutting Point and the Interpolation Factor

In this set of simulations, all the parameters and the interpolation filters were identical to those in the previous one. However, only CSA loop #1 was used. To show the influence of the cutting point and the interpolation factor, we give theoretical ERLE in a two-dimensional plot

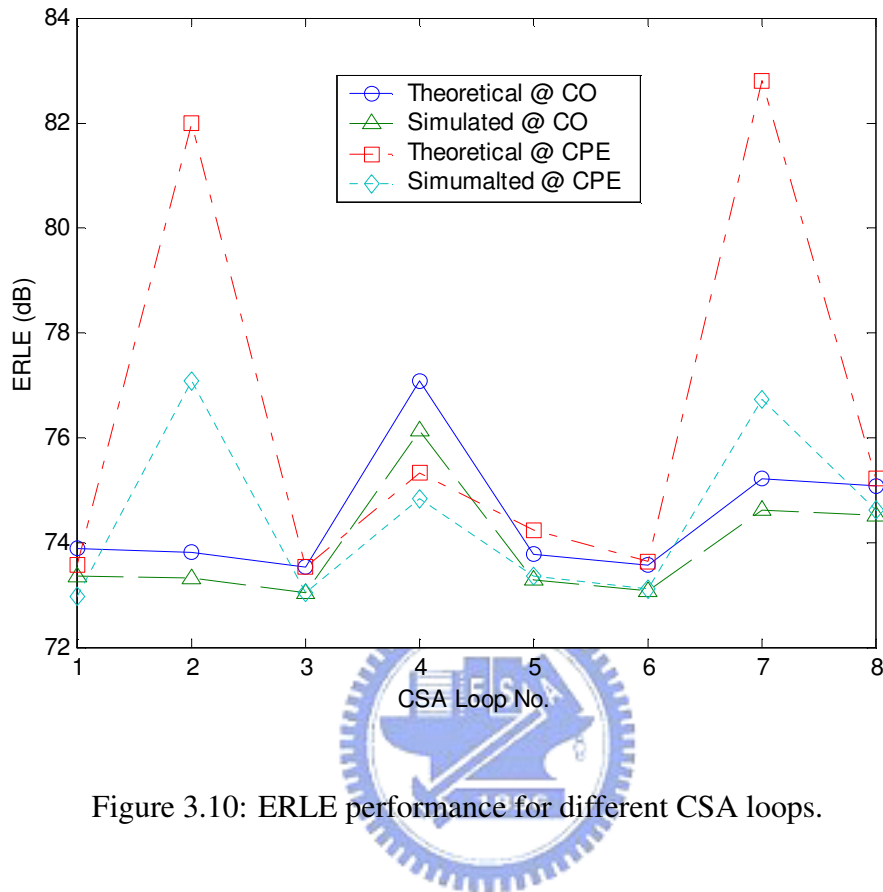


Figure 3.10: ERLE performance for different CSA loops.

in Fig. 3.11. In the figure, the cutting point varies from 30 to 70 and the interpolation factor from two to eight. Note that the performance surface is monotonically increasing with respect to the increasing of the cutting point position and the decreasing with the interpolation factor. The ERLE is always higher than 67.4 dB (76.9 dB in average). For an interpolation factor less than (or equal to) 4 and the cutting point greater than 30, the ERLE performance exceeds 70 dB. A larger cutting point or a smaller interpolation factor implies higher computational complexity. Thus, there must be a compromise between performance and complexity. The computational complexity comparison for the IFIR and the conventional FIR echo cancellers is shown in Table 3.3. The complexity ratio in Table 3.3 is defined as the ratio of the computational complexity of the IFIR canceller and that of the FIR canceller. Here, the cutting point is 39 and the interpolation filter spans 4 samples. For the interpolation factor of 2, 4, and 8, computational

complexity for the IFIR canceller is (in terms of additions and multiplications) 62%, 43%, and 36% of that for the conventional FIR canceller, respectively. Using the interpolation factor of 4, we can achieve more than 70-dB ERLE while reduce 57% complexity.

Table 3.3: Computational complexity ratio for different interpolation factor ( $N_1 = 50, S=2$ )

|       | Echo emulation |     | Taps weight update |     | Complexity ratios |     |
|-------|----------------|-----|--------------------|-----|-------------------|-----|
|       | +              | ×   | +                  | ×   | +                 | ×   |
| $M=2$ | 156            | 158 | 151                | 151 | 62%               | 62% |
| $M=4$ | 114            | 116 | 101                | 101 | 43%               | 43% |
| $M=8$ | 105            | 107 | 76                 | 76  | 36%               | 36% |

### § 3.4.3 Noise Environments: AWGN and NEXT

In this subsection, we evaluate the performance of the optimal IFIR echo canceller under different noise environments. The scenario includes pure AWGN and a composite noise consisting of AWGN and NEXT noise. The power spectral density (PSD) of the coupled NEXT can be evaluated using the formula shown below [21, 22].

$$\begin{aligned} \text{PSD}_{NEXT} &= \text{PSD}_{Disturber} \cdot (x_n f^{3/2}) \\ 0 \leq f < \infty, n < 50, x_n &= 0.8536 \cdot 10^{-14} \cdot n^{0.6} \end{aligned} \quad (3.39)$$

where the  $\text{PSD}_{Disturber}$  is the PSD of a disturbing xDSL line code such as 16-PAM, DMT, 2B1Q, etc. The parameter  $x_n$  is a coupling coefficient and its value depends on the number of disturbers  $n$ . Both self-NEXT as well as foreign-NEXT were considered. For the self-NEXT case, NEXT noise was contributed by 10-SHDSL disturbers. For the foreign-NEXT case, NEXT noise was contributed by 24-ISDN disturbers, 10-HDSL disturbers, non-collocation<sup>1</sup> 24-T1 disturbers, or

<sup>1</sup>Contrary to [8], here, we assume that the T1 and SHDSL terminals are non-collocation, a factor of 15.5 dB downward of interferer PSD is included for modelling the adjacent binder effect [22, 23].

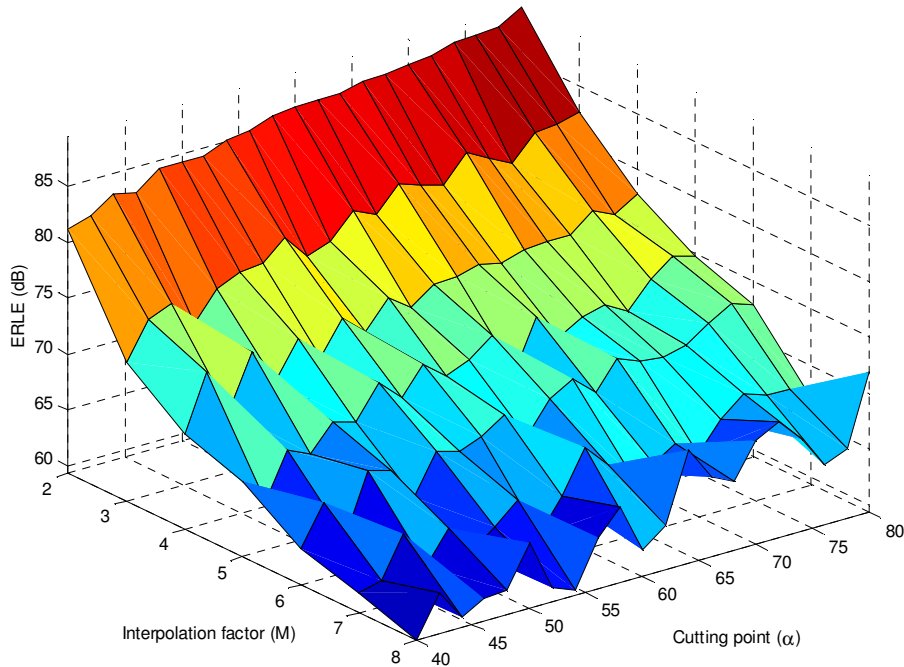


Figure 3.11: ERLE versus the cutting point and the interpolation factor.

10-downstream/upstream-ADSL disturbers. All NEXTs were from pairs in the same binder group except T1 crosstalk that was from pairs in an adjacent binder group. Finally, NEXT and AWGN (-140 dBm/Hz) were then summed together to form a composite noise. The ERLE performance at the CO and the CPE side for different CSA loop under the composite noise environments were simulated and the results are shown in Fig. 3.12 and Fig. 3.13, respectively. In either case, the ERLE under AWGN is the highest. The ERLE is lower in the NEXT environments and inversely proportional to the NEXT power. Note that the step sizes used here were all the same. From (3.8) and (3.13), we can see that the Wiener solution is independent of the noise power. The noise power only affects the misadjustment. It is well known that the smaller the step size, the smaller the misadjustment [19]. This is to say that if we could reduce the step size, all the ERLEs would be as high as that in the AWGN case. However, as we will discuss



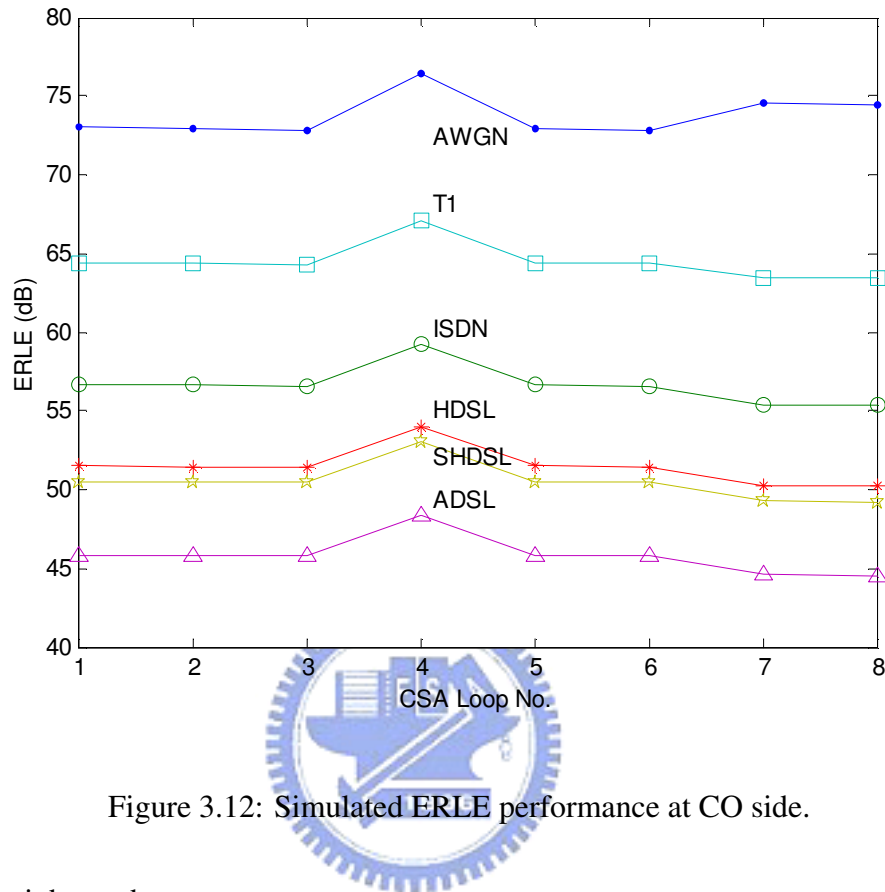


Figure 3.12: Simulated ERLE performance at CO side.

below, this might not be necessary.

### § 3.4.4 Discussions

The mission of an echo canceller is to cancel the echo signal produced by the transmission symbols. It can do nothing about noise. Thus, if the level of residual echo is below that of noise, echo is no longer the main factor limiting the system performance. So, let's define the total echo return loss (TERL) as

$$\text{TERL} = (\text{ERL} + \text{ERLE}), \quad (3.40)$$

where ERL is the echo return loss of the hybrid circuit. The value of ERL usually depends on the loop topology, loop characteristics as well as the receiver location. For convenience, we

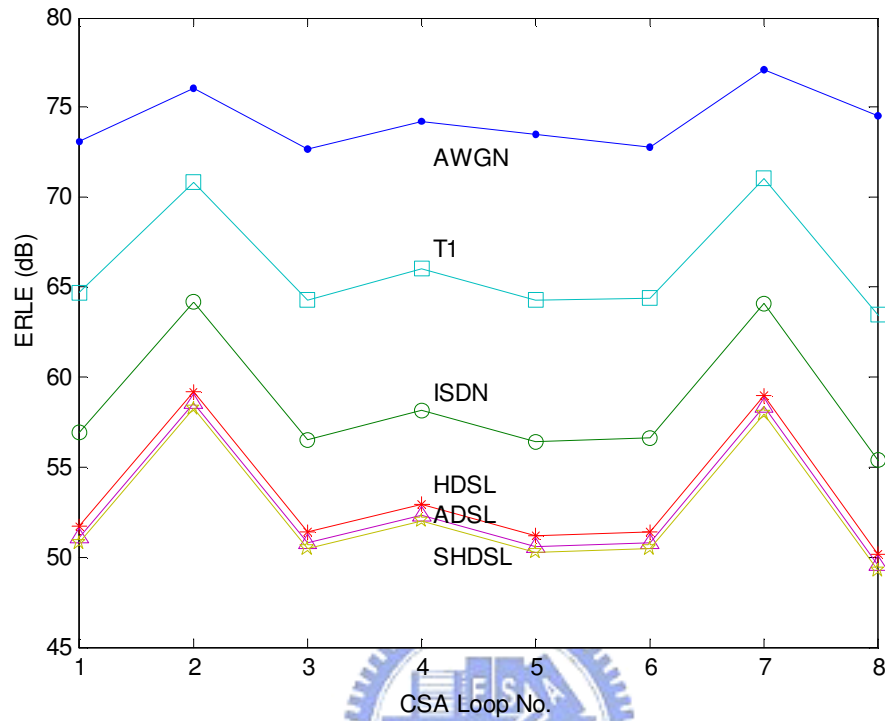


Figure 3.13: Simulated ERLE performance at CPE side.

calculate and list ERLs in Table 3.4 for the scenarios considered above. As the table shows, the value of ERL is between 22.0 and 26.0 dB at the CO side, and 17.0 and 26.0 dB at the CPE side. Because the topology is more complicated at the CPE side, the hybrid circuit usually achieves

Table 3.4: Echo return loss and path loss for CSA loops

| CSA loop No. | 1    | 2    | 3    | 4    | 5    | 6    | 7    | 8    |
|--------------|------|------|------|------|------|------|------|------|
| ERL-CO (dB)  | 24.0 | 24.0 | 25.0 | 22.0 | 24.0 | 25.0 | 26.0 | 26.0 |
| ERL-CPE (dB) | 24.0 | 17.0 | 25.0 | 23.0 | 25.0 | 25.0 | 17.0 | 26.0 |
| PL (dB)      | 32.0 | 30.0 | 33.0 | 32.0 | 32.0 | 34.0 | 31.0 | 33.0 |

smaller ERL. In other words, a higher ERLE must be achieved at the CPE side for a same target SNR at the receiver. We show TERL values in Fig. 3.14. Except for the AWGN case, all TERLs

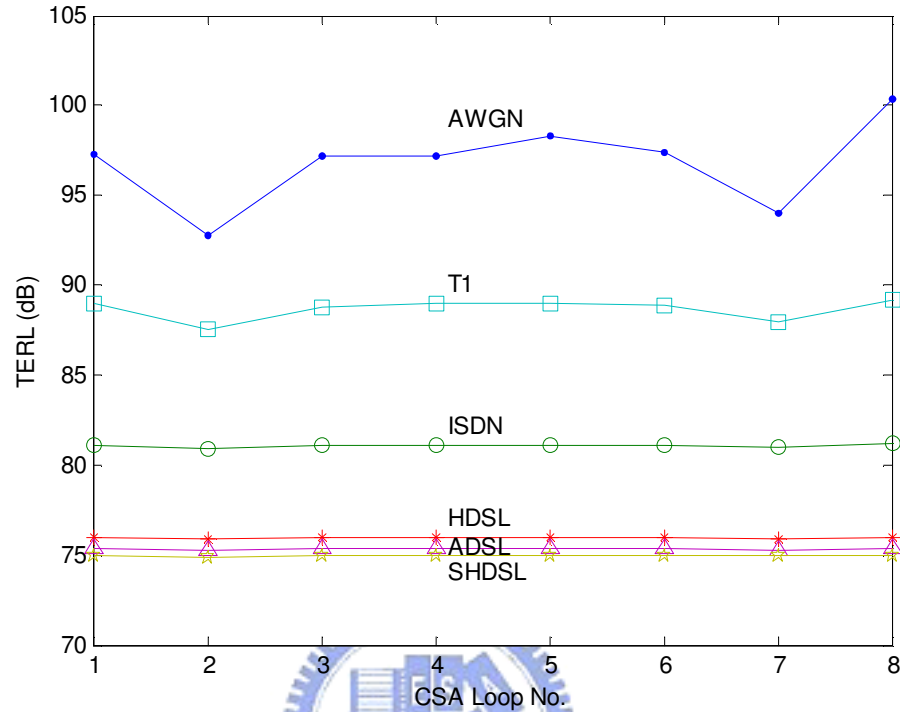


Figure 3.14: Total echo return loss at CPE side.

are quite close. Fig. 3.15 compares PSDs of echo, residual echo and composite noise. As we can see, the residual echo level is below the composite noise in most frequencies. This implies that the echo cancellation level is high enough and a higher ERLE will not improve the system performance. If the channel is perfectly equalized (i.e., intersymbol interference free), the SNR (in dB) after echo canceller can be calculated as follows:

$$\text{SNR} = P_t - [\text{PL} + (N_n + N_e)], \quad (3.41)$$

where  $N_e = P_t - \text{TERL}$ ,  $P_t$  is the transmit power, and equaling 18.9 dBm (-40 dBm/Hz in power spectrum), PL is the path loss of CSA loops as tabulated in Table 3.4,  $N_n$  is the power of the composite noise, and  $N_e$  is the power of the residual echo. Using the above formula, the theoretical SNR, with and without echo interference for CSA #1, is calculated in Table 3.5. For the case with echo interference, the proposed echo canceller was used to cancel the echo

signal. As Table 3.5 shows, both SNRs are very close. This verifies the statement made above that the echo was cancelled to a negligible level. Since the crosstalk noise from 10-SHDSL disturbers is the strongest in the CPE side, the SNR is only 26.8 dB. The highest SNR is 65.4 dB which corresponds to the AWGN case. Finally, the simulated SNR (with echo interference) for all kind of loops and noises is shown in Fig. 3.16. The SNR performance tabulated in Table 3.5 corresponds to one case (for CSA #1) shown in Fig. 3.16. We found that performances for different loops are quite similar. From the simulation and theoretical results shown above, we conclude that the proposed echo canceller can be as effective as the transversal filter echo canceller while at the same time the computational complexity is significantly lower.

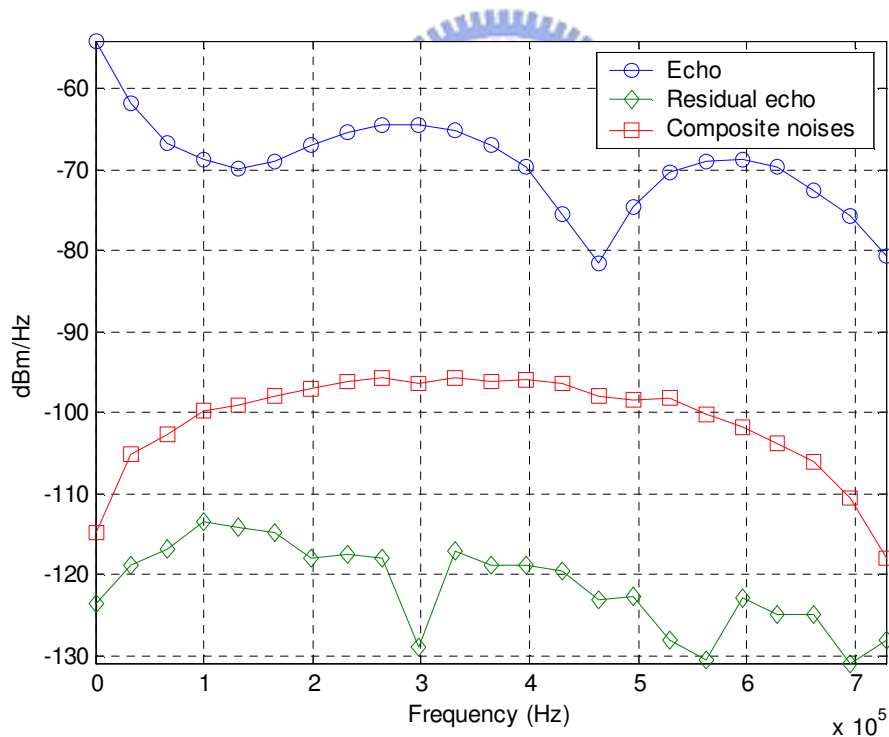


Figure 3.15: The power spectral densities of echo, residual echo, and composite noises.

Table 3.5: SNR with or without echo interference for CSA #1 at CPE side

|         | AWGN | ISDN | HDSL | T1   | ADSL-US | SHDSL |
|---------|------|------|------|------|---------|-------|
| Without | 68.1 | 39.7 | 33.4 | 37.2 | 34.2    | 26.8  |
| With    | 65.4 | 39.7 | 33.4 | 37.2 | 34.2    | 26.8  |

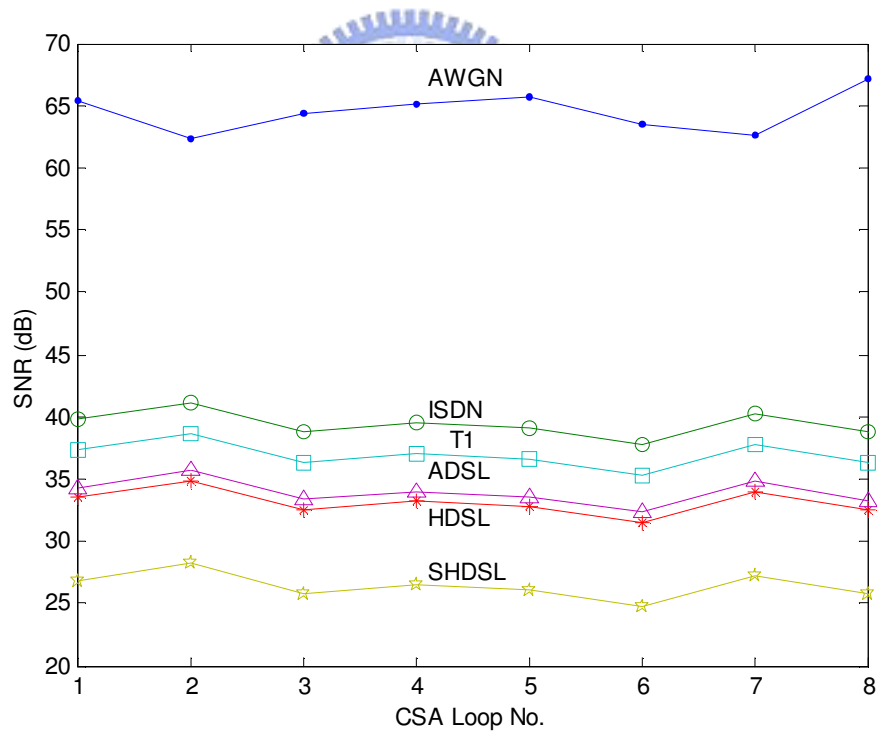


Figure 3.16: SNR (with echo cancellation) at CPE receiver.


## § 3.5 Conclusions

An optimal IFIR echo canceller is proposed for the echo cancellation application in high-speed baseband xDSL systems. The IFIR echo canceller inherits all the numerical stability advantages of the conventional FIR filter while effectively reducing its computational complexity. Optimal interpolation filters for DSL applications are designed using a least-squares method. Finally, extensive simulations using standard test loops were conducted to demonstrate the effectiveness of the optimal IFIR echo canceller.



## Chapter 4

# Interpolated Decision Feedback Equalizer and Precoder



Decision feedback equalization [24] has been widely used as an efficient intersymbol interference (ISI) cancellation technique for several decades. In high-speed wireless and wireline communications, the transmission bandwidth is wider and intersymbol interference is more severe. The DFE then requires tens even hundreds taps in order to cancel the ISI to an acceptable level. This will significantly increase the computational complexity of the DFE. Thus, many methods have been proposed to solve the problem [25–30]. A common idea for complexity reduction is to remove the redundancy commonly observed in the DFE feedback filter response. For wireless applications, the channel may be highly sparse, so is the DFE feedback filter. This characteristic is then used in some reduced complexity DFEs [28–30].

For wireline applications, the channel often possesses strong lowpass characteristics and this makes the channel response look like a smoothly decaying function. In [25], Crespo et al. proposed to use an infinite-impulse-response (IIR) filter as the DFE feedback filter (instead of an FIR filter). This approach only uses a two-pole feedback filter to avoid possible stability problem in the adaptive implementation. Simulation results for a 12-kft 24-gauge twisted-pair copper wire channel were reported. Since the order of the IIR filter is low, this approach

may not yield good performance for all practical channels. Young [26] used a fixed first-order FIR feedforward filter to shorten the digital subscriber line (DSL) channel and assumed that the resultant channel had postcursors only. The resultant channel response was then modelled as an exponential decaying function approximated by the response of a low-order FIR filter cascaded with a first-order IIR filter. Simulation results reported that this approach could have comparable performance with the conventional DFE. Similar to [25], the channel modelling is oversimplified and its applicability may be limited.

In [27], Al-Dhahir et al. used an ARMA-model to describe the channel response and proposed a matrix-based multistage method to derive a low-complexity DFE. The method first estimates the channel impulse response and solves the optimal DFE solution using the fast Cholesky factorization algorithm [31]. Then, it uses low-order IIR filters to model the FIR feedforward and feedback filters, and identifies their coefficients with a generalized ARMA-Levinson algorithm [27]. Finally, the low-order IIR filters is used to implement the DFE in run-time. The problem of this approach is that the preprocessing requires a high computational complexity. Also, the modelling may not be adequate for all DSL channels.

In this dissertation, we focus on the DFE complexity reduction issue in DSL. A typical DSL channel response is shown in Fig. 4.1. The response consists of a short and rapidly changing precursor ISI, and a long and slowly decaying postcursor ISI. As we can see, the channel response is long and highly lowpassed. For a conventional DFE, it is well known that the feedback filter length is on the same order of the postcursor ISI length. Thus, the feedback filter may also require hundreds of tap weights, and the computational complexity of a transversal FIR feedback filter may be high. This problem is more apparent in a precoding system. The purpose of precoding is to avoid error propagation problem inherent in the DFE. A commonly used precoding technique is the Tomlinson-Harashima precoder (THP) [32–34]. The key to avoid error propagation is to place the DFE feedback filter in the transmit side instead of the receive side. Though the coefficients of the THP are the same as the feedback filter coefficients, the input data characteristics are different. The input to a THP has a continuous



value rather than discrete. That means the computation complexity requirement when precoding is used will be much higher than a conventional DFE. Since the length of the feedback filter may be many times greater than that of the feedforward filter, the computational complexity of the DFE (particularly the THP) will be dominated by the feedback filter.

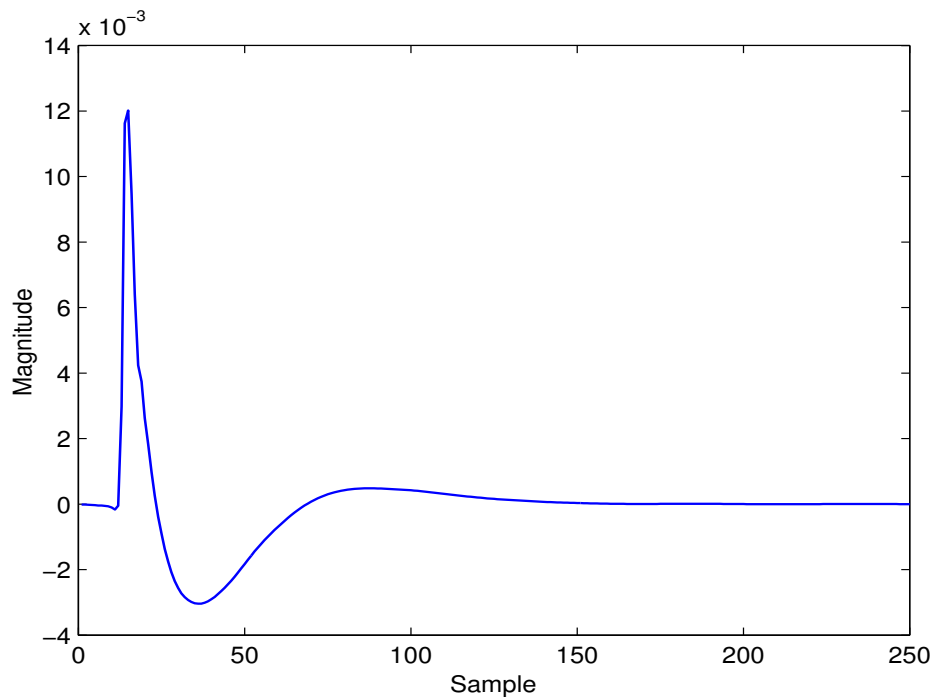
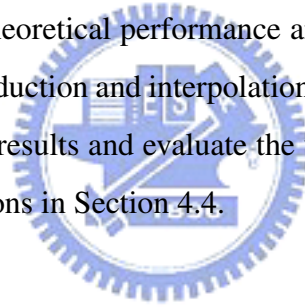


Figure 4.1: Typical channel impulse response in digital subscriber loop application.

Fortunately, it is observed that the response of the DFE feedback filter is similar to that of the postcursor ISI. Thus, the optimal response of the feedback filter is also a slowly decaying function [25–27]. Note that the overall ISI is jointly cancelled by the feedforward and feedbackward filters in a DFE. That means there may exist some sub-optimal pairs of the feedbackward and feedback filter solution that can cancel the ISI without too much performance degradation. This is to say that when the feedback filter response is not able to approach the optimal one, the feedforward filter will try to compensate for that. Using the jointly ISI cancellation property, we propose a new DFE structure called interpolated DFE that can significantly reduce the com-

computational complexity in the feedback filter. As mentioned, the feedback filter response always decays smoothly and an interpolated FIR (IFIR) filter [17, 18] with a small number of coefficients is able to cancel most of the postcursor ISI effectively. We further apply the least mean square (LMS) algorithm to train the tap weights and obtain an adaptive IDFE. Using the similar approach, we also propose a low-complexity interpolated Tomlinson-Harashima precoder (ITHP). The distinct feature of the proposed algorithm, compared to existing ones, is that the feedback filter remains the FIR structure. Thus, the proposed algorithm is free of the stability problem. Note that the idea of the interpolated filtering has been successfully applied in the problem of echo cancellation [15]. However, the problem becomes more involved here.

This chapter is organized as follows. In Section 4.1, we briefly describe the system model, the conventional DFE, and its optimal tap weights solution. In Section 4.2, the proposed IDFE and adaptive IDFE are introduced, theoretical performance and convergence behavior are analyzed in detail, and the complexity reduction and interpolation filter design issues are discussed. In Section 4.3, we report simulation results and evaluate the validity of derived close-form expressions. Finally, we draw conclusions in Section 4.4.



## § 4.1 System Model and Conventional DFE

Let the length of an equivalent discrete response of a DSL channel be  $N_h$  and its response be  $\mathbf{h} = [h_0, h_1, \dots, h_{N_h-1}]^T$ . The received signal corrupted by additive noise can be described as follows:

$$y_k = \sum_{i=0}^{N_h-1} h_i x_{k-i} + n_k \quad (4.1)$$

where  $x_k$  denotes the transmit signal and  $n_k$  denotes the noise. Here, we assume that both  $n_k$  and  $x_k$  have zero means and they are independent each other. A conventional DFE is shown in Fig. 4.2.

Let  $\mathbf{f}$  and  $\mathbf{b}$  be the tap weight vectors of the feedforward and feedback filters and  $N_f$  and  $N_b$

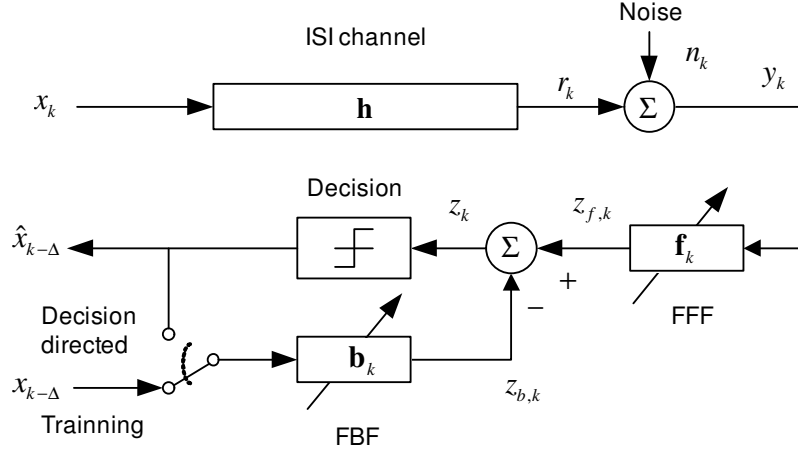


Figure 4.2: Conventional adaptive DFE structure.

be their tap weight lengths, respectively. Then, we have

$$\mathbf{f} = [f_0, f_1, \dots, f_{N_f-1}]^T \quad (4.2)$$

$$\mathbf{b} = [b_1, b_2, \dots, b_{N_b}]^T \quad (4.3)$$

Also, let the input vectors to the feedforward and feedback filters be expressed as

$$\mathbf{y}_k = [y_k, y_{k-1}, \dots, y_{k-N_f+1}]^T \quad (4.4)$$

$$\hat{\mathbf{x}}_k = [\hat{x}_{k-\Delta-1}, \hat{x}_{k-\Delta-2}, \dots, \hat{x}_{k-\Delta-N_b+1}]^T, \quad (4.5)$$

where  $\hat{x}_{k-\Delta}$  is the decision value with the desired delay  $\Delta$ . Then the DFE output value (before decision) is then

$$z_k = z_{f,k} - z_{b,k} \quad (4.6)$$

where  $z_{f,k} = \mathbf{f}^T \mathbf{y}_k$  and  $z_{b,k} = \mathbf{b}^T \hat{\mathbf{x}}_k$  are the output values of the feedforward and feedback filters, respectively. The Wiener solution for  $\mathbf{f}$  and  $\mathbf{b}$  can be derived as follows. Assume that all the decisions are correct, i.e.,  $\hat{x}_{k-\Delta} = x_{k-\Delta}$ ,  $\hat{\mathbf{x}}_k = \mathbf{x}_k = [x_{k-\Delta-1}, x_{k-\Delta-2}, \dots, x_{k-\Delta-N_b+1}]^T$ . Let  $e_k = x_{k-\Delta} - z_k$  be the estimation error. Then, the mean square error (MSE), denoted as  $J$ ,

is

$$\begin{aligned}
J &= E \{e_k^2\} \\
&= E \left\{ \left| x_{k-\Delta} - (\mathbf{f}^T \mathbf{y}_k - \mathbf{b}^T \hat{\mathbf{x}}_k) \right|^2 \right\} \\
&= \sigma_x^2 + \mathbf{f}^T \mathbf{R}_{yy} \mathbf{f} + \mathbf{b}^T \mathbf{R}_{xx} \mathbf{b} - 2\mathbf{f}^T \mathbf{R}_{yx} \mathbf{b} - 2\mathbf{p}_{yx}^T \mathbf{f}
\end{aligned} \tag{4.7}$$

where  $E \{\cdot\}$  denotes the expectation operation,  $\sigma_x^2 = E \{x_k^2\}$  is the transmit signal power,  $\mathbf{R}_{xx} = E \{\hat{\mathbf{x}}_k \hat{\mathbf{x}}_k^T\} = E \{\mathbf{x}_k \mathbf{x}_k^T\}$ ,  $\mathbf{R}_{yy} = E \{\mathbf{y}_k \mathbf{y}_k^T\}$ ,  $\mathbf{R}_{yx} = E \{\mathbf{y}_k \hat{\mathbf{x}}_k^T\}$ , and  $\mathbf{p}_{yx} = E \{\mathbf{y}_k x_{k-\Delta}\}$ .

Taking the partial derivatives of  $J$  with respect to  $\mathbf{f}$  and  $\mathbf{b}$ , we have

$$\frac{\partial J}{\partial \mathbf{f}} = 2\mathbf{R}_{yy} \mathbf{f} - 2\mathbf{R}_{yx} \mathbf{b} - 2\mathbf{p}_{yx} \tag{4.8}$$

$$\frac{\partial J}{\partial \mathbf{b}} = 2\mathbf{R}_{xx} \mathbf{b} - 2\mathbf{R}_{yx}^T \mathbf{f} \tag{4.9}$$

Then, setting both (4.8) and (4.9) to zero, we have the optimal  $\mathbf{f}$  and  $\mathbf{b}$  as

$$\mathbf{f}_{mmse} = (\mathbf{R}_{yy} - \mathbf{R}_{yx} \mathbf{R}_{xx}^{-1} \mathbf{R}_{yx}^T)^{-1} \mathbf{p}_{yx} \tag{4.10}$$

$$\mathbf{b}_{mmse} = \mathbf{R}_{xx}^{-1} \mathbf{R}_{yx}^T \mathbf{f}. \tag{4.11}$$

Applying the above optimal tap weights  $\mathbf{f}_{mmse}$  and  $\mathbf{b}_{mmse}$  into (4.7), we can obtain the minimum MSE (MMSE) of the conventional DFE denoted as  $J_{\min,DFE}$ . The signal to noise ratio (SNR) of the MMSE DFE before decision is found to be [35]

$$\text{SNR}_{DFE,MMSE} = \frac{\sigma_x^2}{J_{\min,DFE}}. \tag{4.12}$$

Let  $n_k$  be white and its variance be  $\sigma_n^2$ . Correlation matrices and the cross-correlation vector in (4.10) can be then expressed as follows:

$$\mathbf{R}_{yy} = \sigma_x^2 \mathbf{H} \mathbf{H}^T + \sigma_n^2 \mathbf{I}_{N_f \times N_f} \tag{4.13}$$

$$\mathbf{R}_{yx} = \sigma_x^2 \mathbf{H} \begin{bmatrix} \mathbf{0}_{N_b \times (\Delta+1)} & \mathbf{I}_{N_b \times N_b} & \mathbf{0}_{N_b \times (N_f + N_h - N_b - \Delta - 2)} \end{bmatrix}^T \tag{4.14}$$

$$\mathbf{p}_{yx} = \sigma_x^2 \mathbf{H} \begin{bmatrix} \mathbf{0}_{N_b \times \Delta} & 1 & \mathbf{0}_{N_b \times (N_f + N_h - \Delta - 2)} \end{bmatrix}^T \tag{4.15}$$

where  $\mathbf{H}$  is the channel matrix given by

$$\mathbf{H} = \begin{bmatrix} h_0 & h_1 & \cdots & h_{N_h-1} & 0 & \cdots & 0 \\ 0 & h_0 & h_1 & \cdots & h_{N_h-1} & 0 & \cdots \\ \vdots & & & & & & \vdots \\ 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{N_h-1} \end{bmatrix}_{N_f \times (N_f + N_h - 1)}. \quad (4.16)$$

The detailed derivation for (4.13)~(4.15) can be found in Appendix A. Substituting (4.13)~(4.15) into (4.10)~(4.12), we can obtain the theoretical SNR and optimal tap weights. In simulations, these expressions will be used to obtain theoretical DFE performance bounds.

## § 4.2 The Proposed Interpolated DFE

In this section, the proposed IDFE will be elaborated in detail. First, we formulate the IDFE structure and derive the Wiener solution as well as the MMSE for the IDFE filter. Using these results, we can calculate the SNR performance bound. Applying the LMS algorithm to the IDFE filter, we can obtain an adaptive IDFE. Then, we analyze the convergence behavior of the adaptive IDFE. This includes the step size bounds and the steady state misadjustment. Finally, the complexity reduction and interpolation filter design issues are discussed.

### § 4.2.1 Interpolated DFE

Usually, the optimal feedback filter response of a conventional DFE is a smoothly decaying function as shown in Fig. 4.3(a). The smooth shape makes it interpolatable and a low-complexity structure possible. As assumed, the feedback filter length of the DFE is  $N_b$ . We can then select a cutting point called  $\alpha$  to separate the optimal feedback filter response into two parts; one is an  $\alpha$ -tap head response and the other is an  $(N_b - \alpha)$ -tap tail response. In general,  $(N_b - \alpha)$  is much larger than  $\alpha$ . As a result, if the tail response is implemented by a low-complexity filter, the overall computational complexity can be reduced dramatically. As

shown in Fig. 4.3(b), we use a low-complexity IFIR filter to model the tail response, and use an FIR filter to model for the head response. Thus, an interpolated feedback equalizer shown in Fig. 4.4 can be constructed. The proposed interpolated DFE consists of a conventional feedforward filter, and an interpolated feedback filter. The interpolated feedback filter consists of a feedback head filter and an interpolated feedback tail filter. Generally, an interpolated filter is a cascade of an interpolation filter and a filter with an upsampled response. Let  $\tilde{\mathbf{f}}$ ,  $\mathbf{b}_1$ ,  $\mathbf{g}$ ,  $\mathbf{b}_2$  be the tap weight vectors of the feedforward, the interpolated feedback tail, the interpolation, and the feedback head filter, respectively. Then, we define

$$\tilde{\mathbf{f}} = [\tilde{f}_0, \tilde{f}_1, \dots, \tilde{f}_{N_f-1}]^T \quad (4.17)$$

$$\mathbf{b}_1 = [b_{1,1}, b_{1,2}, \dots, b_{1,N_{b1}}]^T \quad (4.18)$$

$$\mathbf{g} = [g_{-(M-1)}, g_{-(M-2)}, \dots, g_{(M-1)}]^T \quad (4.19)$$

$$\mathbf{b}_2 = [b_{2,1}, b_{2,2}, \dots, b_{2,N_{b2}}]^T \quad (4.20)$$

where  $M$  is the interpolation factor,  $N_g = 2M - 1$  is the length of the interpolation filter,  $N_{b1}$  is the length of the interpolated feedback tail filter, and  $N_{b2}$  is the length of the feedback head filter. Note that  $N_{b1} = \lfloor \frac{(N_b - \alpha)}{M} \rfloor$ , where  $\lfloor \cdot \rfloor$  takes the nearest integer towards zero. The length of the feedback head filter is then  $N_{b2} = \alpha + (M - 1)$ .

Generally, the impulse response of the interpolation filter  $\mathbf{g}$  is peaking at the center, slowly decaying to its two sides and is symmetric around the center. The simplest response of  $\mathbf{g}$  is a triangular window function with  $2M-1$  taps, which gives a linear interpolation result. Since the impulse response of the interpolated feedback tail filter is the convolution of  $\mathbf{g}$  and  $\mathbf{b}_1^{\uparrow M}$ , where  $\mathbf{b}_1^{\uparrow M} = [b_1, \underbrace{0, \dots, 0}_{M-1}, b_2, \underbrace{0, \dots, 0}_{M-1}, b_3, 0, \dots, b_{N_{b1}}]^T$  is a  $M$ -upsampled response of  $\mathbf{b}_1$ , as shown in Fig. 4.3(b), it exhibits transient responses in two sides, each one decaying to zero. The first one is in the front end of  $\mathbf{g} * \mathbf{b}_1^{\uparrow M}$ , where  $*$  stands for the convolution operation, and the other one is in the tail end. Since the tail of the feedback filter response always decays to zero, there is no problem with the transient response in the tail. But the front-end response of

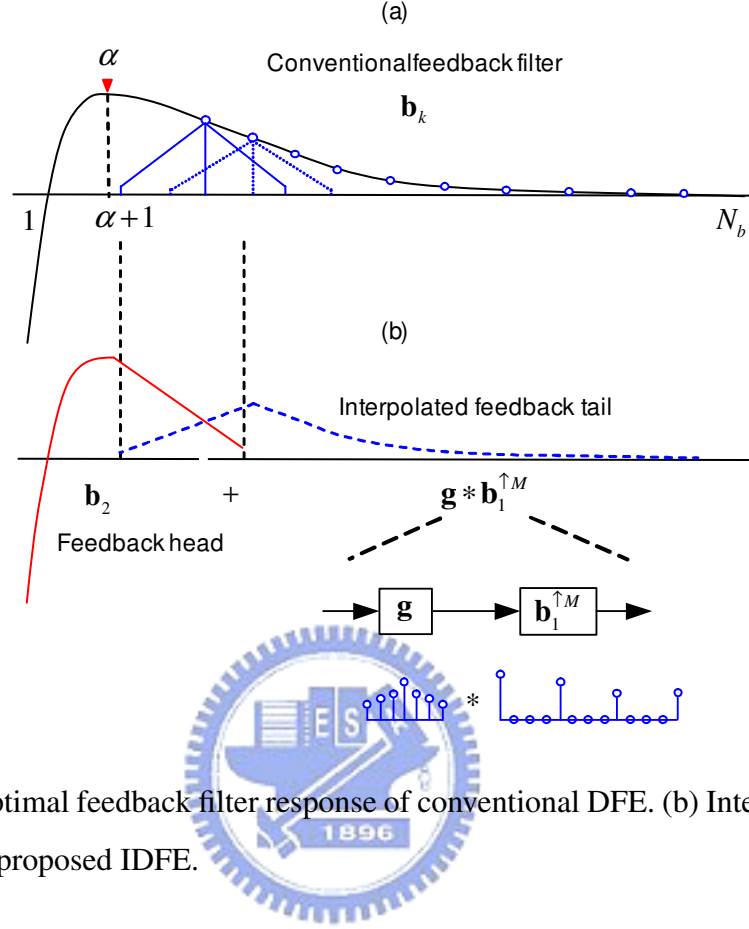


Figure 4.3: (a) Optimal feedback filter response of conventional DFE. (b) Interpolated feedback filter response of proposed IDFE.

a conventional feedback filter usually does not decay to zero and it cannot be modelled by the front-end transient response of  $\mathbf{g} * \mathbf{b}_1^{\uparrow M}$ . One simple way to solve this problem is to use a FIR filter to compensate for the transient response in the front-end. We can then increase the length of the feedback head filter,  $\mathbf{b}_2$ , such that it overlaps its last  $M - 1$  taps with the first  $M - 1$  taps of  $\mathbf{g} * \mathbf{b}_1^{\uparrow M}$ . The combined response of the overlapped effect is illustrated in Fig. 4.3(b). Since the feedback head filter is an FIR filter, the IDFE allows a fast-varying response in the head portion of the feedback filter. This design extends the applicability of the IDFE to the case where the feedback filter has a partial smooth response.

Express the input vectors to the interpolated feedback tail and the feedback head filters as

$$\mathbf{x}_{1,k} = [\tilde{x}_{k-\Delta-(\alpha+1)}, \tilde{x}_{k-\Delta-(\alpha+1)-M}, \dots, \tilde{x}_{k-\Delta-(\alpha+1)-(N_{b_1}-1)M}]^T \quad (4.21)$$

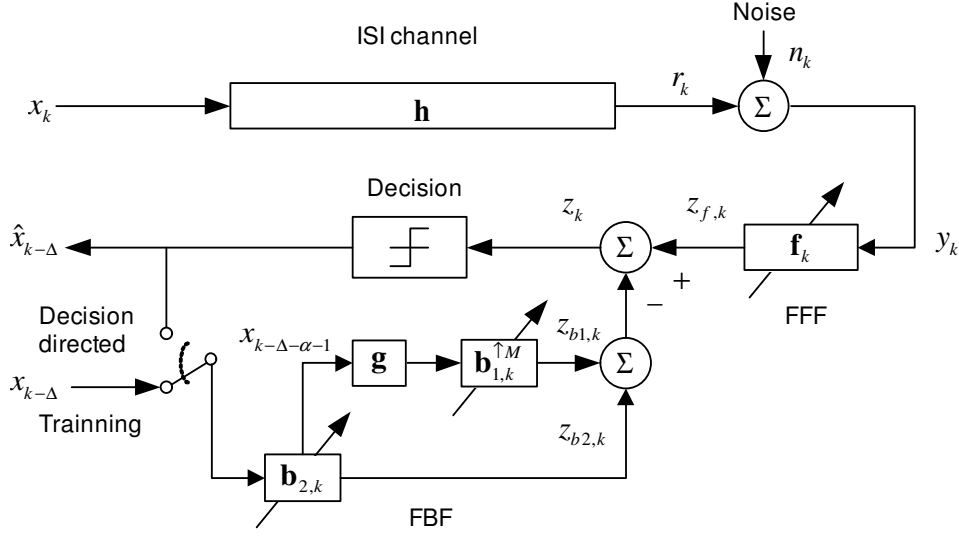


Figure 4.4: Proposed low-complexity adaptive IDFE structure.

$$\mathbf{x}_{2,k} = [\hat{x}_{k-\Delta-1}, \hat{x}_{k-\Delta-2}, \dots, \hat{x}_{k-\Delta-N_{b2}}]^T \quad (4.22)$$

where  $\tilde{x}_k$  is the output from the interpolation filter and

$$\tilde{x}_{k-\Delta-(\alpha+1)} = \sum_{i=-(M-1)}^{(M-1)} g_i \hat{x}_{k-\Delta-\alpha-M-i}. \quad (4.23)$$

Note that  $\mathbf{x}_{1,k}$  is a vector with downsampled  $\tilde{x}_k$ 's as its components. The dimension of  $\mathbf{x}_{1,k}$  is  $N_{b1}$  and that of  $\mathbf{x}_{2,k}$  is  $N_{b2}$ . From (4.17) to (4.23), we then have  $z_{f,k} = \tilde{\mathbf{f}}^T \mathbf{y}_k$  and  $z_{b,k} = \mathbf{b}_1^T \mathbf{x}_{1,k} + \mathbf{b}_2^T \mathbf{x}_{2,k}$ , where  $z_{f,k}$  and  $z_{b,k}$  is the output value of the feedforward and feedback filters, respectively.

Given the interpolation filter  $g$ , the Wiener solution for  $\tilde{\mathbf{f}}$ ,  $\mathbf{b}_1$ , and  $\mathbf{b}_2$  can be derived as follows. Assume that all the decisions are correct, i.e.,  $\hat{x}_{k-\Delta} = x_{k-\Delta}$ . Let  $z_k = z_{f,k} - z_{b,k}$ , and



be the estimation error, the MSE, denoted as  $J$ , is then

$$\begin{aligned}
J &= E \{e_k^2\} \\
&= E \left\{ \left| x_{k-\Delta} - \left[ \tilde{\mathbf{f}}^T \mathbf{y}_k - \mathbf{b}_1^T \mathbf{x}_{1,k} - \mathbf{b}_2^T \mathbf{x}_{2,k} \right] \right|^2 \right\} \\
&= \sigma_x^2 + \tilde{\mathbf{f}}^T \mathbf{R}_{yy} \tilde{\mathbf{f}} + \mathbf{b}_1^T \mathbf{R}_{x_1x_1} \mathbf{b}_1 + \mathbf{b}_2^T \mathbf{R}_{x_2x_2} \mathbf{b}_2 \\
&\quad - 2\tilde{\mathbf{f}}^T \mathbf{R}_{yx_1} \mathbf{b}_1 - 2\tilde{\mathbf{f}}^T \mathbf{R}_{yx_2} \mathbf{b}_2 + 2\mathbf{b}_1^T \mathbf{R}_{x_1x_2} \mathbf{b}_2 - 2\mathbf{p}_{yx}^T \tilde{\mathbf{f}}
\end{aligned} \tag{4.24}$$

where  $\sigma_x^2 = E \{x_k^2\}$  is the transmit signal power,  $\mathbf{R}_{yy} = E \{\mathbf{y}_k \mathbf{y}_k^T\}$ ,  $\mathbf{R}_{x_1x_1} = E \{\mathbf{x}_{1,k} \mathbf{x}_{1,k}^T\}$ ,  $\mathbf{R}_{x_2x_2} = E \{\mathbf{x}_{2,k} \mathbf{x}_{2,k}^T\}$ ,  $\mathbf{R}_{yx_1} = E \{\mathbf{y}_k \mathbf{x}_{1,k}^T\}$ ,  $\mathbf{R}_{yx_2} = E \{\mathbf{y}_k \mathbf{x}_{2,k}^T\}$ ,  $\mathbf{R}_{x_1x_2} = E \{\mathbf{x}_{1,k} \mathbf{x}_{2,k}^T\}$ , and  $\mathbf{p}_{yx} = E \{\mathbf{y}_k x_{k-\Delta}\}$ . Taking the partial derivatives of  $J$  with respect to  $\tilde{\mathbf{f}}$ ,  $\mathbf{b}_1$ , and  $\mathbf{b}_2$ , we have

$$\frac{\partial J}{\partial \tilde{\mathbf{f}}} = 2\mathbf{R}_{yy} \tilde{\mathbf{f}} - 2\mathbf{R}_{yx_1} \mathbf{b}_1 - 2\mathbf{R}_{yx_2} \mathbf{b}_2 - 2\mathbf{p}_{yx} \tag{4.25}$$

$$\frac{\partial J}{\partial \mathbf{b}_1} = 2\mathbf{R}_{x_1x_1} \mathbf{b}_1 - 2\mathbf{R}_{yx_1}^T \tilde{\mathbf{f}} + 2\mathbf{R}_{x_1x_2} \mathbf{b}_2 \tag{4.26}$$

$$\frac{\partial J}{\partial \mathbf{b}_2} = 2\mathbf{R}_{x_2x_2} \mathbf{b}_2 - 2\mathbf{R}_{yx_2}^T \tilde{\mathbf{f}} + 2\mathbf{R}_{x_1x_2}^T \mathbf{b}_1. \tag{4.27}$$

Using a compact representation, we have

$$\begin{bmatrix} \mathbf{R}_{yy} & -\mathbf{R}_{yx_1} & -\mathbf{R}_{yx_2} \\ -\mathbf{R}_{yx_1}^T & \mathbf{R}_{x_1x_1} & \mathbf{R}_{x_1x_2} \\ -\mathbf{R}_{yx_2}^T & \mathbf{R}_{x_1x_2}^T & \mathbf{R}_{x_2x_2} \end{bmatrix} \cdot \begin{bmatrix} \tilde{\mathbf{f}} \\ \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}_{mmse} = \begin{bmatrix} \mathbf{p}_{yx} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}. \tag{4.28}$$

The Wiener solution for the IDFE can be obtained as

$$\begin{bmatrix} \tilde{\mathbf{f}} \\ \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}_{mmse} = \begin{bmatrix} \mathbf{R}_{yy} & -\mathbf{R}_{yx_1} & -\mathbf{R}_{yx_2} \\ -\mathbf{R}_{yx_1}^T & \mathbf{R}_{x_1x_1} & \mathbf{R}_{x_1x_2} \\ -\mathbf{R}_{yx_2}^T & \mathbf{R}_{x_1x_2}^T & \mathbf{R}_{x_2x_2} \end{bmatrix}^{-1} \cdot \begin{bmatrix} \mathbf{p}_{yx} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}. \tag{4.29}$$

Invoking the assumptions made previously, we can obtain the results shown below.

$$\mathbf{R}_{yy} = \sigma_x^2 \mathbf{H} \mathbf{H}^T + \sigma_n^2 \mathbf{I}_{N_f \times N_f} \tag{4.30}$$

$$\mathbf{R}_{x_1x_1} = \sigma_x^2 \mathbf{M} \mathbf{M}^T \tag{4.31}$$

$$\mathbf{R}_{\mathbf{x}_2\mathbf{x}_2} = \sigma_x^2 \mathbf{I}_{(M-1) \times (M-1)} \quad (4.32)$$

$$\mathbf{R}_{\mathbf{y}\mathbf{x}_1} = \sigma_x^2 \mathbf{H} \begin{bmatrix} \mathbf{0}_{(N_b-\alpha) \times (\Delta+1)} & \mathbf{I}_{(N_b-\alpha) \times (N_b-\alpha)} & \mathbf{0}_{(N_b-\alpha) \times (N_f+N_h-(N_b-\alpha)-\Delta-2)} \end{bmatrix}^T \mathbf{M}^T \quad (4.33)$$

$$\mathbf{R}_{\mathbf{y}\mathbf{x}_2} = \sigma_x^2 \mathbf{H} \begin{bmatrix} \mathbf{0}_{(M-1) \times (\Delta+1)} & \mathbf{I}_{(M-1) \times (M-1)} & \mathbf{0}_{(M-1) \times (N_f+N_h-M-\Delta-1)} \end{bmatrix}^T \quad (4.34)$$

$$\mathbf{R}_{\mathbf{x}_1\mathbf{x}_2} = \sigma_x^2 \mathbf{M} \begin{bmatrix} \mathbf{0}_{(M-1) \times \alpha} & \mathbf{I}_{(M-1) \times (M-1)} \\ \mathbf{0}_{(N_b-M+1) \times \alpha} & \mathbf{0}_{(N_b-M+1) \times (M-1)} \end{bmatrix} \quad (4.35)$$

$$\mathbf{p}_{\mathbf{y}x} = \sigma_x^2 \mathbf{H} \begin{bmatrix} \mathbf{0}_{1 \times \Delta} & 1 & \mathbf{0}_{1 \times (N_f+N_h-\Delta-2)} \end{bmatrix}^T \quad (4.36)$$

where

$$\mathbf{M} = \begin{bmatrix} \underbrace{\mathbf{g}^T, 0, \dots, 0}_{(N_{b1}-1)M} \\ \underbrace{0, \dots, 0}_M, \underbrace{\mathbf{g}^T, 0, \dots, 0}_{(N_{b1}-2)M} \\ \vdots \\ \underbrace{0, \dots, 0, \mathbf{g}^T}_{(N_{b1}-1)M} \end{bmatrix} \quad (4.37)$$

is the interpolation-then-decimation matrix and  $\mathbf{H}$  is the channel matrix shown in (4.16). The detailed derivation can be found in Appendix B. Applying the above optimal tap weights  $\tilde{\mathbf{f}}_{mmse}$ ,  $\mathbf{b}_{1,mmse}$  and  $\mathbf{b}_{2,mmse}$  in (4.24), we can then obtain the MMSE of the IDFE, denoted as  $J_{\min, IDFE}$ . The SNR for the MMSE IDFE before decision is then

$$\text{SNR}_{IDFE, MMSE} = \frac{\sigma_x^2}{J_{\min, IDFE}}. \quad (4.38)$$

We will use the above formula to compute the theoretical SNR bounds for the IDFE. Note that when the cutting point equals zero and the interpolation factor equals one, the IDFE degenerates to a conventional DFE.

### § 4.2.2 Adaptive IDFE

Rewrite the inputs and weights of the IDFE in an augmented vector form as

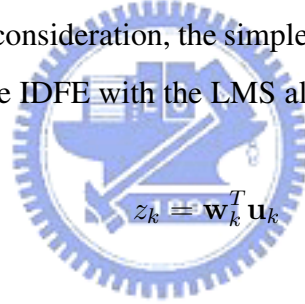
$$\mathbf{u}_k = \begin{bmatrix} \mathbf{y}_k \\ \mathbf{x}_{1,k} \\ \mathbf{x}_{2,k} \end{bmatrix}, \mathbf{w}_k = \begin{bmatrix} \tilde{\mathbf{f}}_k \\ -\mathbf{b}_{1,k} \\ -\mathbf{b}_{2,k} \end{bmatrix}. \quad (4.39)$$

We can then have the DFE output expressed as

$$z_k = \mathbf{w}_k^T \mathbf{u}_k. \quad (4.40)$$

From (4.40), we can see that the proposed IDFE has a linear structure similar to a conventional DFE. As a result, adaptive algorithms developed for the conventional DFE can be directly applied here. For the complexity consideration, the simplest adaptive algorithm, namely the LMS, is employed. Then, the adaptive IDFE with the LMS algorithm can be summarized as follows:

Filter output:



$$z_k = \mathbf{w}_k^T \mathbf{u}_k \quad (4.41)$$

Estimation error:

$$e_k = x_{k-\Delta} - z_k \quad (4.42)$$

Tap-weight adaptation:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \mu e_k \mathbf{u}_k \quad (4.43)$$

where  $\mu$  is the step size controlling the convergence rate. The convergence behavior of the adaptive IDFE will be analyzed below.

#### A) Convergence in the Mean

Using the independence theory [19], we assume that  $\{\mathbf{u}_k\}$  is a sequence of statistically independent vectors. Defining the weight-error vector as  $\varepsilon_k = \mathbf{w}_k - \mathbf{w}_o$ , we then have [19]

$$E[\varepsilon_{k+1}] = (\mathbf{I} - \mu \mathbf{R}_{\mathbf{u}\mathbf{u}}) E[\varepsilon_k], \quad (4.44)$$

where  $\mathbf{R}_{\mathbf{uu}} = E\{u_k u_k^T\}$ . Thus, if the following condition is satisfied, the mean of  $\varepsilon_k$  converges to zero as  $k$  approaches infinity:

$$0 < \mu < \frac{2}{\lambda_{\max}}, \quad (4.45)$$

where  $\lambda_{\max}$  is the largest eigenvalue of the correlation matrix  $\mathbf{R}_{\mathbf{uu}}$ . If a proper step size is chosen in the adaptive IDFE, the convergence can be guaranteed. However, the actual  $\mathbf{R}_{\mathbf{uu}}$  depends on the channel response. A simple way to solve the problem is to use a conservative bound. Since  $\lambda_{\max} \leq \text{tr}[\mathbf{R}_{\mathbf{uu}}]$ , we can then use  $0 < \mu < 2/\text{tr}[\mathbf{R}_{\mathbf{uu}}]$ . Note that  $\text{tr}[\mathbf{R}_{\mathbf{uu}}]$  is just the power of the input signal vector. The proposed adaptive IDFE inherits the same stability advantage as that of the conventional adaptive DFE.

### B) Convergence in the MSE

Let the transient MSE of the adaptive IDFE at the  $k$ -th iteration be expressed as  $J_k = E[e_k^2]$ . The MSE  $J_k$  converges to a steady-state value denoted by  $J_\infty$ , if and only if, the step-size parameter  $\mu$  satisfies the following two conditions [19]:

$$0 < \mu < \frac{2}{\lambda_{\max}}, \quad \sum_{n=1}^{N_f+N_{b1}+N_{b2}} \mu \lambda_n / 2 (1 - \mu \lambda_n) < 1 \quad (4.46)$$

where  $\lambda_n$  is the  $n$ -th eigenvalue of the correlation matrix  $\mathbf{R}_{\mathbf{uu}}$ , and  $N_f + N_{b1} + N_{b2}$  is the number of adjustable tap weights in the adaptive IDFE. The MSE in the steady state value is given by:

$$J_\infty = \frac{J_{\min}}{1 - \sum_{n=1}^{N_f+N_{b1}+N_{b2}} \mu \lambda_n / 2 (1 - \mu \lambda_n)} \quad (4.47)$$

where  $J_{\min}$  is the MMSE yielded by the Wiener solution. The misadjustment  $\psi$ , defined as  $(J_\infty - J_{\min})/J_{\min}$ , is then equal to:

$$\psi = \frac{\sum_{n=1}^{N_f+N_{b1}+N_{b2}} \mu \lambda_n / 2 (1 - \mu \lambda_n)}{1 - \sum_{n=1}^{N_f+N_{b1}+N_{b2}} \mu \lambda_n / 2 (1 - \mu \lambda_n)}. \quad (4.48)$$

From simulations, we found that the convergence rate of the adaptive IDFE is almost the same as the conventional adaptive DFE for the application of single-pair high-speed digital subscriber line (SHDSL). The convergence behavior can be explained as follows. The eigenvalue spread of the correlation matrix in the IDFE is larger than that in the DFE (due to the interpolation) and this will slow the convergence. However, the number of the filter tap weights is significantly smaller and this will accelerate the convergence. These two effects cancel out each other and the convergence rate of the IDFE remains the same as that of the conventional DFE.

The computational complexity of the adaptive IDFE can be easily evaluated. Table 4.1 summarizes the number of the additions and the multiplications required in an adaptive IDFE and a conventional DFE. As we can see, the complexity reduction for the proposed structure comes from the feedback filter. We now use an example to illustrate the low-complexity merit of the proposed IDFE. We define the complexity ratio as the ratio of the computational complexity of the feedback filter in the IDFE over that in the DFE. Consider a DFE used in the SHDSL application where the length of the feedforward filter is 16 and that of the feedback filter is 180. Here, we let the cutting point be zero. Fig. 4.5 shows the complexity ratio for different interpolation factors. As we can see, the complexity ratio can be as low as 24% when  $M$  equals 8. If the computational complexity of the feedforward filter is also taken into account, the ratio will be 31%. Clearly, the IDFE can reduce the complexity effectively.

Table 4.1: Computational complexity analysis

| Complexity                 | Filtering                        |                                  | Taps weight update      |                         |
|----------------------------|----------------------------------|----------------------------------|-------------------------|-------------------------|
|                            | +                                | ×                                | +                       | ×                       |
| Adaptive DFE               | $N_f + N_b - 1$                  | $N_f + N_b$                      | $N_f + N_b$             | $N_f + N_b$             |
| Adaptive IDFE <sup>a</sup> | $N_f + N_{b1} + N_{b2} + 2M - 3$ | $N_f + N_{b1} + N_{b2} + 2M - 1$ | $N_f + N_{b1} + N_{b2}$ | $N_f + N_{b1} + N_{b2}$ |

---

<sup>a</sup> $N_{b1} = \left\lfloor \frac{(N_b - \alpha)}{M} \right\rfloor$ , where  $\lfloor \cdot \rfloor$  is the nearest integer towards zero.

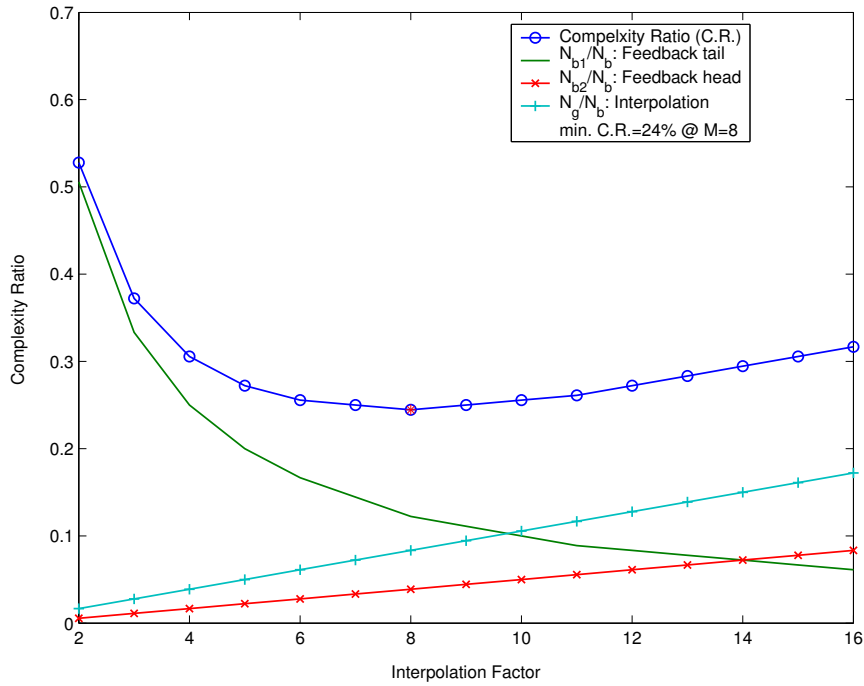


Figure 4.5: Complexity ratio with respect to a conventional DFE,  $(N_f, N_b) = (16, 180)$ . The minimum complexity ratio (C.R.) is 24% when the interpolation factor equals 8.

For the design of the optimal interpolation filter, we may apply the least-squares method proposed in [36]. The idea is to minimize the MSE between an interpolated and the original responses. This method is effective for the echo cancellation application. However, it is not critical here. This is because the feedforward and feedback filters perform channel equalization jointly. If the feedback filter deviates from the optimal due to interpolation, the feedforward filter will compensate for that making the performance loss small. From simulations, we found that the performance of the adaptive IDFE with a simple linear interpolation filter is almost the same as the conventional DFE.

## § 4.3 Simulation Results

In this section, we report some computer simulation results to demonstrate the effectiveness and robustness of the proposed adaptive IDFE. Specifically, we take SHDSL [7–10] as the application example. The channel impulse responses were modelled by the method described in [1]. The channel consist of a transmit shaping filter, a transmit differential hybrid circuit (with a  $135\Omega$  termination impedance), a CSA loop [7], a receive differential hybrid circuit, and a receive filter. The transmit/receive filter was modelled as a 6-th order Butterworth lowpass filter with a 3-dB cutoff frequency at 775 KHz. The primary inductance for the transformer in the hybrid circuit was set as 3 mH. For the SHDSL application, the symbol rate is 775 KHz and the line code is the 8-ary pulse-amplitude-modulation (8-PAM). The channel noise was modelled as additive white Gaussian noise. For a lowest complexity consideration, we let the cutting point be zero, the interpolation factor be eight, and the interpolation filter be linear for all simulation scenarios. We let  $\text{DFE}(N_f, N_b)$  denote a DFE with a  $N_f$ -tap feedforward filter and a  $N_b$ -tap feedback filter, and  $\text{IDFE}(N_f, N_{b1}, M)$  denote an IDFE with a  $N_f$ -tap feedforward filter, a  $N_{b1}$ -tap interpolated feedback filter, and an  $M$ -order interpolation.

### § 4.3.1 Channel CSA No. 3

First, Channel CSA No. 3 [7], which consists of a set of different wire gauges and includes most bridged taps, was used to evaluate the validity of the proposed IDFE. For comparison, a  $\text{DFE}(16,180)$  was also considered. Here, the feedback filter length was determined by the THP coefficients length suggested in standards [7,8] and the feedforward filter length was determined by trial-and-error (for accommodating all channel responses of eight CSA loops). Similar parameters were also used in [37]. For a received signal with SNR 30-dB, an  $\text{IDFE}(16,22,8)$  was used to perform equalization. The delay value  $\Delta$  was set to 16 pointing to the peak of the channel response. Fig. 4.6(a) and Fig. 4.6(b) show the optimal feedforward and feedback filter responses, respectively. The feedbackward filter response of the IDFE was obtained by

$\mathbf{g} * \mathbf{b}_1^{\uparrow M} + \mathbf{b}_2$ . As we can see, the response of the IDFF is similar to that of the DFE (but not exactly the same) and the IDFE successfully equalizes the channel. Coincidentally, the SNR performance for the DFE and the IDFE is about the same (20.4 dB for this loop).

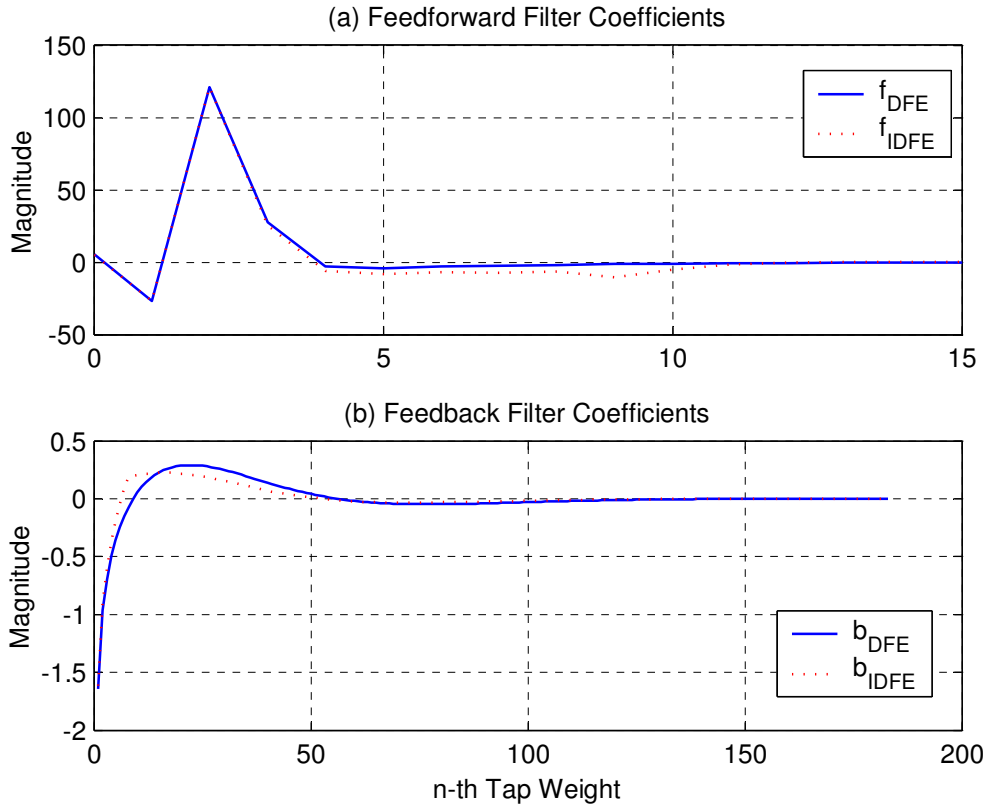


Figure 4.6: (a) Optimal feedforward filter coefficients for IDFE and DFE. (b) Optimal feedback filter coefficients for IDFE and DFE.

### § 4.3.2 Different Loop Topologies

To test the robustness of the IDFE, we used eight CSA loops in [7] for simulations. The theoretical performance results was evaluated using (4.12) and (4.38). The delay value  $\Delta$  was set to 21 here for accommodating all eight different loops. Fig. 4.7(a) shows the performance of the DFE and the IDFE for a received SNR of 40 dB. The averaged SNRs for the DFE and the IDFE are 29.4 dB and 29.1 dB, respectively. Fig. 4.7(b) shows the performance of the DFE and the IDFE



for a received SNR of 20 dB. The averaged SNRs for the DFE and the IDFE are both 11.5 dB. Though the performance is highly depends on the loop, the performance of the IDFE is always close to that of the DFE. For the higher received SNR of 40 dB, the performance difference is only 0.3 dB. For the lower received SNR of 20 dB, the performance of both equalizers is nearly the same.

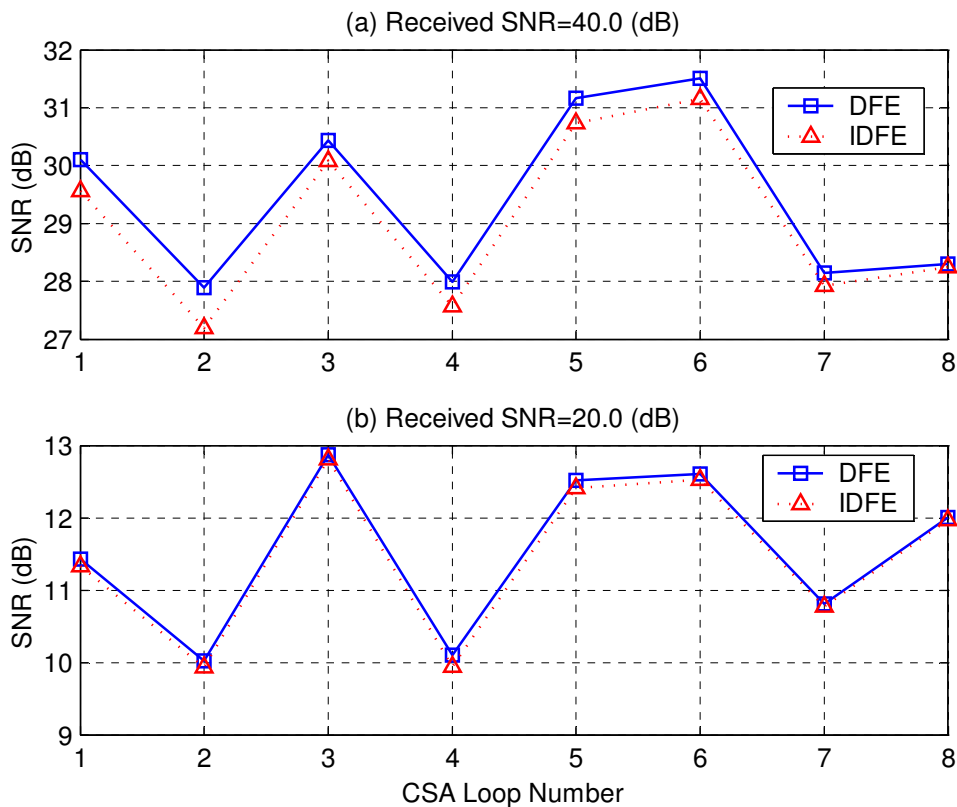


Figure 4.7: (a) Performance of DFE and IDFE for eight CSA loops at received SNR=40 dB. (b) Performance of DFE and IDFE for eight CSA loops at received SNR=20 dB.

### § 4.3.3 The Adaptive IDFE

In this subsection, we will evaluate the validity of derived expressions for the adaptive IDFE. The filter parameters are the same as those used above. We let the step size of the LMS algorithm

be  $1/8/(N_f + N_b) = 1/8/(16+180) = 0.000638$ . In this case, we need 500,000 iterations to achieve convergence. A 9-kft 26 gauge loop (CSA No. 6), which is the longest CSA test loop and usually considered as the worst case in many works, was simulated. The received SNR was assumed to be 30 dB. Fig. 4.8 shows the simulation result. In the figure, the theoretical steady-state MSE,  $J_\infty$ , and the MMSE,  $J_{\min}$ , are indicated with different horizontal lines. The adaptive DFE is also simulated for performance comparison. For easier inspection and comparison, all learning curves were filtered by an averaging filter with a window size of 1,000. From the learning curves, we see that the convergence rate of an adaptive IDFE is almost the same as the adaptive DFE and the convergence behavior is consistent with the theoretical analysis shown in Section 4.2. For a same step size, the simulated steady-state MSEs of the adaptive IDFE and the DFE are both -21.3 dB. Theoretical steady-state MSEs (including the excess MSE) are -21.5 dB and -21.6 dB, respectively, and theoretical MMSEs for the IDFE and the DFE are -21.7 dB and -21.9 dB, respectively. The adaptive IDFE and DFE converge to theoretical bounds within 0.2 dB and 0.4 dB, respectively. These results verify that our theoretical analysis is accurate.

#### § 4.3.4 Symbol Error Rate vs. SNR (for DFE and THP)

In SHDSL applications, equalization is incorporating with the THP to overcome the error propagation problem. Here, two communication systems with or without error propagation were simulated. For the “without error propagation” case, the system consists of a THP at the transmit side, an adaptive IDFE and a modulo remover (for the THP) at the receive side. In this approach, the feedback filtering operation in the DFE is moved to the transmit side. We can first train the IDFE in the receive side and then construct the interpolated feedback filter in the transmit side. This will result in a low-complexity interpolated THP structure. The interpolated THP structure is shown in Fig. 4.9. Again, the complexity is only one fourth of the conventional THP when interpolation factor  $M$  equals eight. In simulations, the adaptive IDFE was trained with 500,000 samples. After convergence, the filter coefficients of the interpolated feedback tail filter  $\mathbf{b}_1$ , and the feedback head filter  $\mathbf{b}_2$  were transmitted back to the transmit side by a reliable

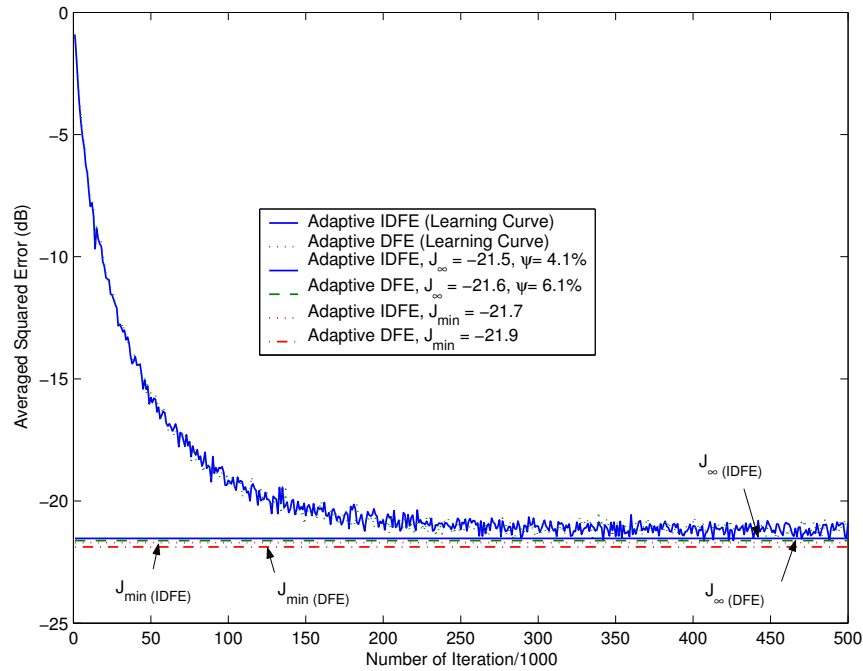


Figure 4.8: Learning curves of adaptive IDFE and DFE with step size 0.000638. Theoretical steady-state MSE ( $J_{\infty}$ ), MMSE ( $J_{\min}$ ), and misadjustment ( $\psi$ ) are also shown.

reverse channel. Note that the filter coefficients of the interpolation filter  $g$  were known *a priori* at both sides. After the training period, an 8-PAM data sequence was then transmitted. A conventional THP (with adaptive DFE) was also simulated for comparison. For the “error propagation” case, the scenario was the same except that the THP was not used. A 9-kft 26 gauge loop (CSA No. 6) was used as the test loop, and scenarios with different received SNRs from 20 to 36 dB were considered. Fig. 4.10 shows the simulation results. From the figure, we can find that the performance of the adaptive IDFE in either case is almost the same as the adaptive DFE. However, the adaptive IDFE requires a much lower computational complexity.

### § 4.3.5 Discussions

The cutting point  $\alpha$  is used to separate the fast and slow varying response of the feedback filter. The larger the  $\alpha$ , the higher the complexity ratio. However, the ability to well model

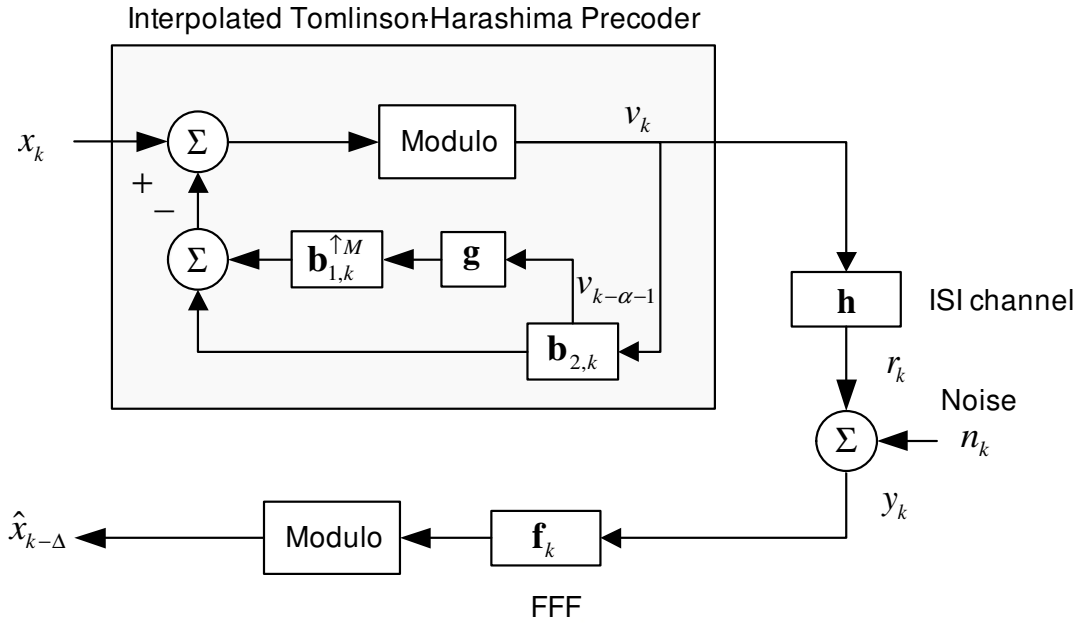


Figure 4.9: Low-complexity interpolated THP (associated with adaptive IDFE)

the head portion response is higher. As long as the response is smooth in the tail region, we can always find a proper  $\alpha$  and reduce complexity without compromising performance. For SHDSL applications, the performance of the IDFE and DFE is nearly the same even when  $\alpha = 0$ . The feedback filter response of the IDFE may exhibit a different shape from that of the DFE. This indicates that the interpolated feedback filter does not approximate the original feedback response. It is easy to deduce that the MSE surface for the DFE has a flat bottom near the Wiener solution. This property makes the performance of the IDFE comparable to that of the DFE.

From our simulation experience, we found other properties with the proposed IDFE. The IDFE is insensitive to the selection of delay value  $\Delta$ . That means a wide range of delay value reaches nearly the same performance. When the length of feedforward filter  $N_f$  becomes larger, the feedback filter response of the IDFE looks more similar to that of the DFE. This is because the feedforward filter can help to compensate for the interpolation error in the feedback filter.

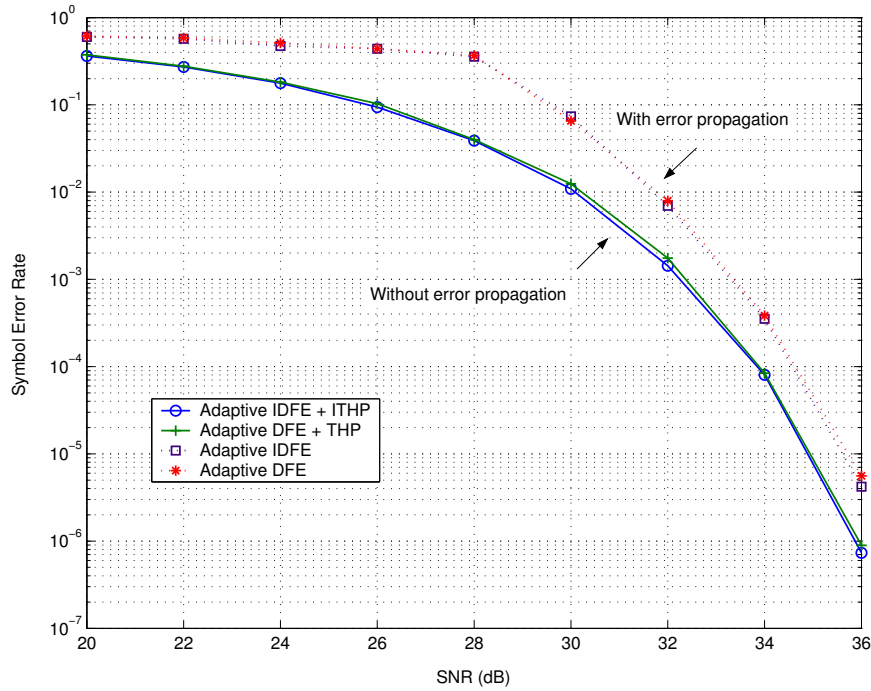


Figure 4.10: Symbol error rate comparison for adaptive IDFE plus ITHP and DFE plus THP (without error propagation), and for adaptive IDFE and DFE (with error propagation).

The other property is that a larger noise level will make the feedback filter response in the DFE smoother. Thus, for a lower received SNR, the feedback filter response of the IDFE approaches closer to that of the DFE.

## § 4.4 Conclusions

Using the idea of filter interpolation, we have proposed a low-complexity and fully FIR-based adaptive filter structure for the high-speed equalization application in DSL. We have shown that the computational complexity reduction can be as high as 76%. We have also derived theoretical expressions regarding the adaptive IDFE and analyzed its convergence behavior. The IDFE approach can also be extended to the THP yielding a high performance yet very efficient equalization scheme. Simulations have shown that the proposed IDFE can have the

same performance as the conventional DFE. Also, derived theoretical expressions predicting the IDFE performance are accurate. Although the proposed IDFE is derived for symbol-rate equalization, it can be generalized to fractionally-spaced (FS) equalization as well. In this case, we have to use a FS feedforward filter instead of a symbol-spaced filter. The feedback filter can then be interpolated using the method discussed above and a low-complexity adaptive FS-IDFE can be obtained.



## Chapter 5

# Fast Interpolated Turbo Equalizer

Intersymbol interference (ISI) and channel noise are the major impairments in a communication system. Typically, the transmitter and receiver in the system must be designed to combat these effects. In the transmitter, there is an error correction encoder adding redundant information for bit protection, and a signal modulator converting the bit stream into a valid analog waveform for transmission. In the receiver, there is an equalizer for ISI compensation and demodulation, and a decoder for error bits correction. For a long period of time, the receiver performs equalization and decoding separately. Although this receiver architecture is popular and works well for many systems, the achievable capacity is far from Shannon's bound. To have better performance or higher capacity, a turbo equalization [38] architecture, conducting equalization and decoding jointly and iteratively, was proposed [39]. The turbo equalizer enhances the performance iteratively with the soft extrinsic information exchange between a soft-in/soft-out (SISO) equalizer and a SISO decoder. The extrinsic information is extracted from the equalizer and the decoder at an iteration and used as *a priori* information in the next iteration. It has been shown that the turbo equalization can greatly enhance the receiver performance and the achievable capacity can help to approach the Shannon's bound.

In 1995, the first turbo equalizer [39] employing a soft output Viterbi algorithm (SOVA) [40] was introduced. In this work, the relationship between original and the encoded bits in the error

correction encoder is constructed as a trellis, so is that between the input and output signals of the channel. Two serially concatenated SOVAs were then used to detect and decode the received signal iteratively. The performance was improved iteratively toward the bound corresponding a coded transmission over an additive white Gaussian noise (AWGN) channel. From a standalone soft equalizer's (or decoder's) point of view, the SOVA is a suboptimal algorithm though it is simpler, and further performance improvement is possible. Later, a turbo equalizer with the maximum *a posteriori* probability (MAP) criterion [41] was reported. The equalizer was realized with the BCJR algorithm which is an optimal SISO processing algorithm for the turbo equalizer. Theoretically, the optimal receiver should consider a single but larger trellis that combines the encoder trellis and channel trellis. However, the size of combined trellis is usually so large that the computational complexity of the BCJR algorithm becomes prohibitive and unrealizable. Thus, theoretical optimal receiver is rarely considered. The optimality for a serially concatenated turbo equalizer is evaluated on a single SISO detector or decoder only. Since both the SOVA and BCJR turbo equalizers are constructed based on the trellis structure, such category of turbo equalizers was classified as the trellis-based turbo equalizer.

The performance of trellis-based turbo equalizer was shown to be excellent but the computational complexity is a big penalty. The complexity depends on the number of processing iterations, the signal constellation size, the memory length of the encoder, and the memory length of the channel. The main problem is that the complexity increases exponentially with the memory length of the encoder and that of the channel. While we can control the memory length of the encoder, we cannot control that of the channel. For a channel with medium or large delay spread, it is easy for the complexity to become impractically high.

The high computational complexity problem of the optimal trellis-based turbo equalizers motivated the study of low complexity suboptimal turbo equalizers. In 1999, Wang *et al.* [42] proposed a lower complexity filter-based turbo interference canceller, which consists of a soft multiuser detector and a soft channel decoder, for the multiuser coded CDMA system. Instead of using complex SOVA or BCJR algorithm as the multiuser detector, a low-complexity



filter-based soft multiuser detector was developed. Based on this concept, a low-complexity filter-based turbo equalizer was first proposed in [43]. Although this approach does have lower computational complexity, the optimal filter weights are time-varying (even the channel is time-invariant) and they are updated per symbol. The complexity for the optimal weights calculation is still high.

Motivated by [42], Tüchler *et al.* [44–46] also proposed a category of filter-based turbo equalizers with much lower complexity than ever. The soft equalizer in their works modifies the conventional linear equalizer or the decision feedback equalizer (DFE) with soft inputs. It is well known that the optimal filter weights [35, 47] of a linear equalizer or DFE equalizer is a function of channel response and signal-to-noise ratio (SNR), and can be solved with corresponding Wiener equations. However, the Wiener solution involves matrix inversion operations. For a direct matrix inversion, the computational complexity is on the order of  $O(N^3)$ , where  $N$  is the filter length. The complexity is still high especially when the filter tap weights are updated per symbol. In [44], a time-recursive updated algorithm for calculating the optimal time-varying filter was proposed; the complexity was reduced to the order of  $O(N^2)$ . Exploring special matrix properties in a time-varying soft equalizer, a less complexity block-invariant sub-optimal soft equalizer was proposed in [45] and the complexity is further reduced to the order of  $O(N)$ . Later, in [46], three different kinds of low-complexity time-invariant soft equalizer were proposed. The complexities are on the same order with [45], but the convergence behaviors are quite different.

The adaptive turbo equalizer [48–52] is another low-complexity alternative. The efficiency of an adaptive turbo equalizer depends on the convergence behavior of the adaptive algorithm. The convergence of the adaptive algorithm can be very slow for a long-response channel such as the wireline channel. The training period, which is usually thousands time of the received block length, becomes much longer than the equalization period. Thus, the adaptive turbo equalizer cannot be directly used in the application. The frequency-domain turbo equalizers were also studied extensively [53–56]. Its complexity is usually an order of magnitude less than

a time-domain turbo equalizer. However, it is only valid for the single-carrier frequency-domain equalization (SC-FDE) system, which is a block-by-block transmission communication system with periodically inserted guard intervals. It cannot be applied to the conventional single carrier modulation system such as wireline SHDSL and wireless GSM, etc.

In this dissertation, we focus on the problem of time-domain turbo equalization over static frequency selective channels. We propose a fast turbo equalizer with complexity an order of magnitude less than the conventional. The proposed algorithm is based on the structure of [45] in which the most computationally intensive operations are the optimal filter coefficients calculation and filtering (equalization). We explore the characteristics of the wireline channel responses and the optimal equalizers and propose interpolation schemes to reduce the complexity. We found that the relationship between an optimal filter coefficient and reliability information, which indicates the correctness of soft bits from the decoder, is nearly one-to-one and monotonic. Once the reliability is estimated, the corresponding optimal filter response can be calculated easily by interpolating two pre-calculated known optimal filter responses. In other words, we can bypass the computationally intensive matrix operations to obtain optimal filter coefficients. Since the reliability function ranges from 0 to 1, we only have to pre-calculate a small number of filter coefficient sets corresponding to specific reliability values and store those sets during initialization. Then, we can interpolate all possible optimal filters in run-time. This will dramatically reduce the computational requirement of the filter-based turbo equalizer. If the channel response is long and changing smoothly, we can apply the interpolated filtering [15] to reduce the complexity further. Combining above schemes, we are able to obtain a fast interpolated turbo equalizer required very low-complexity. Simulation results show that while the computational complexity is reduced dramatically, the performance of the proposed algorithm is almost not affected; it is about the same as that in [45].

This chapter is organized as follows. In Section 5.1, we briefly describe the system model and define some notations. In Section 5.2, the optimal BCJR turbo equalizer and suboptimal filter-based turbo equalizers are summarized. In Section 5.3, the proposed fast interpolated

turbo equalizer is introduced, the interpolation method for optimal filters and interpolated filtering coefficients are derived, and the computational complexity is also analyzed. In Section 5.4, the simulation results are reported and comparison to existing solutions are also made. Finally, we draw conclusions in Section 5.5.

## § 5.1 System Model and Notations

The system model of turbo equalization signal processing for coded data transmitted over ISI channel is shown in Fig. 5.1. The transmitter shown in the upper part consists of a convolutional encoder, an interleaver, and a signal modulator. The information bit stream  $a_m$  is encoded with a binary convolutional encoder to generate the coded bits stream  $b_{m,l}$ , where the subscript stands for the  $l$ -th bit of the  $m$ -th codeword and  $l \in \{1, 2, \dots, r\}$  for a code with rate  $1/r$  ( $r \in \mathbb{Z}, r \geq 2$ ). In other words, an information bit  $a_m$  will generate  $r$  coded bits  $\{b_{m,1}, b_{m,2}, \dots, b_{m,r}\}$ . Then,  $b_{m,l}$  is fed into the interleaver bit-by-bit with the order of  $(m, l) = (1, 1), (1, 2), \dots, (1, r), (2, 1), (2, 2), \dots$ . The interleaver permutes and randomizes  $b_{m,l}$  and yields an independent interleaved bits stream  $c_{k,j}$ , where the subscript stands for the  $j$ -th bit of the  $k$ -th modulated symbol and  $j$  is in  $\{1, 2, \dots, \rho\}$  for a  $\rho$ -bit modulator. That is,  $\rho$  interleaved bits are grouped and mapped to a transmit symbol. Note that the interleaver considered here is a bit interleaver [57] and it also performs the grouping operation for the modulator. Generally, the last error correction codeword may not form a complete symbol and zero padding may be necessary in that case.

The function of a modulator includes symbol mapping and analog waveform conversion. The symbol mapping function is denoted by  $\mathbf{m}_i \leftrightarrow \chi_i$ , where  $\mathbf{m}_i = [m_{i,1}, m_{i,2}, \dots, m_{i,\rho}]^T$  is a  $\rho$ -bit vector and  $\chi_i$  is a symbol belonging to one of the  $M$ -ary signal constellation set  $X$ . Here,  $M = 2^\rho$  and  $X = \{\chi_0, \chi_1, \dots, \chi_{M-1}\}$ . Let the interleaved codeword be  $\mathbf{c}_k = [c_{k,1}, c_{k,2}, \dots, c_{k,\rho}]^T$ . Then,  $\mathbf{c}_k$  is mapped to a symbol  $x_k = \chi_i$  and converted to an analog waveform transmitted over the ISI channel. Except for the ISI effect, the channel also introduces

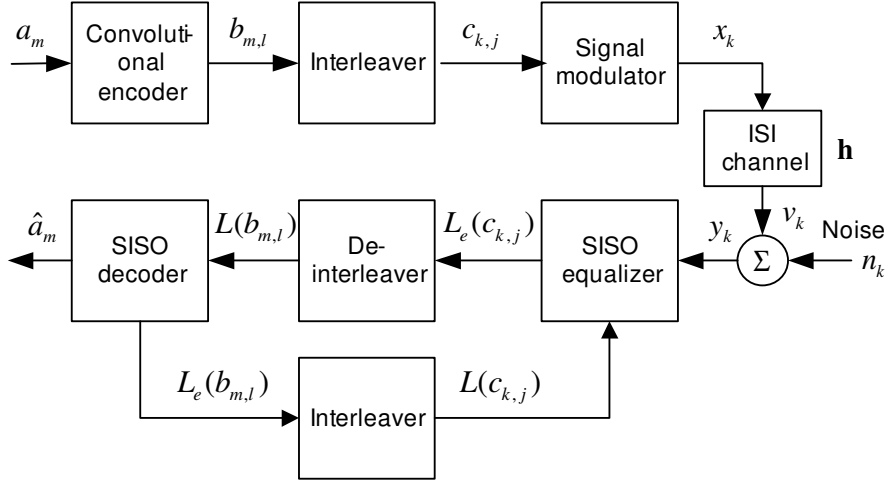


Figure 5.1: The system model of turbo equalization for coded data transmitted over ISI channel.

AWGN,  $n_k$ . In the subject of turbo equalization, most of the previous works focus on wireless or magnetic channels. In this dissertation, we will focus on the wireline communication channel. Here, we assume that the baseband modulation scheme is the M-ary pulse amplitude modulation (M-PAM). The symbol mapping for M-PAM can be defined as

$$\chi_i = \frac{(M-1) - 2i}{\sqrt{\frac{1}{M} \cdot \sum_{j=0}^{M-1} |(M-1) - 2j|^2}}, \quad (5.1)$$

where  $i = \sum_{j=1}^{\rho} (m_{i,j} \cdot 2^{\rho-j})$ . As we can see, the symbol power is normalized to one. We also assume that the channel impulse response and the noise are also real valued. For complex-valued passband communication systems such as QAM and PSK, please see [45,58] for details.

The receiver shown in the lower part of Fig. 5.1 consists of a SISO equalizer, a deinterleaver, a SISO decoder, and an interleaver. At first iteration, the SISO equalizer reduces the ISI channel effect without the assistance of any *a priori* information, and it outputs the soft decision  $L_e(c_{k,j})$  as the demodulated data for the SISO decoder. A deinterleaver is inserted on the forward path to permute the demodulated data stream back into the original coded bit stream. The interleaver design is important and closely related with the performance of the

turbo equalizer. We use a random interleaver called S-random interleaver [59] in this dissertation. The main characteristic of the interleaver is that it can guarantee a minimum distance of two consecutive bits after interleaving. Then, the SISO decoder performs the error correcting function on the deinterleaved bits  $L(b_{m,l})$  and generates a series of more reliable soft decisions outputs  $L_e(b_{m,l})$ . These outputs are interleaved to yield  $L_e(c_{k,j})$  feedback to the SISO equalizer as *a priori* information. Note here that the so-called intrinsic information  $L(b_{m,l})$  and  $L(c_{k,j})$  have been subtracted before yielding the extrinsic information  $L_e(b_{m,l})$  and  $L_e(c_{k,j})$ , respectively. The process can be repeated until a stopping criterion is met. It has been found that there exists a certain SNR threshold [60] which the turbo equalizer can work properly. When the SNR is higher the threshold, the performance can be improved iteratively. However, if the SNR is below the threshold, no performance gain can be obtained.

It is well known that the optimal turbo equalizer uses the BCJR algorithm as the SISO processing unit for both the equalizer and the decoder. For convenient comparison with previous works [45], the SISO decoder is also implemented with the BCJR algorithm in this dissertation. The computational complexity of the BCJR SISO equalizer is on the order of  $O(M^\mu)$ , which grows exponentially with the channel memory length  $\mu$  [58]. Similarly, the complexity of a BCJR SISO decoder grows exponentially with the encoder memory length. Usually the memory length of a convolutional code is constrained on a length compromising the computational complexity with performance gain. However, the channel memory length depends on applications having a large variation; it ranges from few taps in wireless systems to hundreds of taps in wireline systems. It is apparent that with existing results, turbo equalization is difficult to apply for systems with long channel responses. The algorithms developed in this chapter is aimed to solve the problem.

Since turbo equalization is an iterative processing scheme, it must operate with a block-by-block manner. Here, we let the block length be equal to  $K_c$  bits, where  $K_c = rK_i + K_o$ ,  $K_i$  is the information bits,  $K_o$  is the overhead needed to terminate the trellis states, and  $1/r$  ( $r \in \mathbb{Z}$ ,  $r \geq 2$ ) is the rate of convolutional code. The trellis state of the convolutional code is both

initialized and terminated to zero state but the equalizer is left unterminated when the BCJR algorithm is applied.

For convenience of later use, following notations are pre-defined.  $\Pr(\cdot)$  is the probability function,  $p(\cdot)$  is the probability density function (PDF),  $L(\cdot)$  is the log-likelihood ratio (LLR) function,  $\ln(\cdot)$  is the natural logarithm function,  $\max(\cdot)$  is the maximum function,  $\text{sign}(\cdot)$  is the sign function,  $E(\cdot)$  is the expectation function,  $\text{cov}(\mathbf{x}, \mathbf{y}) \triangleq E(\mathbf{x}\mathbf{y}^T) - E(\mathbf{x})E(\mathbf{y}^T)$  is the covariance matrix function,  $\text{diag}[v_1, v_2, \dots]$  is a diagonal matrix with the vector elements in the diagonal, and  $\|\mathbf{x}\| = \sqrt{\sum |x_i|^2}$  is the Euclidean vector norm (or 2-norm).

## § 5.2 Summary of Previous Works

Before the development our interpolated turbo equalizer, we first review and summarize the existing turbo equalization schemes. Starting with the basic SISO processing unit, the BCJR algorithm, we describe the conventional turbo equalizer. Then, we continue to the subject of the filter-based and other low-complexity approaches.

### § 5.2.1 The BCJR Equalizer/Decoder

#### A) Trellis Diagram Representation of an ISI Channel

In the BCJR algorithm, the relationship between the input and output of an ISI channel or a convolutional encoder is described with a trellis diagram. For a discrete-equivalent ISI channel shown in Fig. 5.2, the received signal  $y_k$  corrupted by additive noise can be described as follows:

$$y_k = \sum_{i=0}^{\mu} h_i x_{k-i} + n_k \quad (5.2)$$

where  $\mathbf{h} = [h_0, h_1, \dots, h_{\mu}]^T$  denotes the real-valued channel with *memory* length  $\mu$  (note that  $L = \mu + 1$  is defined as the channel *response* length),  $x_k$  with a unit variance denotes the transmit signal, and  $n_k$  with a variance of  $\sigma_n^2$  denotes the AWGN. Here, we assume that both  $n_k$  and  $x_k$  have zero means and they are independent each other.

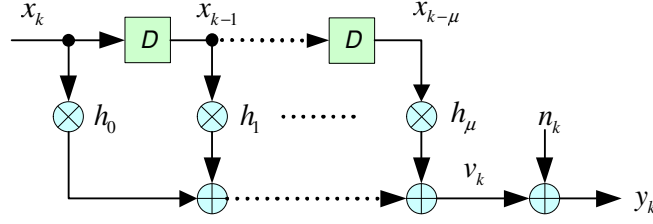


Figure 5.2: A discrete-equivalent ISI channel with memory length  $\mu$

Define the values in the channel memory at time  $k$  as a  $\mu$ -tuple,  $(x_{k-1}, x_{k-2}, \dots, x_{k-\mu})$ . We then have  $Q$  possible combination of the  $\mu$ -tuple where  $Q = M^\mu$ . We call each of the  $Q$  possible combinations as a channel state or just state. We label the states from 0 to  $Q - 1$ , with state 0 reserved as the idle state. Usually, the idle state is defined as the state with all values in the memory are equals zeros. Let  $s_k$  be the state label at time instant  $k$ . The diagram shown in Fig. 5.3 is defined as a state transition diagram at time  $k$ . In the diagram, one generic state transition (or branch) denoted as a two-tuple  $(p, q)$  is shown as a connection between these states from the state  $p$  at time  $k$  to state  $q$  at time  $k + 1$ . Meanwhile,  $x_{(p,q)} / v_{(p,q)}$  denotes the associated channel input/output pair. Note that at the time instant  $k$ ,  $x_k = x_{(p,q)} \in X$  and  $v_k = v_{(p,q)} = \sum_{i=0}^{\mu} h_i x_{k-i}$ . Thus, given a sequence of inputs and outputs, we can have a sequence of state transitions. Combining all state transitions, we can then form a trellis diagram.

## B) Trellis Diagram Representation of a Convolutional Code

A convolutional encoder with rate  $1/r$  and the memory length  $\eta$  can be described as follows:

$$b_{m,l} = \sum_{i=0}^{\eta} q_{i,l} a_{m-i}, l = \{1, 2, \dots, r\}, \quad (5.3)$$

where  $b_{m,l}$  is the  $l$ -th bit of the  $m$ -th codeword,  $q_{i,l} \in \{0, 1\}$  denotes  $l$ -th convolutional code response, and  $a_m$  is the input bits. Here, we assume that  $a_m$ 's are independent and identically distributed (i.i.d.) binary bits. An example code with rate  $1/2$  is shown in Fig. 5.4. Note that addition is performed with the binary field operation, i.e., a logic exclusive "OR".

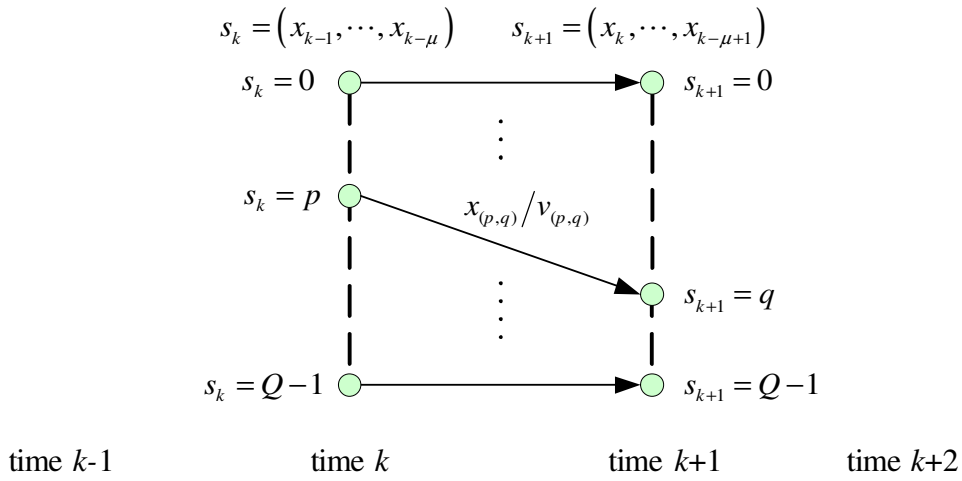


Figure 5.3: The state transition diagram for the ISI channel shown in Fig. 5.2, where  $Q = M^\mu$  is total numbers of states for the channel.

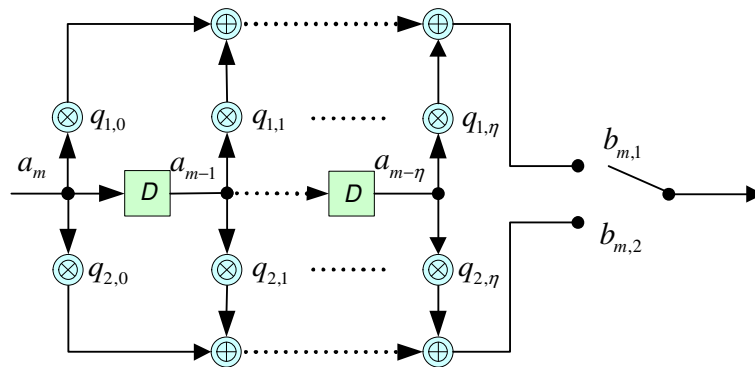


Figure 5.4: A half rate convolutional encoder with memory length of  $\eta$



Similar to the case in ISI channel, we define a specific combination in a  $\eta$ -tuple of  $(a_{m-1}, a_{m-2}, \dots, a_{m-\eta})$  as the encoder state. Thus, we have  $P = 2^\eta$  possible states and label them from state 0 through state  $P - 1$ . State 0 is reserved as the idle state in which all input values are zeros. We assume that the states outside a transmitted data block are terminated with idle states. Let  $s_m$  denote the state label at time  $m$  and the state transition diagram of the encoder is shown in Fig. 5.5 in which  $a_{(p,q)}/\mathbf{b}_{(p,q)}$  denotes the associated input/output pair. Note that  $\mathbf{b}_m = [b_{m,1}, b_{m,2}, \dots, b_{m,r}]^T$  is the  $m$ -th coded codeword. Given a sequence of input and output bits, we can then have a sequence of state transitions from which we can construct a trellis diagram.

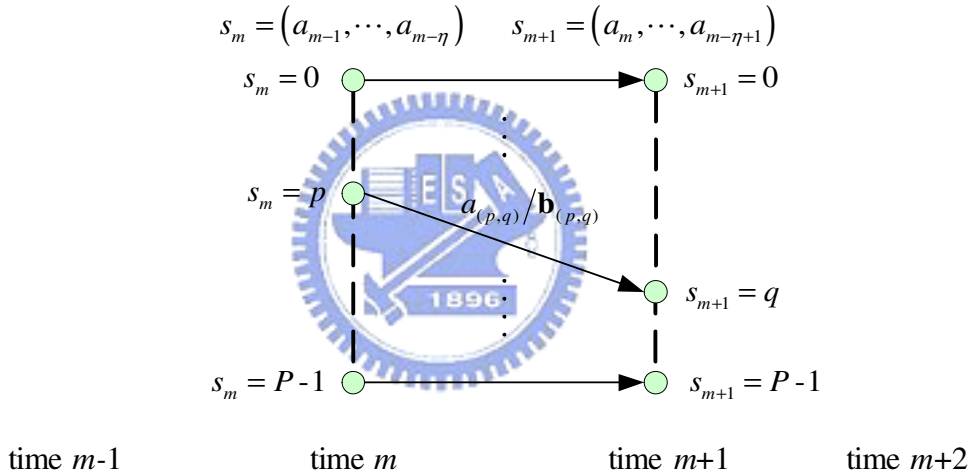


Figure 5.5: The state transition diagram of the convolutional encoder shown in Fig. 5.4, where  $P = 2^\eta$  is total numbers of states for the code.

### § 5.2.2 The BCJR Equalizer

Two kinds of BCJR algorithms are well-known; one is expressed with the probabilistic form and the other is with the logarithmic form. The original BCJR algorithm [61] was expressed in the probabilistic form. The dynamic range of numerical values is larger and the computational complexity is higher. The logarithmic BCJR algorithm is equivalent to the original BCJR al-

gorithm and it has lower computational complexity and better numerical properties. Thus, we use the logarithmic BCJR (simply named as BCJR without ambiguity) algorithm in our development. The major function of the BCJR algorithm involves the calculation of branch metrics, forward/backward state probabilities, the *a posteriori* probabilities (APP), and hard decisions. We summarize the algorithm below and leave the derivation details in Appendix C.

Consider a block of symbols  $\mathbf{x} = [x_0, x_1, \dots, x_{K-1}]^T$  sent by the transmitter. Here,  $K$  is the block length and it can be calculated as  $K = \lceil K_c/\rho \rceil$ , where  $\lceil \cdot \rceil$  denotes the minimum integer operation (toward infinity). Thus, it may be necessary to pad the last codeword with zero bits. Assume that the idle symbols are transmitted outside the block (i.e., for all  $k < 0$  and  $k > K - 1$ ). At the receiver, the corresponding data block  $\mathbf{y} = [y_0, y_1, \dots, y_{K+\mu-1}]^T$  is received and buffered.

For a given state transition ( $s_k = p, s_{k+1} = q$ ) starting at time  $k$ , we first define the BCJR related variables as  $\gamma_k(p, q)$ , branch metric associated with the state transition from state  $p$  at time  $k$  to state  $q$  at time  $k + 1$ ,  $\alpha_k(p)$ , forward state probability on state  $p$ , and  $\beta_k(p)$ , backward state probability on state  $p$ . Then, the algorithm of the BCJR equalizer is summarized as follows:

1. Calculate logarithmic branch metrics :

$$\ln \gamma_k(p, q) = \frac{-|y_k - v_{(p,q)}|^2}{2\sigma_n^2} + K_\gamma, \quad (5.4)$$

where  $K_\gamma = \ln(\Pr(x_k = a_{(p,q)})) - \ln(2\pi\sigma_n^2)$ , for  $p = 0, 1, \dots, Q-1, q = 0, 1, \dots, Q-1, k = 0, 1, \dots, K + \mu - 1$ . Note that  $K_\gamma$  is a constant and can be omitted if the transmitted symbol is equally probable. The probability  $\Pr(x_k = x_{(p,q)})$ , which comes from the SISO decoder, serves as the soft input of the SISO equalizer.

2. Calculate the forward/backward state probabilities:

$$\ln \alpha_{k+1}(q) = \ln \sum_{p=0}^{Q-1} \exp[\ln r_k(p, q) + \ln \alpha_k(p)], \quad (5.5)$$

$$\ln \beta_k(p) = \ln \sum_{q=0}^{Q-1} \exp[\ln r_k(p, q) + \ln \beta_{k+1}(q)]. \quad (5.6)$$

The forward state probability is initialized with the idle state, i.e.,  $\ln \alpha_0 = [0, -\infty, \dots, -\infty]$ , while the backward state probability is with an unknown state, i.e.,  $\ln \beta_{K+\mu} = [-\ln(M), -\ln(M), \dots, -\ln(M)]^T$ .

3. Normalize the forward/backward state probabilities:

$$\alpha_k(p) = \alpha_k(p) - \max \{ \alpha_k(p) \}, \quad (5.7)$$

$$\beta_k(p) = \beta_k(p) - \max \{ \beta_k(p) \}. \quad (5.8)$$

This step will limit the maximum probability to avoid numerical overflow.

4. Calculate APPs for every signal constellation ( $x_k = \chi, \chi \in X$ ):

$$\ln \Pr(x_k = \chi | \mathbf{y}) \triangleq \ln \sum_{(p,q) \in B_\chi} \exp [\ln \alpha_k(p) + \ln \gamma_k(p, q) + \ln \beta_{k+1}(q)]. \quad (5.9)$$

where  $B_\chi$  is the subset of state transitions for the input symbol equals  $\chi$ . This quantity is calculated for every  $\chi$  in  $X$ , and for each input symbols from  $x_0$  to  $x_{K+\mu-1}$ . Note that the constant term  $-\ln p(\mathbf{y})$  is omitted in (5.9) for simplicity.

5. Select the symbol corresponds to the maximum APP value (hard decision output):

$$\hat{x}_k = \max \{ \ln \Pr(x_k = \chi | \mathbf{y}) \}. \quad (5.10)$$

For BPSK modulation, we can define the log-likelihood ratio (LLR) as

$$L(x_k) \triangleq \ln \Pr(x_k = +1 | \mathbf{y}) - \ln \Pr(x_k = -1 | \mathbf{y}). \quad (5.11)$$

Then, (5.10) can be simplified as

$$\hat{x}_k = \text{sign}(L(x_k)). \quad (5.12)$$

### § 5.2.3 The BCJR Decoder

The BCJR convolutional decoder described in (5.3) can be summarized below. Let a block of information bit  $a_m$  with size  $K_a$  be transmitted. The  $rK_a$  coded bit vector are then generated

from the encoder. To have better performance, the encoder is initialized and terminated with idle states. Let  $K_o$  denote the additional bits for idle state initialization and termination. Thus, the overall block length is  $K = rK_a + K_o$ . Let the codeword be modulated by BPSK and sent to the receiver via a memoryless AWGN channel. The received signal vector  $\mathbf{y}_m$  at time  $m$  can be described as

$$\mathbf{y}_m = \mathbf{x}_m + \mathbf{n}_m \quad (5.13)$$

where  $\mathbf{x}_m = [x_{m,0}, x_{m,1}, \dots, x_{m,r-1}]^T$  is the  $m$ -th transmitted signal vector,  $x_{m,l} = (-1)^{b_{m,l}}$ ,  $l = \{0, 1, \dots, r-1\}$ ,  $r \in \mathbb{Z}$ ,  $r \geq 2$ , and  $\mathbf{n}_m$  is the noise vector with variance  $\sigma_n^2$  in each component. Similar to (5.4), we have the logarithmic branch metrics  $\ln \gamma_m(p, q)$  as

$$\ln \gamma_m(p, q) = \frac{-|\mathbf{y}_m - \mathbf{x}_{(p,q)}|^2}{2\sigma_n^2} + K_\gamma, \quad (5.14)$$

where  $K_\gamma = \ln(\Pr(a_m = a_{(p,q)}) \Pr(\mathbf{b}_m = \mathbf{b}_{(p,q)})) - \ln(2\pi\sigma_n^2)$  and  $\mathbf{x}_{(p,q)} = (-1)^{\mathbf{b}_{(p,q)}}$ . Note that since redundant bits are inserted to protect information bits, bits in the codeword are usually not equally probable. However, for a BCJR decoder, it is common to have the block size on the order of hundreds. Evaluating the *a priori* information becomes prohibitive. Thus, the probability  $\Pr(\mathbf{b}_m = \mathbf{b}_{(p,q)})$  is usually assumed to be equal for all bit patterns. Calculation of the forward and the backward state probabilities are both initialized to the idle state, i.e.,  $\ln \alpha_0 = [0, -\infty, \dots, -\infty]$  and  $\ln \beta_{K+\mu} = [0, -\infty, \dots, -\infty]^T$ , respectively. Except for a different signal constellation set, the procedure for finding APPs and hard decisions are the same as the BCJR equalizer.

#### § 5.2.4 The BCJR Turbo Equalizer

With a series of observations (received signal) and the *a priori* information, the generic SISO processing unit shown in Fig. 5.6 calculates and then generates a series of APPs and soft decisions. The BCJR is an optimal implementation for the SISO processor. The BCJR equalizer and decoder described are two examples of the standalone SISO application. From (5.4) and (5.14), we can see that the *a priori* information play a crucial role in the SISO processing. The

more reliable the *a priori* information, the better the performance. A BCJR turbo equalizer shown in Fig. 5.7 is constructed as a serial concatenation of a BCJR equalizer and a BCJR decoder, where the output APP of each one is fed back to the other as the *a priori* information. The idea is to enhance the *a priori* information through the iteration. However, the feedback loop should be carefully connected to avoid a *strongly* positive feedback, which will lead to fast convergence to a local solution. That is the so-called turbo principle.

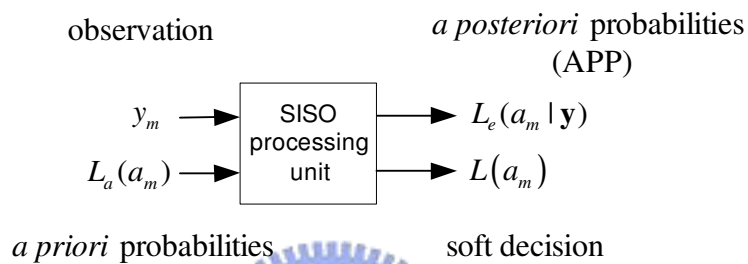


Figure 5.6: A generic SISO processing unit.

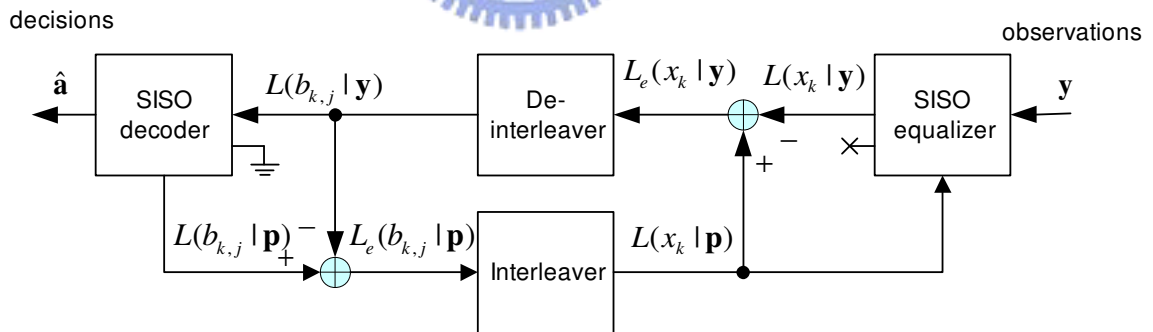


Figure 5.7: The block diagram of the turbo equalizer.

To satisfy the turbo principle, the *a priori* information is subtracted from the APP to yield the extrinsic information feeding to another SISO processing unit. The principle can be mathematically expressed as follows. For a block of transmitted binary sequence  $\mathbf{a}$  and the received

observations  $\mathbf{y}$ , the APP  $L(a_m|\mathbf{y})$  for a transmit bit  $a_m$  at time  $m$  can be decomposed as

$$\begin{aligned}
L(a_m|\mathbf{y}) &= \ln \frac{\sum_{\forall \mathbf{a}: a_m=0} p(\mathbf{y}|\mathbf{a}) \prod_{i=1}^K \Pr(a_i)}{\sum_{\forall \mathbf{a}: a_m=1} p(\mathbf{y}|\mathbf{a}) \prod_{i=1}^K \Pr(a_i)} \\
&= \ln \frac{\sum_{\forall \mathbf{a}: a_m=0} p(\mathbf{y}|\mathbf{a}) \prod_{i=1: i \neq m}^K \Pr(a_i)}{\sum_{\forall \mathbf{a}: a_m=1} p(\mathbf{y}|\mathbf{a}) \prod_{i=1: i \neq m}^K \Pr(a_i)} + \underbrace{\ln \frac{\Pr(a_m=0)}{\Pr(a_m=1)}}_{a \text{ priori}} \\
&= \underbrace{L_e(a_m|\mathbf{y})}_{\text{extrinsic}} + L_a(a_m), \tag{5.15}
\end{aligned}$$

where  $K$  is the block length in bits,  $L_e(a_m|\mathbf{y})$  is the extrinsic information, and  $L_a(a_m)$  is the *a priori* information. In other words, we have

$$L_e(a_m|\mathbf{y}) = L(a_m|\mathbf{y}) - L_a(a_m). \tag{5.16}$$

Note that the operation of the *a priori* information subtraction does not show in Fig. 5.1. The output APP ( $L_e(c_{k,j})$  and  $L_e(b_{m,l})$ ) in Fig. 5.1 (subscripts and conditional observations are ignored) is the extrinsic information.

For the SISO equalizer, the soft inputs are the observation  $y_k$  and the *a priori* probability  $\Pr(x_k)$  in (5.4). Thus, the configuration is the same as a standalone BCJR equalizer. Define  $u_{k,j} = (-1)^{c_{k,j}}$ . From (5.11), we have

$$\begin{aligned}
\Pr(c_{k,j}) &= \frac{e^{u_{k,j}L(c_{k,j})}}{1 + e^{u_{k,j}L(c_{k,j})}} \\
&= \left( \frac{1}{e^{L(c_{k,j})/2} + e^{-L(c_{k,j})/2}} \right) \cdot e^{u_{k,j}L(c_{k,j})/2} \\
&= K_u \cdot e^{u_{k,j}L(c_{k,j})/2}, \tag{5.17}
\end{aligned}$$

where  $K_u$  are equal for all transitions and can be ignored. Assume that bits  $c_{k,j}$ 's within a code-word are independent. The *a priori* probability  $\Pr(x_k)$ , which corresponds the soft decision,

can be obtained from the LLR  $L(c_{k,j})$  as follows:

$$\begin{aligned} \Pr(x_k = \chi_i) &= \prod_{j=0}^{\rho-1} \Pr(c_{k,j} = m_{i,j}) \\ &= \prod_{j=0}^{\rho-1} K_u \exp(u_{i,j} L(c_{k,j})/2), \end{aligned} \quad (5.18)$$

In summary, the extrinsic information of the SISO equalizer is calculated with (5.16) and is connected to the SISO decoder. The soft decision, however, is left unconnected.

For a SISO decoder used in a turbo equalizer, there are no observations. It only has the extrinsic information from the SISO equalizer and this serves as the *a priori* information for the SISO decoder. It is possible to consider the extrinsic information as the observation for the SISO decoder. However, it was reported in [62] that the performance will become poorer. Due to the lack of the observation, the configuration of a SISO decoder used in a turbo equalizer is different from a standalone BCJR decoder. Thus, some modifications are necessary in branch metrics calculation. The extrinsic information  $L(b_{m,l}|\mathbf{y})$  is first converted into its equivalent probability  $\Pr(b_{m,l}|\mathbf{y})$  with (5.18). The branch metric in (5.14) is then degenerated as

$$\begin{aligned} \gamma_m(p, q) &= \Pr(a_m = a_{(p,q)}) \Pr(\mathbf{b}_m = \mathbf{b}_{(p,q)}) \\ &= \Pr(a_m = a_{(p,q)}) \prod_{l=1}^r \Pr(b_{m,l} = b_{(p,q),l}). \end{aligned} \quad (5.19)$$

Here,  $b_{m,l}$ 's are assumed to be independent after deinterleaving,  $a_m$  an i.i.d. binary bit stream, and  $\Pr(a_m = a_{(p,q)})$  equally probable (could be ignored). The SISO decoder yields its extrinsic information in each iteration and output hard decisions at the last iteration. An interleaver is applied to the extrinsic information and the output is feedback to the SISO equalizer. The interleaver can reduce possible signal correlation and further reduce the positive feedback effect in the loop.

### § 5.2.5 Filter-based MMSE SISO Equalizer

It is well known that the complexity of conventional filter-based equalizers such as linear equalizer, decision feedback equalizer (DFE) is much lower than the trellis-based equalizer. Since no soft inputs and outputs are involved in those conventional equalizers, they can not be directly used in the turbo equalizer. In [44], the SISO entries are introduced and a filter-based SISO equalizer with minimized mean square error (MMSE) criterion was constructed. The overall system structure is shown in Fig. 5.8. The linear equalizer considered here is an interference cancellation (IC) linear equalizer [63–65], which is a generalization of the conventional linear equalizer. It can cancel not only casual (post-cursor) but also non-casual (pre-cursor) ISI. The non-casual ISI is cancelled with the soft decision feedback. The SISO linear equalizer consists of a soft-input MMSE equalizer, a soft-input converter, and a soft-output converter. The MMSE equalizer estimates the transmitted symbols from the received symbols by minimizing the mean square error (MSE) cost function  $E(|x_k - \hat{x}_k|^2)$ . The soft-input converter converts the *a priori* information (with the LLR format) into the soft-input ( $\bar{x}_k = E\{x_k\}$ ) and its associated reliability  $v_k = \text{cov}(x_k, x_k)$  for the equalizer. The reliability is used in the computation of the optimal filter coefficients. If the *a priori* information is more reliable, the soft decisions from the decoder will have more weights and vice versa. Note that a perfect ISI cancellation needs a perfect knowledge about detected data and the channel response; either error will result in residual ISI. Since the soft-input for the SISO decoder is the *a priori* information with the LLR format, the equalized  $\hat{x}_k$  cannot be applied to the SISO decoder directly. A soft-output converter is then necessary to perform the conversion. Detailed functions will be discussed in the later subsections. Now, we will discuss the mathematical details about the optimal filter-based SISO (OSL) equalizer.



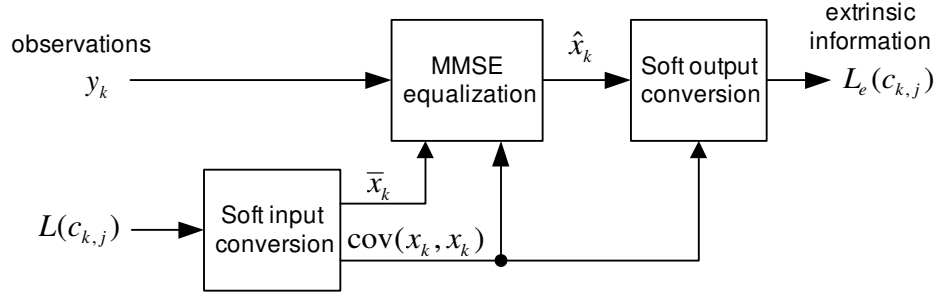


Figure 5.8: A filter-based MMSE SISO linear equalizer.

### A) OSL Equalizer

In this section, the OSL equalizer proposed in [45] is derived. Consider the structure shown in Fig. 5.8. Given a sequence of symbols  $y_k$ , we can express the linear estimate  $\hat{x}_k$  of the transmitted symbol  $x_k$  with the *a priori* information as [44]

$$\hat{x}_k = \mathbf{w}_k^T \mathbf{y}_k + d_k \quad (5.20)$$

where

$$\mathbf{y}_k = [y_{k+N_1}, y_{k+N_1-1}, \dots, y_{k-N_2}]^T, \quad (5.21)$$

$$\mathbf{w}_k = [w_{k,-N_1}, w_{k,-N_1+1}, \dots, w_{k,N_2}]^T, \quad (5.22)$$

$\mathbf{w}_k$  is the time-varying optimal filter tap weight vector and  $d_k$  is an biased term and  $N = N_1 + N_2 + 1$  is the length of  $\mathbf{w}_k$ . Here,  $N_1$  and  $N_2$  denote the length of the filter's non-causal and casual parts, respectively. The optimal solution using the MMSE criterion can be solved with

$$(\mathbf{w}_k, d_k) = \arg \min_{\mathbf{w}_k \in \mathbb{R}^N, d_k \in \mathbb{R}} E (|x_k - \hat{x}_k|^2). \quad (5.23)$$

The optimization problem can be easily solved by setting the partial derivatives of  $E (|x_k - \hat{x}_k|^2)$  with respect to  $\mathbf{w}_k$  and  $d_k$  to zeros.

$$\frac{\partial E (|x_k - \hat{x}_k|^2)}{\partial \mathbf{w}_k} = -2E [(x_k - \mathbf{w}_k^T \mathbf{y}_k - d_k) \mathbf{y}_k^T] = \mathbf{0}_N, \quad (5.24)$$

$$\frac{\partial E (|x_k - \hat{x}_k|^2)}{\partial d_k} = -2E [(x_k - \mathbf{w}_k^T \mathbf{y}_k - d_k)] = 0. \quad (5.25)$$

Or equivalently,

$$E (x_k \mathbf{y}_k) - \mathbf{w}_k^T E (\mathbf{y}_k \mathbf{y}_k^T) - d_k E (\mathbf{y}_k^T) = \mathbf{0}_N, \quad (5.26)$$

$$E (x_k) - \mathbf{w}_k^T E (\mathbf{y}_k) - d_k = 0. \quad (5.27)$$

From (5.26),

$$d_k = E (x_k) - \mathbf{w}_k^T E (\mathbf{y}_k). \quad (5.28)$$

Substituting (5.28) into (5.26), we have

$$\mathbf{w}_k = (E (\mathbf{y}_k \mathbf{y}_k^T) - E (\mathbf{y}_k) E (\mathbf{y}_k^T))^{-1} (E (x_k \mathbf{y}_k) - E (x_k) E (\mathbf{y}_k)), \quad (5.29)$$

The optimal MMSE solutions are then

$$\mathbf{w}_k = \text{cov} (\mathbf{y}_k, \mathbf{y}_k)^{-1} \text{cov} (\mathbf{y}_k, x_k), \quad (5.30)$$

$$d_k = E (x_k) - \mathbf{w}_k^T E (\mathbf{y}_k). \quad (5.31)$$

From (5.20), (5.30), and (5.32), we have the filter output as

$$\hat{x}_k = E(x_k) + \text{cov} (x_k, \mathbf{y}_k) \text{cov} (\mathbf{y}_k, \mathbf{y}_k)^{-1} (\mathbf{y}_k - E(\mathbf{y}_k)) \quad (5.32)$$

Define the soft-input  $\bar{x}_k$  and associated reliability  $v_k$  as

$$\bar{x}_k \triangleq E (x_k) = \sum_{\chi_i \in X} \chi_i \Pr(x_k = \chi_i), \quad (5.33)$$

$$v_k \triangleq \text{cov} (x_k, x_k) = \left( \sum_{\chi_i \in X} |\chi_i|^2 \Pr(x_k = \chi_i) \right) - |\bar{x}_k|^2. \quad (5.34)$$

Similar to (5.18), we assume that bits  $c_{k,j}$  within a codeword are independent. Then, the *a priori* probability can be calculated as

$$\begin{aligned} \Pr (x_k = \chi_i) &= \prod_{j=0}^{\rho-1} \Pr (c_{k,j} = m_{i,j}) \\ &= \prod_{j=0}^{\rho-1} K_u \exp (u_{i,j} L (c_{k,j})/2), \end{aligned} \quad (5.35)$$

where  $u_{i,j} \triangleq (-1)^{m_{i,j}}$ ,  $K_u$  and is a scaling factor.

Let  $\mathbf{H}$  be the time-invariant channel matrix as

$$\mathbf{H} = \begin{bmatrix} h_0 & h_1 & \cdots & h_\mu & 0 & \cdots & 0 \\ 0 & h_0 & h_1 & \cdots & h_\mu & 0 & \cdots \\ \vdots & & & & & & \vdots \\ 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_\mu \end{bmatrix}_{N \times (N+\mu)}. \quad (5.36)$$

We can then express the observations with a vector form.

$$\mathbf{y}_k = \mathbf{H}\mathbf{x}_k + \mathbf{n}_k, \quad (5.37)$$

where  $\mathbf{x}_k = [x_{k+N_1}, x_{k+N_1-1}, \cdots, x_{k-N_2-\mu}]^T$  is the transmit signal vector, and  $\mathbf{n}_k = [n_{k+N_1}, n_{k+N_1-1}, \cdots, n_{k-N_2}]^T$  is the noise vector. The noise  $n_k$  is AWGN with mean zero and variance  $\sigma_n^2$ , and is independent with  $x_k$ . If we assume that  $x_k$  is i.i.d., the covariance matrix  $\text{cov}(\mathbf{x}_k, \mathbf{x}_k)$  becomes diagonal matrix. We have

$$\text{cov}(\mathbf{y}_k, \mathbf{y}_k) = \sigma_n^2 \mathbf{I}_N + \mathbf{H}\mathbf{V}_k\mathbf{H}^T, \quad (5.38)$$

$$\text{cov}(\mathbf{y}_k, x_k) = v_k \mathbf{s}^T, \quad (5.39)$$

$$\bar{\mathbf{y}}_k \triangleq E(\mathbf{y}_k) = \mathbf{H}E(\mathbf{x}_k) = \mathbf{H}\bar{\mathbf{x}}_k, \quad (5.40)$$

where

$$\mathbf{V}_k \triangleq \text{cov}(\mathbf{x}_k, \mathbf{x}_k) = \text{diag}[v_{k+N_1}, v_{k+N_1-1}, \cdots, v_{k-N_2-\mu}], \quad (5.41)$$

$$\mathbf{s} \triangleq \mathbf{H} \begin{bmatrix} \mathbf{0}_{1 \times (N_2+\mu)} & 1 & \mathbf{0}_{1 \times N_1} \end{bmatrix}^T, \quad (5.42)$$

$$\bar{\mathbf{x}}_k = [\bar{x}_{k+N_1}, \bar{x}_{k+N_1-1}, \cdots, \bar{x}_{k-N_2-\mu}]^T. \quad (5.43)$$

The estimate  $\hat{x}_k$  without applying the turbo principle becomes

$$\hat{x}_k = \bar{x}_k + v_k \mathbf{s}^T (\sigma_n^2 \mathbf{I}_N + \mathbf{H}\mathbf{V}_k\mathbf{H}^T)^{-1} (\mathbf{y}_k - \mathbf{H}\bar{\mathbf{x}}_k). \quad (5.44)$$

With the turbo principle,  $\hat{x}_k$  must be independent with the *a priori* information  $L(c_{k,j})$ . We then set  $L(c_{k,j})$  to zero, or equivalently  $\bar{x}_k = 0$  and  $v_k = 1$  in the computation of  $\hat{x}_k$ . Thus, (5.43) and (5.41) can be rewritten accordingly. Let

$$\begin{aligned}\bar{\mathbf{x}}'_k &= \bar{\mathbf{x}}_k|_{\bar{x}_k=0} \\ &= [\bar{x}_{k+N_1}, \dots, \bar{x}_{k+1}, 0, \bar{x}_{k-1}, \dots, \bar{x}_{k-N_2-\mu}]^T \\ &= \bar{\mathbf{x}}_k - [0, \dots, 0, \bar{x}_k, 0, \dots, 0]^T,\end{aligned}\quad (5.45)$$

$$\begin{aligned}\mathbf{V}'_k &= \mathbf{V}_k|_{v_k=1} \\ &= \text{diag}[v_{k+N_1}, \dots, v_{k+1}, 1, v_{k-1}, \dots, v_{k-N_2-\mu}] \\ &= \mathbf{V}_k - \text{diag}[0, \dots, 0, (1-v_k), 0, \dots, 0].\end{aligned}\quad (5.46)$$

Rewriting (5.38) and (5.39), we have

$$\text{cov}(\mathbf{y}'_k, \mathbf{y}'_k) = (\sigma_n^2 \mathbf{I}_N + \mathbf{H}\mathbf{V}'_k\mathbf{H}^T + (1-v_k)\mathbf{s}\mathbf{s}^T), \quad (5.47)$$

$$\text{cov}(\mathbf{y}'_k, x_k) = \mathbf{s}^T, \quad (5.48)$$

$$\bar{\mathbf{y}}'_k \triangleq E(\mathbf{y}_k) = \mathbf{H}\bar{\mathbf{x}}'_k. \quad (5.49)$$

Thus, the estimate  $\hat{x}'_k$  with the turbo principle applied becomes

$$\begin{aligned}\hat{x}'_k &= \hat{x}_k|_{\bar{x}_k=0, v_k=1} \\ &= \mathbf{s}^T(\sigma_n^2 \mathbf{I}_N + \mathbf{H}\mathbf{V}'_k\mathbf{H}^T)^{-1}(\mathbf{y}_k - \mathbf{H}\bar{\mathbf{x}}'_k) \\ &= \mathbf{s}^T(\sigma_n^2 \mathbf{I}_N + \mathbf{H}\mathbf{V}_k\mathbf{H}^T + (1-v_k)\mathbf{s}\mathbf{s}^T)^{-1}(\mathbf{y}_k - \mathbf{H}\bar{\mathbf{x}}_k + (\bar{x}_k - 0)\mathbf{s}).\end{aligned}\quad (5.50)$$

Hereafter, we ignore the superscript prime and express the OSL equalizer as

$$\hat{x}_k = \mathbf{f}_k^T(\mathbf{y}_k - \mathbf{H}\bar{\mathbf{x}}_k + \bar{x}_k\mathbf{s}). \quad (5.51)$$

where

$$\mathbf{f}_k = (\sigma_n^2 \mathbf{I}_N + \mathbf{H}\mathbf{V}_k\mathbf{H}^T + (1-v_k)\mathbf{s}\mathbf{s}^T)^{-1}\mathbf{s}. \quad (5.52)$$

Note that the the optimal filter is time-varying.

To obtain the extrinsic information  $L_e(c_{k,j})$ , the equalizer's output  $\hat{x}_k$  is assumed to be Gaussian distributed with  $x_k = \chi_i$  as its mean. The conditional PDF of  $\hat{x}_k$  ( $p(\hat{x}_k|x_k = \chi_i) = p(\hat{x}_k|\mathbf{c}_k = \mathbf{m}_i)$ ) is also approximated by a Gaussian distribution with the mean  $\mu_{k,i} \triangleq E(\hat{x}_k|x_k = \chi_i)$  and variance  $\sigma_{k,i}^2 \triangleq \text{cov}(\hat{x}_k, \hat{x}_k|x_k = \chi_i)$ ,

$$p(\hat{x}_k|x_k = \chi_i) \approx \frac{1}{\sqrt{2\pi}\sigma_{k,i}} \exp\left\{-\frac{|\hat{x}_k - \mu_{k,i}|^2}{2\sigma_{k,i}^2}\right\}. \quad (5.53)$$

This assumption [42,45] simplifies the computation of the output extrinsic information  $L_e(c_{k,j})$  a lot. The mean and variance of  $\hat{x}_k$  are given by

$$\begin{aligned} \mu_{k,i} &= E(\hat{x}_k|x_k = \chi_i) \\ &= \mathbf{f}_k^T (E(\mathbf{y}_k|x_k = \chi_i) - \mathbf{H}\bar{\mathbf{x}}_k + \bar{x}_k\mathbf{s}) \\ &= \chi_i \cdot \mathbf{f}_k^T \mathbf{s}, \end{aligned} \quad (5.54)$$

$$\begin{aligned} \sigma_{k,i}^2 &= \text{cov}(\hat{x}_k, \hat{x}_k|x_k = \chi_i) \\ &= \mathbf{f}_k^T \text{cov}(\mathbf{y}_k, \mathbf{y}_k|x_k = \chi_i) \mathbf{f}_k \\ &= \mathbf{f}_k^T (\sigma_n^2 \mathbf{I}_N + \mathbf{H}\mathbf{V}_k\mathbf{H}^T - v_k\mathbf{s}\mathbf{s}^T) \mathbf{f}_k \\ &= \mathbf{f}_k^T \mathbf{s} (1 - \mathbf{f}_k^T \mathbf{s}). \end{aligned} \quad (5.55)$$

The last equality in (5.55) is obtained with (5.52). Then, the extrinsic information can be calculated as

$$\begin{aligned} L_e(c_{k,j}) &= \ln \frac{\sum_{\forall \mathbf{m}_i: m_{i,j}=0} \left[ p(\hat{x}_k|\mathbf{c}_k = \mathbf{m}_i) \prod_{j'=1:j' \neq j}^{\rho} \Pr(c_{k,j'} = m_{i,j'}) \right]}{\sum_{\forall \mathbf{m}_i: m_{i,j}=1} \left[ p(\hat{x}_k|\mathbf{c}_k = \mathbf{m}_i) \prod_{j'=1:j' \neq j}^{\rho} \Pr(c_{k,j'} = m_{i,j'}) \right]} \\ &= \ln \frac{\sum_{\forall \mathbf{m}_i: m_{i,j}=0} \exp \left[ -\frac{\rho_{k,i}}{2} + \sum_{j'=1:j' \neq j}^{\rho} \frac{u_{i,j'} L(c_{k,j'})}{2} \right]}{\sum_{\forall \mathbf{m}_i: m_{i,j}=1} \exp \left[ -\frac{\rho_{k,i}}{2} + \sum_{j'=1:j' \neq j}^{\rho} \frac{u_{i,j'} L(c_{k,j'})}{2} \right]}, \end{aligned} \quad (5.56)$$

where  $\rho_{k,i} \triangleq \frac{(\hat{x}_k - \mu_{k,i})^2}{\sigma_{k,i}^2}$ . From (5.51), (5.54), and (5.55), we have

$$\rho_{k,i} = \frac{|\hat{x}_k - \chi_i \cdot \mathbf{f}_k^T \mathbf{s}|^2}{\mathbf{f}_k^T \mathbf{s} (1 - \mathbf{f}_k^T \mathbf{s})}. \quad (5.57)$$

Table 5.1 summarizes the formulae for calculating the soft-input, reliability, and extrinsic information with M-PAM modulation. For detailed derivations, please refer to Appendix D. Note that the formulae for soft-input and reliability are valid for all kind of filter-based SISO equalizer described in this dissertation, but the extrinsic information is algorithm dependent. The last row in Table 5.1 is also valid for other modulation schemes such as M-PSK and M-QAM.

Table 5.1: Soft-input and soft-output conversion formulae for M-PAM

| Modulation         | Soft-input                                | Reliability                          | Extrinsic information  |
|--------------------|---|--------------------------------------|--|
| 2-PAM (BPSK)       | $\bar{x}_k = \tanh(L(x_k)/2)$             | $v_k = 1 -  \bar{x}_k ^2$            | $L_e(c_{k,1}) = 2\hat{x}_k / (1 - \mathbf{f}_k^T \mathbf{s})$  |
| 4-PAM <sup>a</sup> | $\bar{x}_k = \frac{2L_1 + L_2}{\sqrt{5}}$ | $v_k = 1 - \frac{4L_1^2 + L_2^2}{5}$ | $L_e(c_{k,1}) =$<br>$\ln \frac{e^{\frac{(-\rho_0 + l_2)}{2}} + e^{\frac{(-\rho_1 - l_2)}{2}}}{e^{\frac{(-\rho_2 + l_2)}{2}} + e^{\frac{(-\rho_3 - l_2)}{2}}}$<br>$L_e(c_{k,2}) =$<br>$\ln \frac{e^{\frac{(-\rho_0 + l_1)}{2}} + e^{\frac{(-\rho_2 - l_1)}{2}}}{e^{\frac{(-\rho_1 + l_1)}{2}} + e^{\frac{(-\rho_3 - l_1)}{2}}}$ |
| M-PAM              | Eq. (5.33)                                | Eq. (5.34)                           | Eq. (5.56)   |

<sup>a</sup>For 4-PAM, we define  $L_1 \triangleq \tanh(L(c_{i,1})/2)$ ,  $L_2 \triangleq \tanh(L(c_{i,2})/2)$ ;  $\rho_i \triangleq \rho_{k,i}$ , and  $l_j = L(c_{k,j})$ .

For BPSK modulation, the operations of the OSL equalizer can be summarized as follows.

1. Soft input conversion: convert the *a priori* information  $L(x_k)$  into soft inputs  $\bar{x}_k$  and reliability  $v_k$ .

$$\bar{x}_k = \tanh(L(x_k)/2), \quad v_k = 1 - |\bar{x}_k|^2. \quad (5.58)$$

2. Optimal filter coefficients calculation: (assume that the noise variance and channel response are known)

$$\mathbf{f}_k = (\sigma_n^2 \mathbf{I}_N + \mathbf{H} \mathbf{V}_k \mathbf{H}^T + (1 - v_k \mathbf{s} \mathbf{s}^T))^{-1} \mathbf{s}. \quad (5.59)$$

3. Filtering: (for simplicity, some constant factors are cancelled by soft output conversion)

$$\hat{x}_k = \mathbf{f}_k^T (\mathbf{y}_k - \mathbf{H}\bar{x}_k + \bar{x}_k \mathbf{s}). \quad (5.60)$$

4. Soft output (extrinsic information) conversion:

$$L_e(c_{k,j}) = 2\hat{x}_k / (1 - \mathbf{f}_k^T \mathbf{s}). \quad (5.61)$$

The power of the filter-based SISO linear equalizer is manifested in (5.59). With the knowledge of the *a priori* information (the reliability matrix  $\mathbf{V}_k$ ), the equalizer changes its filtering strategy (optimal filter response) adaptively. For the case of no *a priori* information (i.e.,  $L(x_k) = 0$ ,  $\bar{x}_k = 0$ ,  $v_k = 1$ , which is the case at the first iteration), it performs like a classical MMSE linear equalizer. For the case of perfect *a priori* information (i.e.,  $|L(x_k)| \rightarrow \infty$ ,  $\bar{x}_k = x_k$ ,  $v_k = 0$ ), it becomes to a perfect MMSE IC linear equalizer [63]. In general, we have the *a priori* information with certain reliability (i.e.,  $0 < |L(x_k)| < \infty \Rightarrow 1 < v_k < 0$ ). The equalizer performs optimal filtering for a given reliability (like the morphing in computer graphics) between these two extreme cases. The similar behavior is also observed in the output extrinsic information, which is also a function of reliability. The iterative equalization will begin with no *a priori* at the first iteration, then with better and better *a priori* (toward the perfect *a priori*) at later iterations. The equalizer's performance is then improved iteratively.

Note that the optimal filter is time-varying and updated on the symbol-by-symbol basis. The major complexity burden comes from the matrix inversion operation in (5.59), resulting in an  $O(N^3)$  complexity per symbol. Though a time-recursive updated algorithm for the optimal time-varying filter was proposed in [44, 45] in which the order was reduced to  $O(N^2)$ , the complexity is still high. To further reduce the complexity, Tüchler *et al.* [45] proposed to approximate the optimal time-variant filter with a suboptimal block time-invariant (time-invariant within a processing block). With this approach, an  $O(N)$  complexity was achieved. We call this a low-complexity SISO linear (LSL) equalizer. The algorithm is summarized as follows.

**B) LSL Equalizer**

The reliability  $v_k$  in (5.59) is replaced with its block-averaged value, i.e.,

$$\bar{v} \triangleq \frac{1}{K} \sum_{k=1}^K v_k. \quad (5.62)$$

Then, the optimal filter  $\mathbf{f}_k$  becomes block time-invariant. We then have the optimal filter  $\mathbf{f}$  as

$$\begin{aligned} \mathbf{f} &\triangleq \left( \sigma_n^2 \mathbf{I}_N + \mathbf{H} \bar{\mathbf{V}} \mathbf{H}^T + (1 - \bar{v} \mathbf{S} \mathbf{S}^T) \right)^{-1} \cdot \mathbf{s} \\ &= \left( \sigma_n^2 \mathbf{I}_N + \bar{v} \mathbf{H} \mathbf{H}^T + (1 - \bar{v} \mathbf{S} \mathbf{S}^T) \right)^{-1} \cdot \mathbf{s}, \end{aligned} \quad (5.63)$$

where

$$\begin{aligned} \bar{\mathbf{V}} &\triangleq \frac{1}{K} \sum_{k=1}^K \mathbf{V}_k \\ &= \bar{v} \mathbf{I}_{N+M-1}. \end{aligned} \quad (5.64)$$

Since the optimal filter is changed, the output statistics of  $\hat{x}_k$  are changed accordingly.

$$\begin{aligned} \mu_{k,i} &= E(\hat{x}_k | x_k = \chi_i) \\ &= \mathbf{f}^T (E(\mathbf{y}_k | x_k = \chi_i) - \mathbf{H} \bar{\mathbf{x}}_k + \bar{x}_k \mathbf{s}) \\ &= \chi_i \cdot \mathbf{f}^T \mathbf{s}, \end{aligned} \quad (5.65)$$

$$\begin{aligned} \sigma_{k,i}^2 &= \text{cov}(\hat{x}_k, \hat{x}_k | x_k = \chi_i) \\ &= \mathbf{f}^T \text{cov}(\mathbf{y}'_k, \mathbf{y}'_k | x_k = \chi_i) \mathbf{f} \\ &= \mathbf{f}^T (\sigma_n^2 \mathbf{I}_N + \mathbf{H} \mathbf{V}_k \mathbf{H}^T - v_k \mathbf{S} \mathbf{S}^T) \mathbf{f}. \end{aligned} \quad (5.66)$$

Note that the equality in (5.52) is not satisfied when the block time-invariant filter  $\mathbf{f}$  is applied. Thus, (5.66) cannot be simplified as (5.55). The output extrinsic information  $L_e(c_{k,j})$  can be still obtained with (5.56), but  $\rho_{k,i}$  is changed to

$$\rho_{k,i} = \frac{|\hat{x}_k - \mu_{k,i}|^2}{\mathbf{f}^T (\sigma_n^2 \mathbf{I}_N + \mathbf{H} \mathbf{V}_k \mathbf{H}^T - v_k \mathbf{S} \mathbf{S}^T) \mathbf{f}}. \quad (5.67)$$



Since the denominator of (5.67) is time-varying, the complexity burden is shifted to the computation of  $\sigma_{k,i}^2$ . It is even larger than the computation of time-varying optimal filter coefficients in (5.59). Fortunately, the averaged reliability can be applied to solve this problem without obvious performance degradation [45]. Now,  $\rho_{k,i}$  is approximated by

$$\begin{aligned}\rho_{k,i} &\approx \frac{|\hat{x}_k - \mu_{k,i}|^2}{2\mathbf{f}^T(\sigma_n^2\mathbf{I}_N + \mathbf{H}\bar{\mathbf{V}}\mathbf{H}^T - \bar{v}\mathbf{s}\mathbf{s}^T)\mathbf{f}} \\ &= \frac{|\hat{x}_k - \mu_{k,i}|^2}{\mu_{k,i}(1 - \mu_{k,i})}.\end{aligned}\quad (5.68)$$

The last equality in (5.68) is obtained with (5.63). That means the variance can be approximated by

$$\sigma_{k,i}^2 \approx \mu_{k,i}(1 - \mu_{k,i}). \quad (5.69)$$

The extrinsic information for M-PAM is given in Table 5.1.

For BPSK modulation, the LSL equalizer can be summarized as follows:

1. Soft input conversion:

$$\bar{x}_k = \tanh(L(x_k)/2), \quad v_k = 1 - |\bar{x}_k|^2, \quad \bar{v} \triangleq \frac{1}{K} \sum_{k=1}^K v_k. \quad (5.70)$$

2. Optimal filter coefficients calculation:

$$\mathbf{f} = (\sigma_n^2\mathbf{I}_N + \bar{v}\mathbf{H}\mathbf{H}^T + (1 - \bar{v})\mathbf{s}\mathbf{s}^T)^{-1} \cdot \mathbf{s}. \quad (5.71)$$

3. Filtering:

$$\hat{x}_k = \mathbf{f}^T(\mathbf{y}_k - \mathbf{H}\bar{\mathbf{x}}_k + \bar{x}_k\mathbf{s}). \quad (5.72)$$

4. Soft output conversion:

$$L_e(c_{k,1}) = 2\hat{x}_k/(1 - \mathbf{f}^T\mathbf{s}). \quad (5.73)$$

Interestingly, the LSL equalizer is just a special case of OSL equalizer when reliability is constant within a block, i.e.,  $v_k = \bar{v}$  (or  $|L(x_k)| = \text{constant}$ ). Note that the algorithm still inherits the capability to change its filtering strategy adaptively. For the LSL equalizer, the

filter is block time-invariant. For some special assumptions, the LSL equalizer can be further simplified [46]. Either in the OSL or LSL equalizer, there are two mechanisms to exploit the knowledge of the *a priori* information. One is in the filtering stage, and the other is in the soft output conversion stage. If only the latter mechanism is preserved, a time-invariant equalizer can be obtained. In other words, the filter will be the same for all iterations and also for all blocks. There are two extreme cases yielding this result; the *a priori* is assumed to be totally unknown and perfectly known all the time. The equalizer is then reduced to the classical linear equalizer and the classical matched filter for the former and later cases, respectively. We name the former LSL equalizer as the LSLN equalizer, and the later as the LSLP equalizer. We can even have a hybrid of the LSLN and LSLP equalizers, which is named as the LSLH equalizer. Since the filter is time-invariant, the complexity is lower than the LSL equalizer.

### C) LSLN, LSLP, and LSLH Equalizers

In this section, we will summarize operations of the LSLN, LSLP, and LSLH equalizers. Since the equalizer becomes time-invariant, the soft output conversion takes full responsibility for reliability adaptation.

**C.1) LSLN Equalizer** For no *a priori* information, i.e.  $\bar{x}_k = 0$ ,  $v_k = 1$  (or  $|L(x_k)| = 0$ ), the optimal filter degenerates to an optimal LE filter as follows:

$$\begin{aligned} \mathbf{f}_{NA} &\triangleq (\sigma_n^2 \mathbf{I}_N + \mathbf{H} \mathbf{V}_k \mathbf{H}^T + (1 - v_k \mathbf{s} \mathbf{s}^T))^{-1} \mathbf{s} \Big|_{\bar{x}_k=0, v_k=1} \\ &= (\sigma_n^2 \mathbf{I}_N + \mathbf{H} \mathbf{H}^T)^{-1} \cdot \mathbf{s}, \end{aligned} \quad (5.74)$$

The output statistics of  $\hat{x}_k$  become

$$\mu_{k,i} = \chi_i \cdot \mathbf{f}_{NA}^T \mathbf{s}, \quad (5.75)$$

$$\sigma_{k,i}^2 = \mathbf{f}_{NA}^T (\sigma_n^2 \mathbf{I}_N + \mathbf{H} \mathbf{V}_k \mathbf{H}^T - v_k \mathbf{s} \mathbf{s}^T) \mathbf{f}_{NA}. \quad (5.76)$$

Unfortunately, (5.76) requires matrix multiplication operations for every symbol and the complexity is high. To reduce the complexity, we can approximate  $\sigma_{k,i}^2$  with its time average

$\bar{\sigma}_i^2$ .

$$\begin{aligned}
\bar{\sigma}_i^2 &\triangleq \frac{1}{K} \sum_{k=1}^K \sigma_{k,i}^2 \\
&= \mathbf{f}_{NA}^T (\sigma_n^2 \mathbf{I}_N + \frac{1}{K} \sum_{k=1}^K (\mathbf{H}\mathbf{V}_k \mathbf{H}^T - v_k \mathbf{s}\mathbf{s}^T)) \mathbf{f}_{NA} \\
&\approx \mathbf{f}_{NA}^T (\sigma_n^2 \mathbf{I}_N + \bar{v} (\mathbf{H}\mathbf{H}^T - \mathbf{s}\mathbf{s}^T)) \mathbf{f}_{NA}.
\end{aligned} \tag{5.77}$$

Since the optimal filter is time-invariant, the extrinsic information should be calculated using the general formula in (5.56).

For BPSK modulation, the LSLN equalizer can be summarized as follows:

1. Soft input conversion:

$$\bar{x}_k = \tanh(L(x_k)/2), \quad v_k = 1 - |\bar{x}_k|^2, \quad \bar{v} \triangleq \frac{1}{K} \sum_{k=1}^K v_k. \tag{5.78}$$

2. Optimal filter coefficients calculation:

$$\mathbf{f}_{NA} = (\sigma_n^2 \mathbf{I}_N + \mathbf{H}\mathbf{H}^T)^{-1} \cdot \mathbf{s}. \tag{5.79}$$

3. Filtering:

$$\hat{x}_k = \mathbf{f}_{NA}^T (\mathbf{y}_k - \mathbf{H}\bar{\mathbf{x}}_k + \bar{x}_k \mathbf{s}). \tag{5.80}$$

4. Soft output conversion:

$$L_e(c_{k,1}) = 2\hat{x}_k \kappa_1 / (\kappa_2 + \kappa_3 \bar{v}). \tag{5.81}$$

where  $\kappa_1 = \mathbf{f}_{NA}^T \mathbf{s}$ ,  $\kappa_2 = \sigma_n^2 \mathbf{f}_{NA}^T \mathbf{f}_{NA}$ , and  $\kappa_3 = \mathbf{f}_{NA}^T (\mathbf{H}\mathbf{H}^T - \mathbf{s}\mathbf{s}^T) \mathbf{f}_{NA}$  are constants.

**C.2) LSLP Equalizer** For perfect *a priori* information, i.e.  $\bar{x}_k = x_k$ ,  $v_k = 0$  (or  $|L(x_k)| = \infty$ ), the optimal filter degenerates to a matched filter as

$$\begin{aligned}
\mathbf{f}_{MF} &\triangleq (\sigma_n^2 \mathbf{I}_N + \mathbf{H}\mathbf{V}_k \mathbf{H}^T + (1 - v_k \mathbf{s}\mathbf{s}^T))^{-1} \mathbf{s} \Big|_{\bar{x}_k=x_k, v_k=0} \\
&= (\sigma_n^2 \mathbf{I}_N + \mathbf{s}\mathbf{s}^T)^{-1} \cdot \mathbf{s} \\
&= 1/(\sigma_n^2 + E_h) \cdot \mathbf{s},
\end{aligned} \tag{5.82}$$

where  $E_h \triangleq \mathbf{s}^T \mathbf{s} = \mathbf{h}^T \mathbf{h}$  is the total energy of channel response. The last equality in (5.82) is obtained with the matrix inversion lemma [19]. The output statistics of  $\hat{x}_k$  are changed to

$$\mu_{k,i} = \chi_i \cdot \mathbf{f}_{MF}^T \mathbf{s} = \chi_i \cdot E_h / (\sigma_n^2 + E_h), \quad (5.83)$$

$$\sigma_{k,i}^2 = \mathbf{f}_{MF}^T (\sigma_n^2 \mathbf{I}_N + \mathbf{H} \mathbf{V}_k \mathbf{H}^T - v_k \mathbf{s} \mathbf{s}^T) \mathbf{f}_{MF}. \quad (5.84)$$

Similar to LSLN equalizer, the computational complexity of (5.84) is high. We can then approximate  $\sigma_{k,i}^2$  with its time average  $\bar{\sigma}_i^2$ . Then, we have

$$\begin{aligned} \bar{\sigma}_i^2 &\triangleq \mathbf{f}_{MF}^T (\sigma_n^2 \mathbf{I}_N + \frac{1}{K} \sum_{k=1}^K (\mathbf{H} \mathbf{V}_k \mathbf{H}^T - v_k \mathbf{s} \mathbf{s}^T)) \mathbf{f}_{MF} \\ &\approx \mathbf{f}_{MF}^T (\sigma_n^2 \mathbf{I}_N + \bar{v} (\mathbf{H} \mathbf{H}^T - \mathbf{s} \mathbf{s}^T)) \mathbf{f}_{MF} \\ &= (\sigma_n^2 E_h + \bar{v} (\mathbf{s}^T \mathbf{H} \mathbf{H}^T \mathbf{s} - E_h^2)) / (\sigma_n^2 + E_h)^2. \end{aligned} \quad (5.85)$$

Since the optimal filter is time-invariant, the extrinsic information should be calculated with the general formula in (5.56).

For BPSK modulation, the LSLP equalizer can be summarized as follows:

1. Soft input conversion:

$$\bar{x}_k = \tanh(L(x_k)/2), \quad v_k = 1 - |\bar{x}_k|^2, \quad \bar{v} \triangleq \frac{1}{L} \sum_{k=1}^L v_k. \quad (5.86)$$

2. Optimal filter coefficients calculation:

$$\mathbf{f}_{MF} = 1 / (\sigma_n^2 + E_h) \cdot \mathbf{s}. \quad (5.87)$$

3. Filtering:

$$\hat{x}_k = \mathbf{f}_{MF}^T (\mathbf{y}_k - \mathbf{H} \bar{\mathbf{x}}_k + \bar{x}_k \mathbf{s}). \quad (5.88)$$

4. Soft output conversion:

$$L_e(c_{k,j}) = 2\kappa_1 / (\kappa_2 + \kappa_3 \bar{v}) \cdot \hat{x}_k. \quad (5.89)$$

where  $\kappa_1 = E_h(\sigma_n^2 + E_h)$ ,  $\kappa_2 = \sigma_n^2 E_h$ , and  $\kappa_3 = \mathbf{s}^T \mathbf{H} \mathbf{H}^T \mathbf{s} - E_h$  are constants.

**C.3) LSLH Equalizer** Though the complexity of LSLN (or LSLP) equalizer is lower than LSL equalizer, the performance and convergence behavior is poorer [66]. This is because the LSLN equalizer and LSLP equalizers only consider simplified cases. For the LSLN equalizer [46], the convergence behavior (using the EXIT charts analysis [67,68]) works well but it cannot reach performance bound even using more iterations due to no *a priori* assumed. For the LSLP equalizer [46], it provides minor improvement at first few iterations then stop quickly due to perfect *a priori* assumed.

Combining the LSLN equalizer and the LSLP equalizer, we can obtain a the LSLH equalizer [46]. The operation of the LSLH equalizer is summarized as follows. For the first few iterations, we use the LSLN equalizer and then switch to the LSLP equalizer for remaining iterations. The hybrid scheme overcomes the drawbacks of the LSLN/LSLP equalizer and provides a low complexity alternative. The problem of the LSLH equalizer is the selection of the switching point, which is system dependent. Determination of the switching point must consider the EXIT behaviors of both the equalizer and the decoder. Since the LSLH equalizer are optimized only at first and last iterations, there is mismatch for the iterations in between. This results in a poor transition behavior and also affect the final performance.

In [46], a SISO DFE is also proposed to the filter-based turbo equalizer. The filter structure is a classical DFE but the optimal feedforward and feedbackward filters are modified to take into account the *a priori* information. The performance of a turbo MMSE DFE is highly channel dependent and it is not necessarily better than a turbo MMSE LE. Since its convergence behavior cannot be analyzed with the EXIT chart and there is no low-complexity alternative of SISO DFE proposed in [46], the turbo MMSE DFE will not be considered in this dissertation.

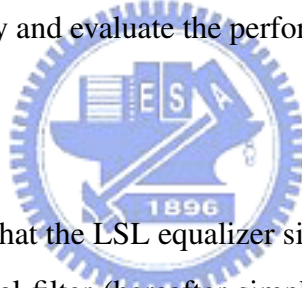
Although the complexity of the LSL equalizer is lower than the OSL equalizer, it is still far from acceptable for long wireline channels. In this chapter, we propose a new filter-based turbo equalizer to solve the problem. The proposed algorithm not only inherits all advantages of LSL equalizer but also with much lower computational complexity.

## § 5.3 The Proposed Fast Interpolated Turbo Equalizer

For wireline applications such as DSL, the channels often possess strong lowpass characteristics and this makes the channel response look like a smoothly decaying function. The channel length is long and usually on the order of hundreds of taps. It is apparent that high complexity trellis-based turbo equalizers can not be applied to such applications. Since the channel consists of the fixed twisted-pair copper wires, the channel is almost time-invariant. In this section, we exploit the time-invariant and lowpass channel characteristics to propose fast interpolated turbo equalizer. First, we formulate the fast singly interpolated SISO linear (FSISL) equalizer for time-invariant wireline channels. Combining the idea of channel response interpolation [15,36], we then further develop a fast doubly interpolated SISO linear (FDISL) equalizer. Finally, we analyze the computational complexity and evaluate the performance of proposed algorithms.

### § 5.3.1 FSISL Equalizer

From previous discussion, we know that the LSL equalizer simplifies the optimal time-varying filter to a block time-invariant optimal filter (hereafter simply named it as the optimal filter). We also find that the optimal filter is a function of the average reliability. The average reliability in a LSL turbo equalizer is then a function of both the iteration and block number (see (5.62)). For simplicity, we just use reliability to refer average reliability hereafter. Thus, the optimal filter of LSL equalizer is also a function of both the iteration and block number as shown in Fig. 5.9. In the figure, the optimal filter of the LSL equalizer is denoted as  $\mathbf{f}_i^n = \mathbf{f}_i^n(\bar{v}_i^n)$ , where the superscript indicates the block number and the subscript the iteration number. As an example, we show the reliability of the LSL turbo equalizer operated on the Proakis B channel [69] with 4-PAM in Fig. 5.10. The Proakis B channel is a medium-ISI channel with a 3-tap response as  $\mathbf{h} = [0.407, 0.815, 0.407]^T$ . Except for the zero-th iteration (which the reliability is always equals 1 because of no *a priori* information is given), it is clear that the reliability for a certain iteration is block-variant. For a given block, the variation process of the optimal filters of LSL equalizer



for different iterations (from no *a priori* to perfect *a priori*) is shown in Fig. 5.11. If we overlap the responses of optimal filters, the same variation process is also shown in Fig. 5.12. From those figures, we can find that the responses of the optimal filters have similar shapes. Also, the variation of a specific tap value is continuous (it will be proven theoretically later) with respect to the reliability. Thus, it is possible to approximate an optimal filter with two other given optimal filters. This motivates the development of our fast algorithms. Recalling (5.71),

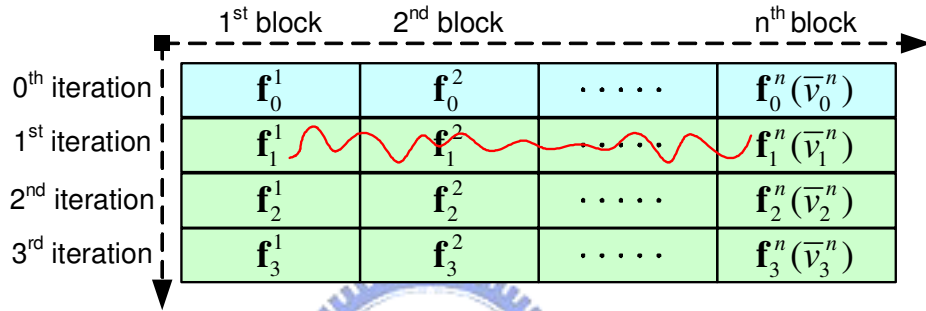


Figure 5.9: The optimal block time-invariant filter vs. the iteration and block number.

we have the optimal filter for a given reliability  $\bar{v}$  as

$$\begin{aligned}
 \mathbf{f} &= (\sigma_n^2 \mathbf{I}_N + \bar{v} \mathbf{H} \mathbf{H}^T + (1 - \bar{v}) \mathbf{s} \mathbf{s}^T)^{-1} \cdot \mathbf{s} \\
 &\triangleq (\mathbf{S}_1 + \mathbf{S}_2 \bar{v})^{-1} \cdot \mathbf{S}_3 \\
 &\triangleq \zeta(\bar{v}).
 \end{aligned} \tag{5.90}$$

where  $\mathbf{S}_1 \triangleq \sigma_n^2 \mathbf{I}_N + \mathbf{s} \mathbf{s}^T$ ,  $\mathbf{S}_2 \triangleq \mathbf{H} \mathbf{H}^T - \mathbf{s} \mathbf{s}^T$ , and  $\mathbf{S}_3 \triangleq \mathbf{s}$ . Thus, if the channel and the noise variance are given, the optimal filter becomes a function of reliability only. Here, we use  $\zeta(\cdot)$  to denote the function. Note that this is a generic result and it is independent of the iteration and block number.

Define the  $j$ -th tap weight of  $\mathbf{f}$  as  $f_j$ . From (5.90), we see that  $f_j$  is a scalar function of  $\bar{v}$ , i.e.,  $f_j(\bar{v})$ . From the well-known Weierstrass' approximation theorem [70], we know that if a function is continuous in a closed interval, there exists a polynomial so that we can approximate

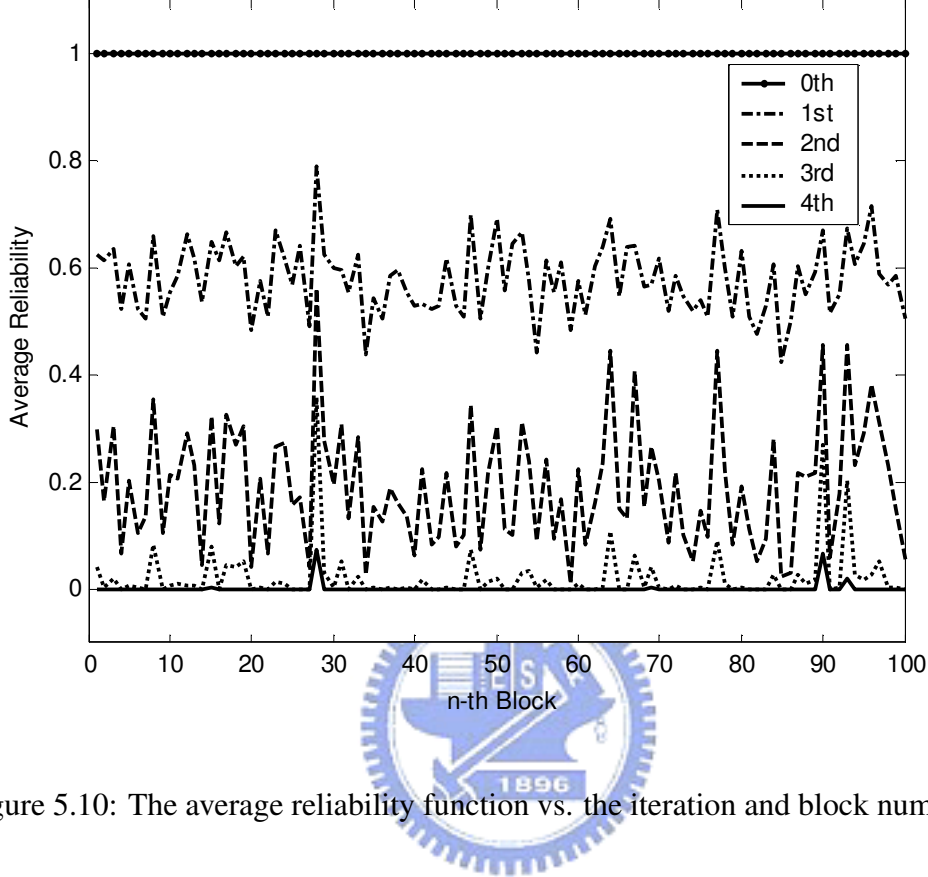


Figure 5.10: The average reliability function vs. the iteration and block numbers.

the function in that interval with infinitesimal error. In other words, if the first order derivative  $f'_j(\bar{v})$  exists in  $[0, 1]$ ,  $f_j(\bar{v})$  is interpolatable with any infinitesimal error. In what follows, we will prove that  $f_j(\bar{v})$  is not only continuous but also an analytic, i.e., the  $n$ -th order derivative  $f_j^{(n)}(\bar{v})$  always exists.

From linear algebra, we know that the inversion for a given square matrix  $\mathbf{A}$  can be calculated with the following general formulae,

$$\mathbf{A}^{-1} = \frac{\text{adj}(\mathbf{A})}{\det(\mathbf{A})}, \quad (5.91)$$

where  $\text{adj}(\mathbf{A})$  is the adjoint matrix and  $\det(\mathbf{A})$  is the determinant of  $\mathbf{A}$ , respectively. Define  $M_{mn}$  as the determinant of the matrix without the  $m$ -th row and the  $n$ -th column. we can obtain the adjoint matrix as

$$\text{adj}(\mathbf{A}) = [(-1)^{(m+n)} M_{mn}]^T. \quad (5.92)$$



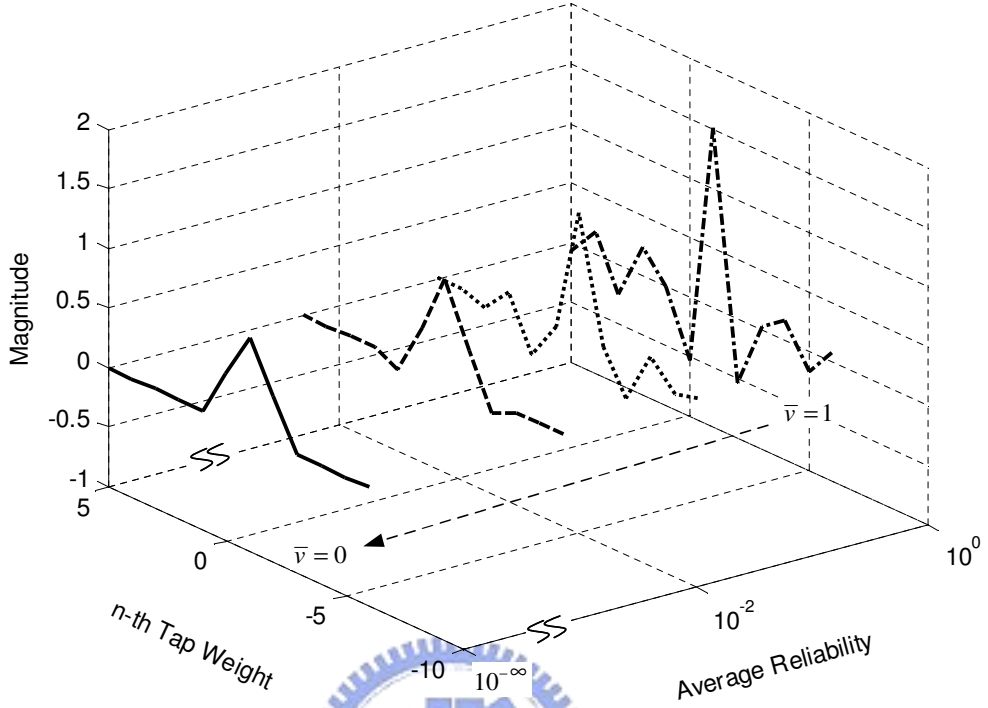


Figure 5.11: Optimal block time-invariant filter vs. the reliability function (Channel: Proakis B channel with 4-PAM at SNR=18.0 dB).

Evaluating  $f$  with (5.90) with (5.91), we can have the  $j$ -th tap weight of an optimal filter as

$$f_j(\bar{v}) = \frac{\sum_{n=0}^{N-1} q_{n,j} \bar{v}^n}{\sum_{n=0}^N p_{n,j} \bar{v}^n} \quad (5.93)$$

where  $p_{n,j}$  and  $q_{n,j}$  are some constants,  $N$  is the filter length, and  $\bar{v} \in [0, 1]$ . Note that  $f_j(\bar{v})$  is a rational function with a numerator polynomial of degree  $N - 1$  over a denominator polynomial of degree  $N$ . Note that the denominator cannot be zero. Otherwise, the optimal solution will not exist. Thus, the  $n$ -th order derivative  $f_j^{(n)}(\bar{v})$  always exists and  $f_j(\bar{v})$  is an analytic function. The analytic property enables use to interpolate the optimal filters achieving computational reduction. We can interpolate each individual tap weight with an set of interpolation parameters. However, for filters with hundreds of tap weights, the complexity for the interpolation is high. For simplicity, we propose to use a suboptimal scheme that the same set of interpolation

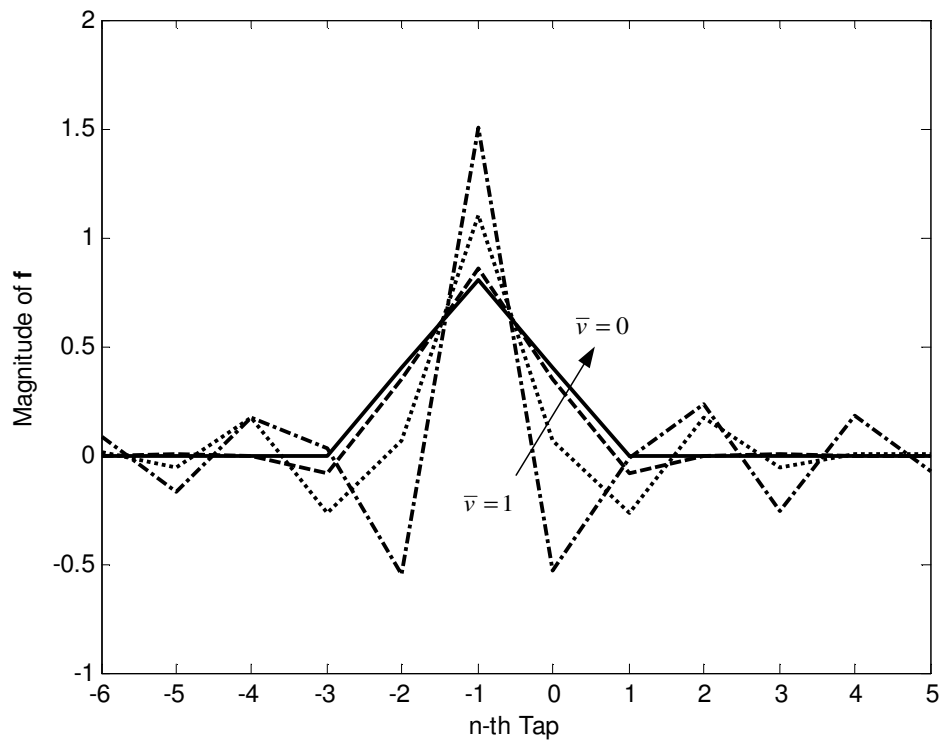


Figure 5.12: Optimal block time-invariant filter vs. the reliability function (Channel: Proakis B channel with 4-PAM at SNR=18.0 dB).

parameters are used for all tap weights.

In this dissertation, we apply the piecewisely linear Lagrange interpolation [70] scheme. Assume that  $(\bar{v}_1, f(\bar{v}_1))$  and  $(\bar{v}_2, f(\bar{v}_2))$  are known as *a priori* with (5.90) and  $\bar{v}_1 < \bar{v} < \bar{v}_2$ . For a single tap weight, we can has the linear interpolation as

$$f_j(\bar{v}) = wf(\bar{v}_1) + (1 - w)f(\bar{v}_2) \quad (5.94)$$

where  $w$  is a weighting factor. The weight factor is given by

$$w = \frac{\bar{v} - \bar{v}_2}{\bar{v}_1 - \bar{v}_2}. \quad (5.95)$$

we can the partition the full range of reliability  $[0, 1]$  into a number of adjoined intervals and approximate  $f_j(\bar{v})$  in each interval with a piecewisely linear function.

As mentioned, we use a suboptimal interpolation scheme. For an optimal filter, assume that  $(\bar{v}_1, \mathbf{f}_1 = \zeta(\bar{v}_1))$  and  $(\bar{v}_2, \mathbf{f}_2 = \zeta(\bar{v}_2))$  are known as *a priori* with (5.90) and  $0 \leq \bar{v}_1 < \bar{v}_2 \leq 1$ . For a reliability  $\bar{v}_i$  located between the interval  $\bar{v}_1 < \bar{v}_i < \bar{v}_2$ , the corresponding optimal filter  $\mathbf{f}_i = \zeta(\bar{v}_i)$  can be interpolated by two given optimal filters.

$$\mathbf{f}_i \approx \tilde{\mathbf{f}}_i \triangleq \xi(\mathbf{f}_1, \mathbf{f}_2) \quad (5.96)$$

where  $\tilde{\mathbf{f}}_i$  is the interpolated optimal filter and  $\xi(\cdot)$  is the linear interpolated function as

$$\xi(\mathbf{f}_1, \mathbf{f}_2) = w\mathbf{f}_1 + (1 - w)\mathbf{f}_2. \quad (5.97)$$

where  $w$  is a weighting factor. The weight factor is given by

$$w = \frac{\bar{v}_i - \bar{v}_2}{\bar{v}_1 - \bar{v}_2}. \quad (5.98)$$

Thus, all coefficients of optimal filters are interpolated with the same set of parameters. From our experience, a better approximation may be obtained in terms of the square root of reliability (i.e., standard deviation of soft decisions). However, the square root operation is required and this will complicate the computation. To compensate the possible performance loss, we make the interpolation interval smaller, which will only increase implementation cost slightly.

Define the normalized interpolated error (NIE) as

$$\text{NIE} \triangleq \frac{\|\tilde{\mathbf{f}}_i - \mathbf{f}_i\|^2}{\|\mathbf{f}_i\|^2}. \quad (5.99)$$

If  $\bar{v}_1, \bar{v}_2$  are close enough, the interpolated optimal filter  $\tilde{\mathbf{f}}_i$  will close to the optimal filter  $\mathbf{f}_i$ , too. The reliability is within the range  $[0, 1]$  and we can select a set of reference reliabilities  $\{\bar{v}_j\} = \{0, \bar{v}_1, \bar{v}_2, \dots, 1\}$ . If the number of the reference reliabilities is large enough, the NIE will be small and the performance loss will be ignorable. Let the full range of reliability be partitioned by  $Z$  sub-intervals, i.e., the number of reference reliabilities be  $(Z + 1)$ . We can then calculate  $(Z + 1)$  corresponding reference optimal filters and store the filter coefficients in a table. For any reliability, we can then look up the table and obtain the corresponding optimal filter through interpolation. This will dramatically reduce the computational requirement for the LSL equalizer. The next problem is how to determine the values of reference reliabilities. This can be seen as a sampling problem and the simplest one is an uniform sampling scheme as

$$\bar{v}_j = j\Delta, j = \{0, 1, \dots, Z\}. \quad (5.100)$$

where  $\Delta = 1/Z$  is the sampling spacing. However, simulations show that the NIE performance is satisfactory for large  $Z$  only. The reason can be explained below. For a specific tap weight, let its value corresponding to  $\bar{v} = 1$  be  $f_N$  and its value corresponding to  $\bar{v} = 0$  be  $f_P$ . A uniform sampling for the reliability between  $\bar{v} = 1$  does not give a uniform sampling for the tap weight between  $f_N$  and  $f_P$ .

Thus, the reference reliabilities should be non-uniformly sampled. The goal is to make the corresponding optimal filter weights be uniformly sampled (because of piecewise linear interpolation). Although we can formulate an MMSE cost function to the nonlinear sampling problem and find the optimal solution by the generalized Pontryagin's maximum principle [71], it will be time-consuming. Here, we only select a common nonlinear function to the job. We observe that when the reliability is large, the variation of optimal weights are small. However, when the reliability is small, the variation will be large. We then need denser sampling when

reliability is small. This motivate us to use an exponential function for sampling. The sampling scheme is given by

$$\bar{v}_j = \begin{cases} e^{-(\frac{\lambda}{Z-1}) \cdot j} & j = \{0, 1, \dots, Z-1\} \\ 0 & j = Z \end{cases}, \quad (5.101)$$

where  $\lambda$  is a factor controlling the decaying rate of the exponential function. The reliability with zero value is considered as the perfect *a priori* case. As we can see, the larger the entries are, the smaller the interpolated error will be. But, the the table size will become larger. There is a tradeoff between interpolation performance and table size.

From (5.101), we also see that the decaying factor also determines the minimum (except zero) reliability value, namely  $\bar{v}_{Z-1} = e^{-\lambda}$ . The minimum reliability value is crucial to both NIE and BER performance. This is because when the number of iteration or the SNR is high, the corresponding reliability will be very small. It will fall into the region between  $\bar{v}_{Z-1} = e^{-\lambda}$  and  $\bar{v}_Z = 0$ . The interpolation in this region becomes critical. The decaying factor is determined through some trial-and-errors. This will be discussed in simulations section later.

The sampled reference reliability  $\bar{v}_j$  begins at  $\bar{v}_0 = 1$  corresponding no *a priori* case. Then, its value is exponentially reduced, and finally ends with  $\bar{v}_Z = 0$  corresponding to the perfect *a priori* case. If the channel is time-invariant, it can be identified during the initialization stage. Also in this stage, we can then calculate a set of reference optimal filters  $\{\mathbf{f}_j\} = \{\mathbf{f}_0, \mathbf{f}_1, \dots, \mathbf{f}_Z\}$  (5.90) according to the sampled reliability. Later, for any reliability  $\bar{v}_i^n$ , we can approximate the optimal filter  $\mathbf{f}_i^n$  with the interpolation shown in (5.97). For any iteration and any block, we do not have to re-calculate the optimal filter using (5.90). Thus, the computational complexity can be very low. Before interpolation, we have to perform a binary search to locate the interpolation interval ( $\bar{v}_1 < \bar{v}_i^n < \bar{v}_2$ ). The required operations are  $\log_2(Z+1)$  comparisons only. Our simulations shown that 16 sampling points are large enough for a channel with length up to hundreds of taps. That means only four comparison operators are required and the complexity is ignorable. The complexity for computing reference optimal filters,  $\{\mathbf{f}_i^n\}$ , is  $(Z+1) \cdot O(N^3)$ . This is also ignorable since we only have to carry out the operations once. Unless the channel

response is changed, we do not have to re-calculate the reference filters.

### § 5.3.2 FDISL Equalizer

For the channel with hundreds of taps, the application of a turbo equalizer is difficult, if not impossible. Even for the LSL equalizer, we may need one million operations to compute an optimal filter. The complexity of LSLH equalizer is very low; however, the performance is usually not satisfactory. For a long response, the convergence behavior of a turbo equalizer become more sensitive and difficult to control. Despite the problem, the filter-based turbo equalizer is still the only possible candidate to apply. In the previous subsection, we have developed an interpolation scheme dramatically reducing the computational complexity of LSL equalizer. Note that the scheme is to interpolate the whole response of an optimal filter. We call this a whole response (WR) interpolation scheme.

Inspecting the structure of LSL equalizer in Fig. 5.13 or (5.72), we find that the equalizer consists of two separate linear filters. One is the optimal filter  $f$  we have been working with, and the other is a filter with soft-decisions as its inputs and channel responses as its coefficients. The length of optimal filter is usually on the same order of the channel length. Thus, the equalization operation will require high computational complexity when the channel length is long. If the response of the channel and optimal filter can be interpolated, the complexity can be reduced further. For wireline channels, this is indeed possible. Note that the scheme here is to interpolate an individual sample of a channel or a filter response and we call this an individual response (IR) interpolation scheme. Note that what discussed in the previous chapters belongs to the IR interpolation schemes. Combining the WR and IR interpolation schemes, we obtain the FDISL equalizer with very low computational complexity.

Similar to the interpolated echo canceller in Chapter 3, the channel response used in (5.72) can be approximated by a low-complexity interpolated FIR (IFIR) filter. For convenience, we rewrite the soft observation again.

$$\bar{y}_k = \mathbf{H}\bar{x}_k. \quad (5.102)$$

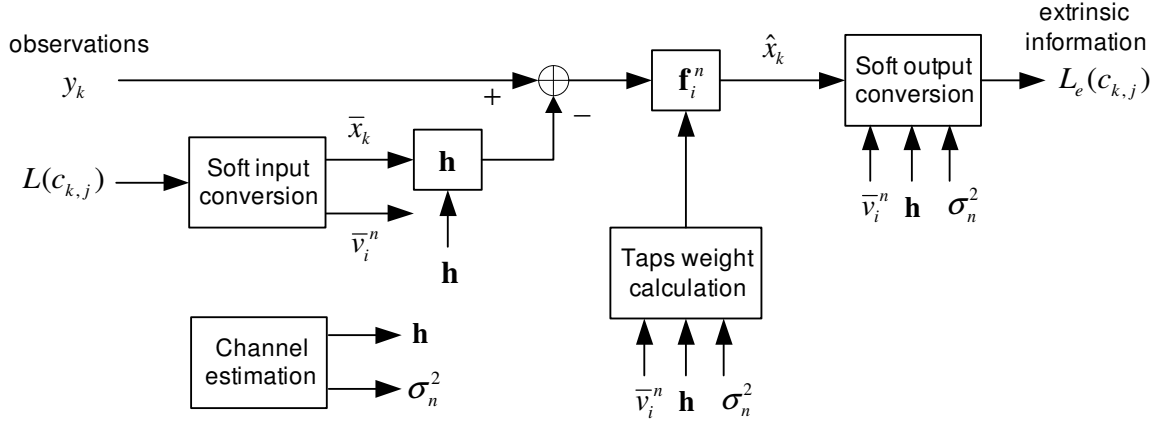


Figure 5.13: The detailed filter structure of the LSL.

Note that every row in the channel matrix  $\mathbf{H}$  contains the channel response  $\mathbf{h}^T$ . In fact, the soft observation can be seen as the received signal with the *a priori* information  $\bar{x}_k$  as transmitted signal over channel  $\mathbf{h}$ . This suggests that the application of the low-complexity IFIR filtering scheme. The approach is similar that described in Chapter 3. The only difference is that the transmit channel instead of the echo channel is considered now. Assume the channel response can be partitioned into two portions, i.e.,

$$\mathbf{h} = \begin{bmatrix} \mathbf{h}_h \\ \mathbf{h}_t \end{bmatrix}, \quad (5.103)$$

$$\mathbf{h}_h = [h_0, h_1, \dots, h_{t-1}]^T, \quad (5.104)$$

$$\mathbf{h}_t = [h_t, h_{t+1}, \dots, h_{t+\mu}]^T, \quad (5.105)$$

where  $t$  is a cutting point selected to partition the channel response into an interpolatable (tail) response  $\mathbf{h}_t$  and a transient (head) response  $\mathbf{h}_h$ . Note that the length of interpolatable response is usually much greater than the length of transient response in DSL channels, i.e.,  $(\mu - t) \gg t$ . Then, the channel response can be approximated by

$$\mathbf{h} \approx \mathbf{w}_1 + \mathbf{g} * \mathbf{w}_2^{\uparrow U}, \quad (5.106)$$

$$\mathbf{g} = [g_{-(U-1)}, g_{-(U-2)}, \dots, g_{(U-1)}]^T, \quad (5.107)$$

$$\mathbf{w}_1 = [w_{1,0}, w_{1,1}, \dots]^T, \quad (5.108)$$

$$\mathbf{w}_2^{\uparrow U} = [w_{2,0}, \underbrace{0, \dots, 0}_{U-1}, w_{2,1}, \underbrace{0, \dots, 0}_{U-1}, w_{2,2}, \dots]^T, \quad (5.109)$$

where  $U$  is the interpolation factor,  $\mathbf{g}$  is the interpolation filter,  $\mathbf{w}_2^{\uparrow U}$  is an  $U$ -downsampled-then-upsampled response,  $\mathbf{g} * \mathbf{w}_2^{\uparrow U}$  is the IFIR response for the interpolatable channel response  $\mathbf{h}_t$  (\* stands for the convolution operation), and  $\mathbf{w}_1$  is the compensated response for  $\mathbf{h}$  at head portion. We do not use the adaptive algorithm or the Wiener solution as that in Chapter 3 to obtain  $\mathbf{w}_1$  and  $\mathbf{w}_2^{\uparrow U}$ . Instead, we use a simple method.

Assume that  $\mathbf{h}$ ,  $\mathbf{g}$ , and a cutting point  $t$  are given. We simple let  $\mathbf{w}_2^{\uparrow U}$  be the direct  $U$ -downsampled-then-upsampled tail response, i.e,  $w_{2,i} = h_{t+iU}$ , and  $\mathbf{w}_1$  be the head portion of  $\mathbf{h}_i$  subtracted from the interpolated tail response. With mathematical expressions, we have

$$\mathbf{w}_2^{\uparrow U} = [h_t, \underbrace{0, \dots, 0}_{U-1}, h_{t+U}, \underbrace{0, \dots, 0}_{U-1}, h_{t+2U}, \dots]^T, \quad (5.110)$$

$$\mathbf{w}_1 = \mathbf{h} - \mathbf{g} * \mathbf{w}_2^{\uparrow U}. \quad (5.111)$$

If we use standard interpolation functions,  $w_{2,i}$  will be equal to the sampled value of  $h_{t+iU}$  exactly. However, if we use optimized functions,  $w_{2,i}$  may not be equal to the sampled value of  $h_{t+iU}$ . This does no harm to the result since the overall modelling error is generally smaller. With proper choice of parameters, the modelling error in with (5.110) is ignorable. The computational complexity for obtaining  $\mathbf{w}_1, \mathbf{w}_2^{\uparrow U}$  is also ignorable. Given a channel response, the optimal interpolation filter  $\mathbf{g}$  can be found offline by the least-squares solution described in Chapter 3. The least-squares solution is capable of minimizing the interpolation error with a single channel response or a set of responses. The later is very useful for DSL applications since the channel responses are usually similar. We can apply the optimization scheme to the optimal filter interpolation discussed next.

For the optimal filters, the problem is a little bit more complex since the optimal filter  $\mathbf{f}_i^n$  is not only iteration dependent but also block dependent. In other words, there are many filters to be interpolated instead of a single one. Fortunately, with our previous development, we only



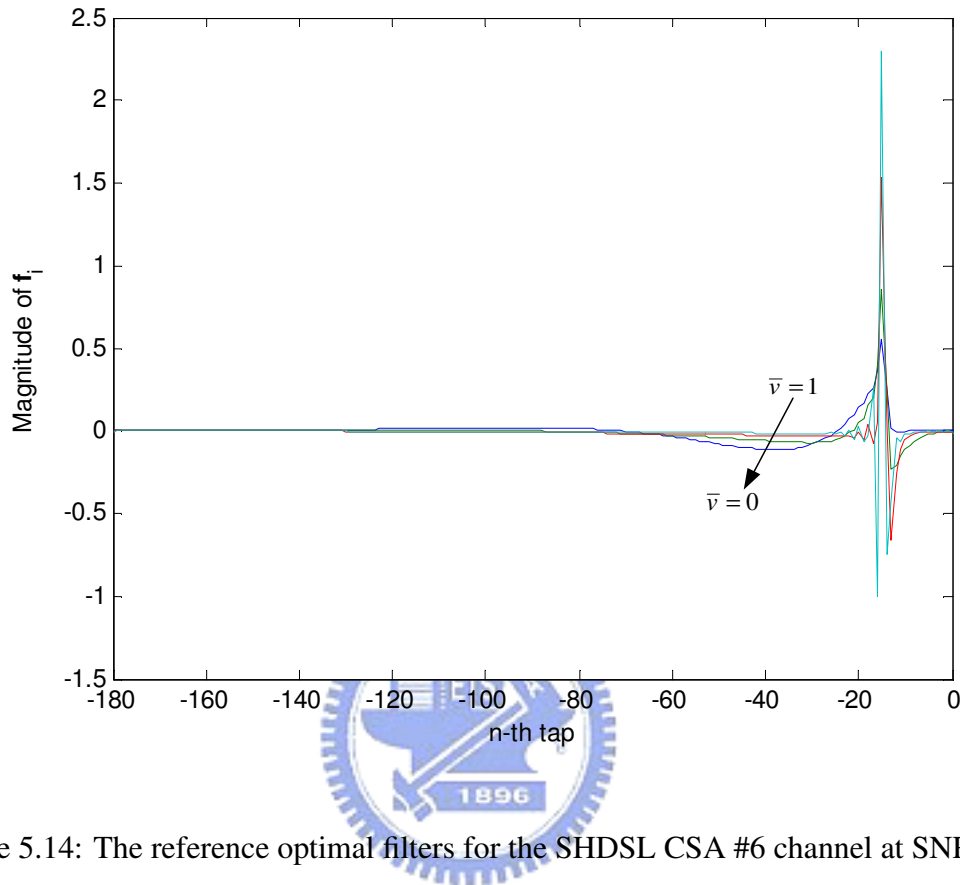


Figure 5.14: The reference optimal filters for the SHDSL CSA #6 channel at SNR=18.0 dB.

have to interpolate a set of reference optimal filters. The problem is that will all the reference optimal filters be interpolatable? Fig. 5.14 shows the reference optimal filters for CSA #6 channel [1, 72] with 4-PAM (SNR=18.0 dB). As we can see that the optimal filter responses varies a lot. For the case where reliability is close to one (no *a priori*), the filter degenerates to the conventional linear equalizer. Since the channel has lowpass response, the equalizer will have highpass response. This make the interpolation difficult. However, the equalizer response is usually short in this case. In addition, this case is often observed in early iteration stages and larger error is tolerable. For the case where reliability is close to zero (perfect *a priori*), the response becomes a matched filter, a time reversal version of the channel response, and interpolation is easy to apply. Note that the cutting point may be different for different

reliability values. Thus,  $\mathbf{f}_i^n$  can be approximated by

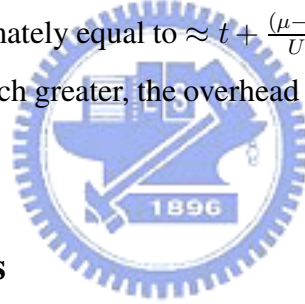
$$\mathbf{f}_i^n \approx \mathbf{g} * \mathbf{w}_4^{\uparrow U} + \mathbf{w}_3. \quad (5.112)$$

Since  $\mathbf{f}_i^n$  is interpolatable at its head portion (which is time reversal compared to channel response), we have  $t \gg (\mu - t)$ . Now,  $\mathbf{g} * \mathbf{w}_4^{\uparrow U}$  models the head response while  $\mathbf{w}_3$  the tail response. Similar to (5.110), the interpolated tap weights are obtained by direct  $U$ -downsampled-and-upsampled on  $\mathbf{f}_i^n$  ( $w_{4,j} = f_{i,jU}^n$ ) and

$$\mathbf{w}_4^{\uparrow U} = [f_{i,0}^n, \underbrace{0, \dots, 0}_{U-1}, f_{i,U}^n, \underbrace{0, \dots, 0}_{U-1}, f_{i,2U}^n, \dots]^T, \quad (5.113)$$

$$\mathbf{w}_3 = \mathbf{f}_i^n - \mathbf{g} * \mathbf{w}_4^{\uparrow U}. \quad (5.114)$$

The overhead complexity is approximately equal to  $\approx t + \frac{(\mu-t)}{U}$  per block or  $(t + \frac{(\mu-t)}{U}) / K$  per symbol. Since the block length is much greater, the overhead complexity is ignorable (less than one per symbol).



### § 5.3.3 Complexity Analysis

A complete turbo equalizer consists of a SISO equalizer, a SISO decoder, an interleaver, and a deinterleaver. The detailed complexity analysis of all kinds of turbo equalizers are discussed in Appendix E. We only summarize the results here. The interleaver or deinterleaver is simply a table lookup operation and its complexity is minor for the overall system [55, pp. 127]. The filter-based SISO linear equalizer consists of linear equalization and the SISO conversion. The complexity of the SISO conversion is the same for the LSL equalizer and proposed SISO equalizers. From Table E.1, we can see that the complexity of SISO conversion is proportional to the signal constellation size. For a small constellation size (e.g. 8-ary modulation), the complexity is minor compared to that of equalization. Thus, we will focus on the equalization complexity only. Since the LSL achieves good performance and complexity tradeoff among all SISO linear equalizers proposed by Tüchler, we use the LSL equalizer as a reference.

For convenience, the computational complexity used here is in terms of FLOPS being able to combine different operations, such as multiplication, division, and table lookup, into a unified complexity index. A FLOP [73, pp. 19] is defined as a real number floating point operation (i.e., a floating point addition or a multiplication, with indexing).

Finding optimal tap weights of the LSL equalizer involves matrix inversion and there are various methods to do the job. A direct-matrix inversion method called Gaussian elimination [73, pp. 99] costs  $\frac{2}{3}N^3 + \frac{3}{2}N^2 - \frac{7}{6}N = O(N^3)$  FLOPS per block. Exploiting the Toeplitz structure of a channel matrix, one can apply the Levison method [73, pp. 187] and this will require  $4N^2 = O(N^2)$  FLOPS per block. Instead of direct matrix inversion, there is a FFT approximation method [55, pp. 111] achieving a lower complexity. The complexity is on the order of  $O(L \log_2 L)$ , where  $L$  is the channel length. Thus, we select the FFT approximation method for matrix inversion. The computational complexities per symbol per iteration for the

Table 5.2: Complexity of SISO equalizers without SISO conversion

| Equalizer type | FLOPS/symbol/iteration                             |
|----------------|--|
| LSL            | $\frac{10L \log_2 L + 12L + 3}{K} + 40L + 4N + 78$ |
| FSISL          | $(4N - 2) + 44$                                    |
| FDISL          | $R(4N - 2) + 44$                                   |

LSL, FSISL, and FDISL equalizers are summarized in Table 5.2. The table is extracted from Table E.2. From the table, we see that the complexity of the LSL equalizer is

$$\frac{10L \log_2 L + 12L + 3}{K} + 40L + 4N + 78 \text{ (FLOPS/symbol/iteration)}, \quad (5.115)$$

where  $L$  is the channel length,  $N$  is the filter length, and  $K$  is the block length. The complexity of FDISL equalizer is

$$R(4N - 2) + 44 \text{ (FLOPS/symbol/iteration)}, \quad (5.116)$$

where  $R \leq 1$  is the IR complexity ratio. The IR complexity ratio is defined as the complexity of a filter (or a channel) with IR interpolation over that without. This ratio is shown to be [15]:

$$\begin{aligned}
 R &= \frac{\left( \underbrace{\left\lfloor \frac{N-t}{U} \right\rfloor}_{IFIR} + \underbrace{t}_{FIR} + \underbrace{(s-1)U}_{\text{overlapped}} + \underbrace{(2sU-1)}_{\text{interpolation filter}} \right)}{N} \\
 &= \frac{\left( \left\lfloor \frac{N-t}{U} \right\rfloor + t + (3s-1)U - 1 \right)}{N}, \tag{5.117}
 \end{aligned}$$

where  $t$  is the cutting point,  $U$  is the interpolation factor, and  $s$  is the span of the interpolation filter. Note that the FSISL equalizer is a special case of FDISL equalizer with  $R = 1$ . For short channel length, the optimal filter is not interpolatable and no gain is achieved using the IR interpolation in FDISL. For longer channel length; however, the IR interpolation can reduce the complexity effectively. From (5.115), we see that the FFT method reduces the complexity dramatically compared to the Levinson method. As shown in Table E.2, it still requires many division operations. Comparing (5.116) with (5.115), we find that the proposed FDISL equalizer not only eliminates the matrix inversion for solving optimal filter tap weights, but also reduces the filtering operations. Also, the proposed fast turbo equalizer is nearly division-free.

For a CSA #6 channel [7, 8] with length 180 taps, the IR complexity ratio with different interpolation factor is shown in Fig. 5.15. From figure, we find that for interpolation factor from 4 to 16, complexity ratio is less than 40%. That means the choice of the interpolation factor is not sensitive and it is easy to obtain a tradeoff between the performance and complexity reduction. In our simulations, we select an interpolation factor of eight and the IR complexity ratio is 35%. The computational complexities for the proposed fast interpolated SISO linear equalizers and the LSL equalizer with different channel length are compared in Fig. 5.16. The corresponding complexity ratio is also shown in Fig. 5.17. Here, the complexity ratio of a compared equalizer is defined as the complexity of the equalizer divided by that of the LSL equalizer. For the FSISL equalizer, the complexity ratio is less than 13% when the channel length is longer than 20, and is less than 10% when the channel length is longer than 80. It is to say that the complexity of the FSISL is an order of magnitude less than that of the LSL equalizer

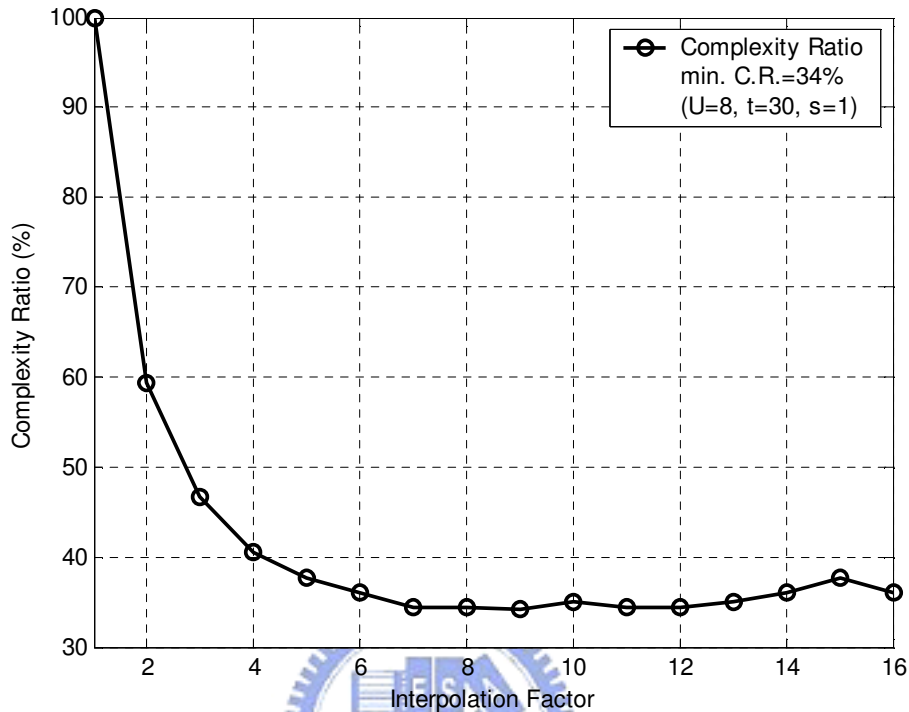


Figure 5.15: The IR complexity ratio vs. interpolation factor for the SHDSL CSA #6 channel.

for medium to long channels. For very short channels, the complexity ratio is less than 38%. For the CSA #6 channel, the complexity ratio of FSISL equalizer is only 9.5%. Since the IR complexity ratio is 34% for this channel, the complexity ratio of FDISL equalizer is only 3.7%. Note that the IR interpolation scheme can not offer any complexity reduction when channel length is less than 20.

## § 5.4 Simulation Results

In this section, we report some simulation and performance comparison results. To demonstrate the effectiveness and robustness of the proposed turbo equalizers, we consider channels shown in Table 5.3. The selected channels cover a wide range of ISI severity level as well as

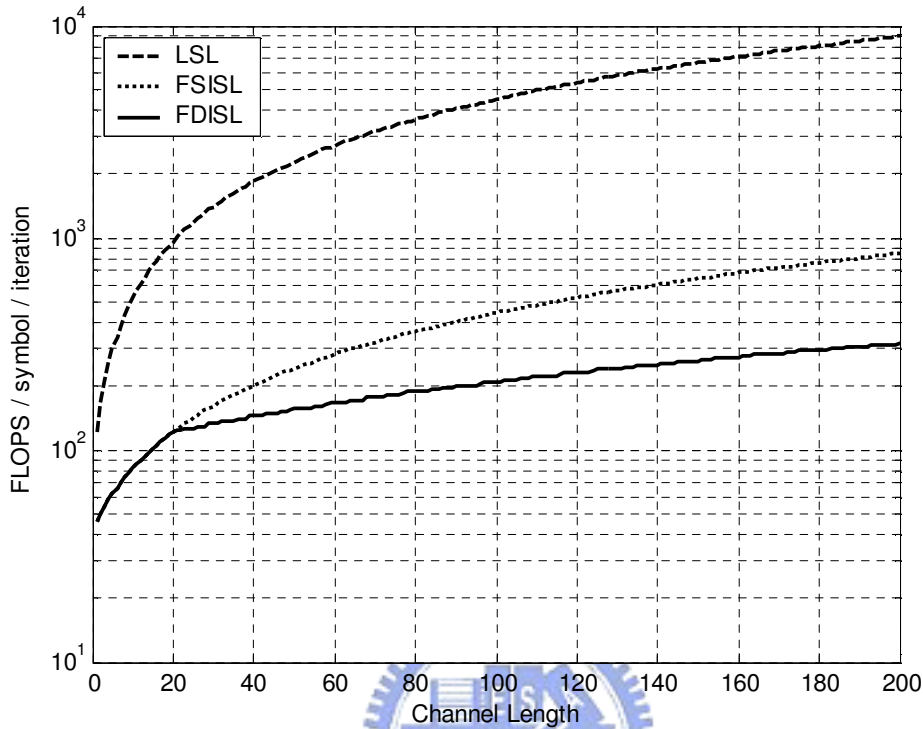


Figure 5.16: The computational complexity of proposed fast interpolated SISO linear equalizers (FSISL and FDISL) and LSL equalizer.

channel length. Proakis C channel possesses many spectral nulls and this can result in a poor equalization performance and convergence behaviors. The CSA #6 channel is a standard test loop for SHDSL transceiver [7, 8]. It is also the longest CSA test loops in standards and usually considered as the worst case in many works. Its response is shown in Fig. 5.18. As we can see, the channel length is almost two hundreds. To the best of our knowledge, there is no suitable turbo equalizer for such application. We use this channel to test the effectiveness of the proposed FDISL turbo equalizer. At the same time, emphasize the importance of the complexity reduction issue.

Without loss of generality, we use a bit-interleaved coded modulation (BICM) [74] transmission system. In the system, we use a half-rate non-systematic convolutional (NSC) code

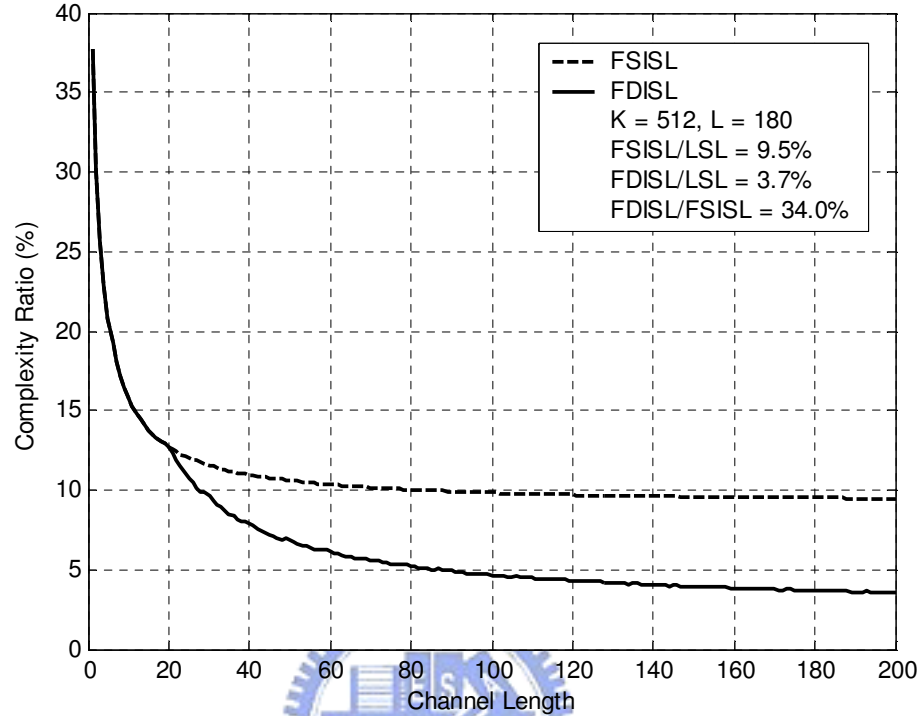


Figure 5.17: The complexity ratio of proposed turbo equalizers (block length: 512 symbols).

Table 5.3: Channel models for performance evaluation

| Channel type   | ISI severity | Channel length |
|----------------|--------------|----------------|
| Proakis A      | low          | medium         |
| Proakis C      | severe       | short          |
| CSA #6 (SHDSL) | medium       | long           |

$G = [5 \ 7]$ , an S-random bit interleaver, and a 4-PAM modulation scheme. The interleaver is a random interleaver with a block length 1024 and the spreading factor, S, is 22. The interleaver is obtained by a random search scheme [59]. Note that the permutation sequence is time-invariant

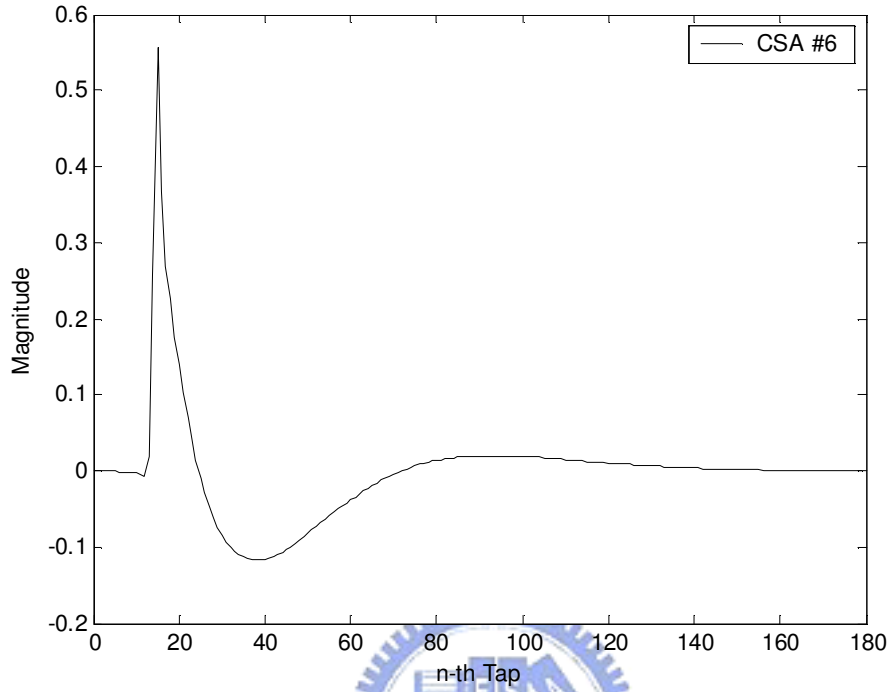


Figure 5.18: The channel response of CSA #6 channel for SHDSL application.

and is known to the transmitter and the receiver. The SNR is defined as follows

$$\text{SNR} \triangleq \frac{E[|y_k|^2]}{\sigma_n^2}. \quad (5.118)$$

Since the convergence behavior is channel and SNR dependent, we perform simulations with a wide range of SNR. In the receiver, the channel response and SNR is assumed to be perfectly known. In the proposed FSISL and FDISL turbo equalizers, there are several additional parameters needed to be determined. These parameters not only influence the equalization performance but also the required computational complexity. Thus, they must be determined properly.

### § 5.4.1 Parameters of Reference Filters

At the initialization stage, we have to find reference optimal filters and store them in a table. Given a channel and tolerable interpolated modelling error (i.e., an NIE threshold), we can



generate reliability according to (5.101) and then obtain the corresponding optimal filters. There are two parameters namely the number of entries  $Z$  and decaying factor  $\lambda$  here. We first select the number of entries  $Z$ , say 15 (16 entries), and then adjust the decaying factor recursively. The main point is to check if the weights have uniform distributions. If not, adjust (increase or decrease) the decaying factor. It is found that uniform weight distributions will lead to a better NIE performance. If the minimum NIE is still under the target performance, then increase the  $Z$  and repeat the process again. Fortunately, the procedure can be completed usually within few iterations. Fig. 5.14, 5.19, and 5.20 show the results.

A table with 16 entries and  $\lambda = 8$  is found to be good enough. For CSA #6 channel, the average NIE is -44.3 dB and the worst-case NIE is -39.7 dB. The parameters are also applicable to other short channels. From Fig. 5.19, we see that weight distributions are roughly uniform (though it is possible to improve further). Due to the exponentially sampling (more dense samples in higher reliability region), the NIE performance is better in the more reliable region. This is good for the turbo equalizer since the optimal filter is more sensitive in the small reliability region. In the earlier iteration stages, the system experiences the transition from no *a priori* to better *a priori*. The modelling errors in optimal filters are more tolerable. However, in the later stages, the system is about to output the results and more accurate modelling will be required. From Fig. 5.20, we see that the proposed equalizers approximate the LSL equalizer very well in all iterations. In our simulations, the total number of entries is 16 for all channels though 8 entries are good enough for short channels. Fig. 5.19 shows a portion of the reference optimal filters for CSA #6 channel with SNR 18.0 dB and filter length 180. Fig. 5.20 shows the corresponding NIE performance where the worst case is located in the insensitive region.

### § 5.4.2 Parameters for IR Interpolation

For FDISL equalizer, we have to determine additional parameters for the IR interpolation of the reference optimal filters. These parameters includes the cutting point, interpolation factor, interpolation span, and the interpolation filter. The target is to minimize the NIE performance

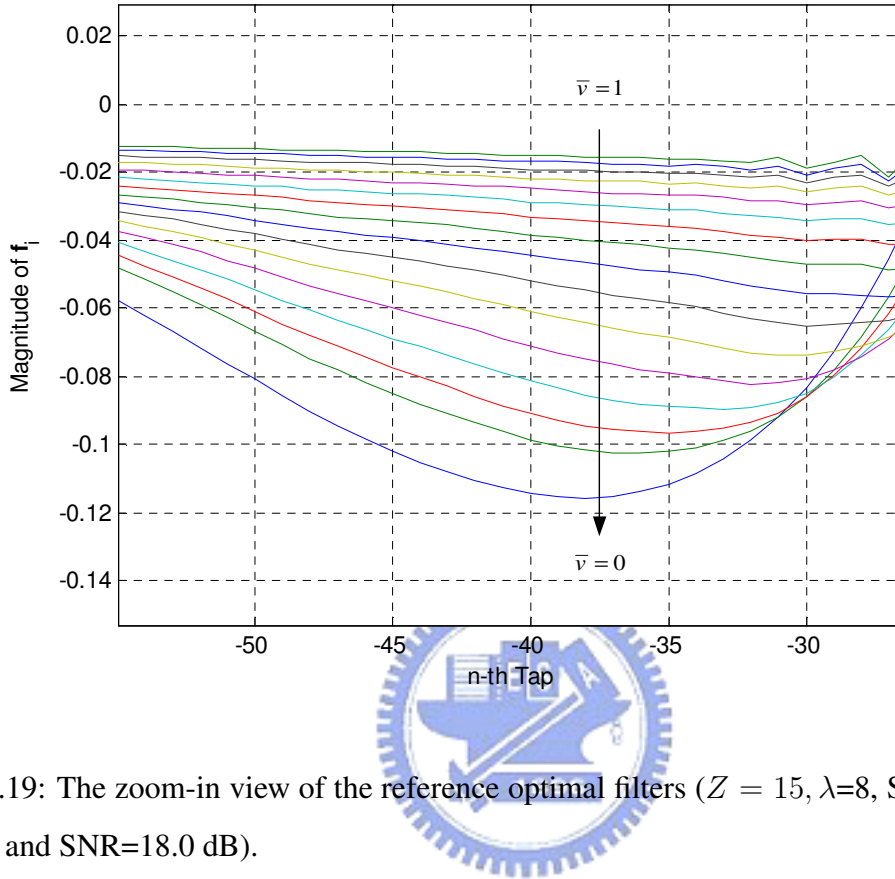


Figure 5.19: The zoom-in view of the reference optimal filters ( $Z = 15$ ,  $\lambda=8$ , SHDSL CSA #6 channel, and SNR=18.0 dB).

with the interpolation in (5.112). The parameters determined for optimal filters interpolation can also be used for channel interpolation in (5.106). Fig. 5.21 shows the interpolation NIE performance for the reference optimal filters in Fig. 5.19 and that for the CSA #6 channel response. Simulations show that with cutting point 30, interpolation factor 8, and interpolation span 1 [15], we can achieve an NIE performance less than -40.0 dB for all filters. Fig. 5.22 shows the interpolation filter optimized for CSA #6 channel. Note that those parameters are not only suitable for the CSA #6 channel but for all other test loops in SHDSL applications [15]. Here, we just use the CSA #6 channel as a testing example. As mentioned in previous sections, if we want to have better results in practical DSL applications, we should jointly optimize the parameters for all CSA loops instead of just a single one.

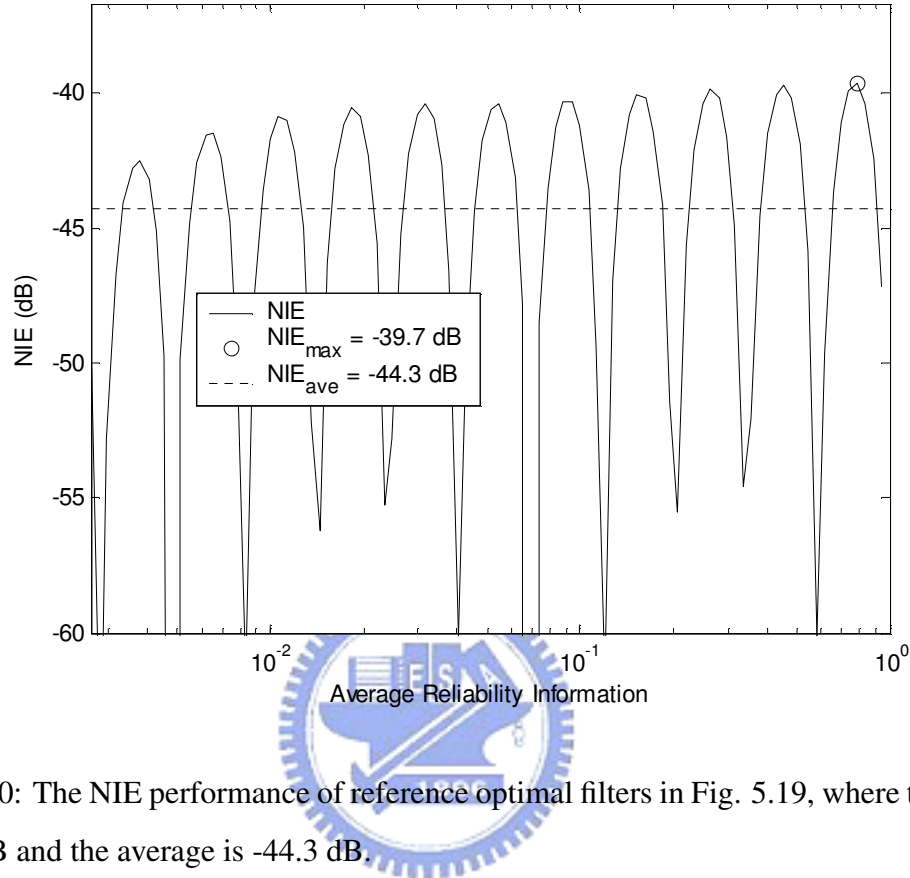


Figure 5.20: The NIE performance of reference optimal filters in Fig. 5.19, where the worst one is -39.7 dB and the average is -44.3 dB.

### § 5.4.3 BER Simulations

#### A) Low-ISI Proakis A Channel

Proakis A channel [49, 69] is a low-ISI channel with response  $\mathbf{h} = [0.04, -0.05, 0.07, -0.21, -0.5, 0.72, 0.36, 0, 0.21, 0.03, 0.07]^T$  (11 taps). We set the filter length as  $N_1 = 15$  and  $N_2 = 10$ . In addition to the BER performance turbo equalization, we also consider the BER performance of an uncoded and a (5,7) coded systems with AWGN channel. The former result can serve as the ISI severity indicator. If the performance of the zero-th iteration is much worse than the uncoded system, the channel ISI is considered as severe. The later result can serve as the performance bound for all filter-based turbo equalizers. Fig. 5.23 shows the BER performance of the LSL

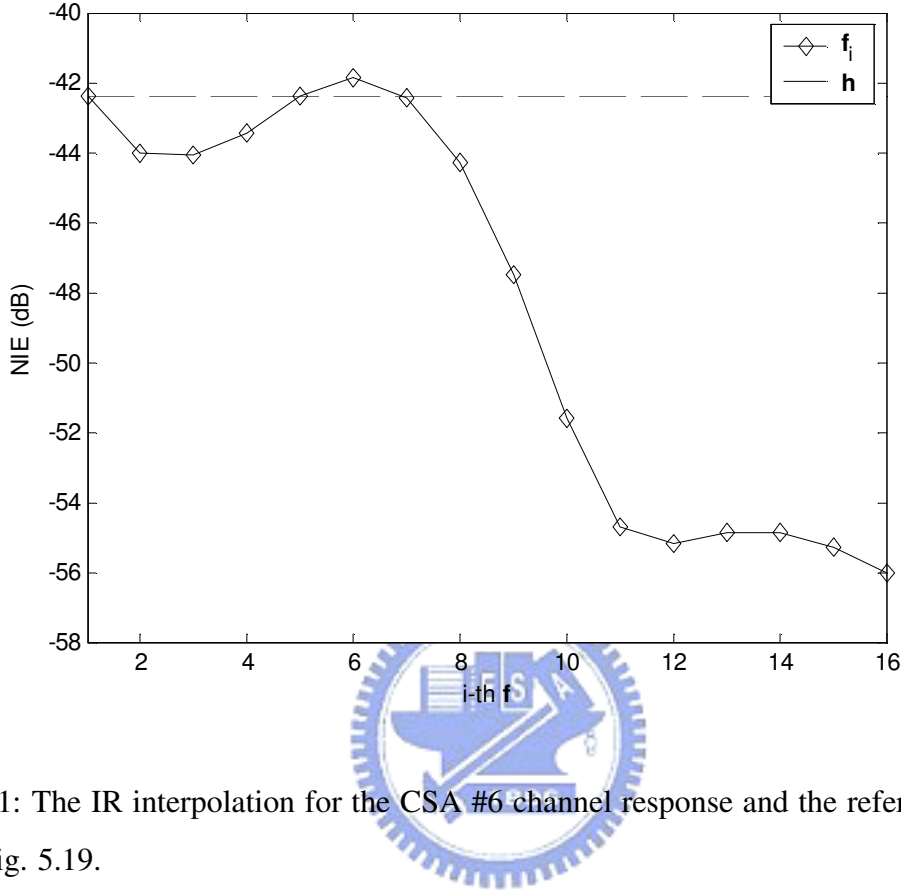


Figure 5.21: The IR interpolation for the CSA #6 channel response and the reference optimal filters in Fig. 5.19.

equalizer and the proposed FSISL equalizer in which  $Z = 15$  and  $\lambda = 8$ . The complexity ratio is 15.3%. As we can see, the convergence behavior and the performance of both equalizer are nearly the same. Since the channel is with low-ISI, both turbo equalizers reach the performance bound quickly after only two iterations.

### B) Severe-ISI Proakis C Channel

Proakis C [49, 69] channel is a severe-ISI channel with response  $\mathbf{h} = [0.227, 0.460, 0.688, 0.460, 0.227]^T$  (5 taps). The response has many spectral nulls and is difficult to equalize with a linear equalizer. Unfortunately, the LSL turbo equalizer embodied a linear equalizer at its first iteration and this results in a poor performance. The filter length is set as  $N_1 = 15$  and  $N_2 = 15$ .

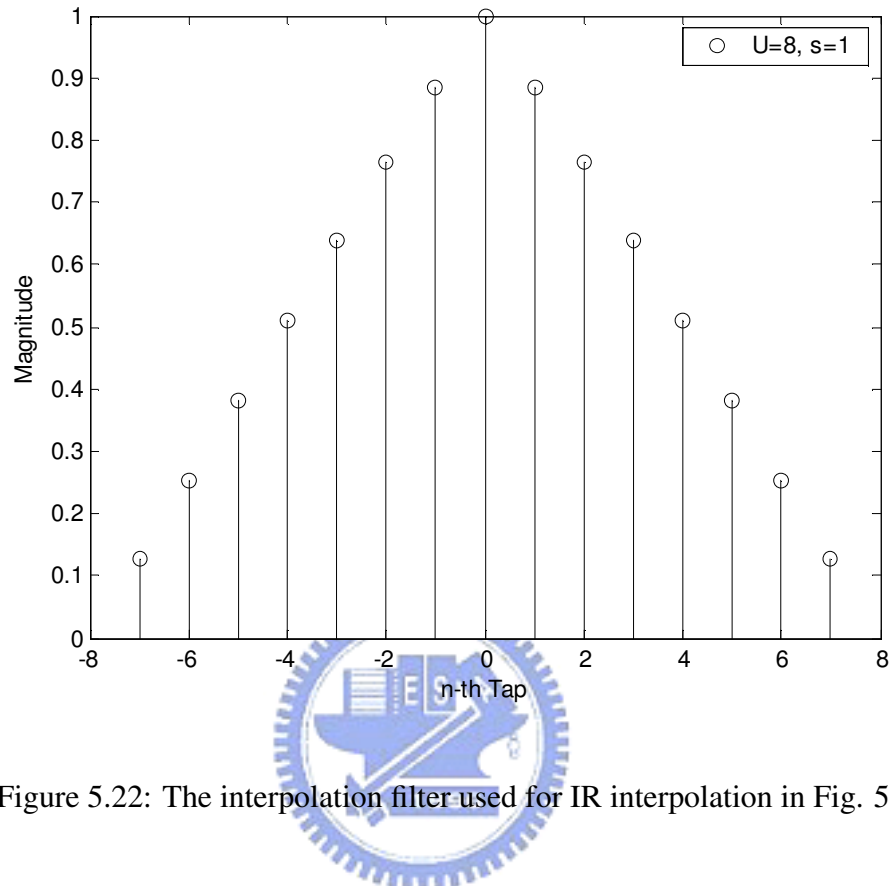


Figure 5.22: The interpolation filter used for IR interpolation in Fig. 5.21.

Fig. 5.24 shows the BER performance of the LSL and the proposed the FSISL turbo equalizers ( $Z = 15$  and  $\lambda = 8$ ). The complexity ratio is 20.8%. Different from the low-ISI Proakis A channel, the channel is with severe-ISI and the SNR threshold for triggering performance improvement is as high as 18 dB. However, the convergence behaviors and performances are still nearly the same for both turbo equalizers.

### C) Long ISI CSA #6 Channel

In contrast to previous hypothetical short channels, the CSA #6 channel is an actual channel modelled with the transmission line theory [1, 72]. The channel consists of a transmit shaping filter, a transmit differential hybrid circuit (with a  $135\Omega$  termination impedance), a 9-Kft 26 gauge loop, a receive differential hybrid circuit, and a receive filter. The transmit/receive filter

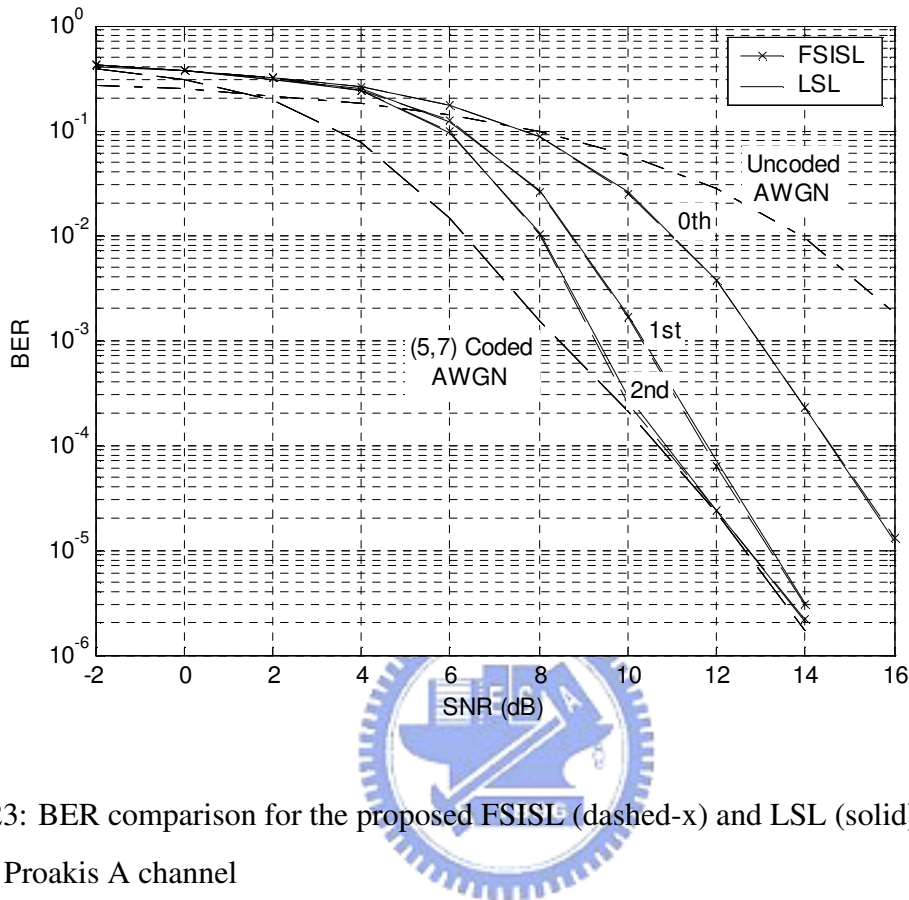


Figure 5.23: BER comparison for the proposed FSISL (dashed-x) and LSL (solid) turbo equalizers over Proakis A channel

was modelled as a 6-th order Butterworth lowpass filter with a 3-dB cutoff frequency at 775 KHz. The primary inductance for the transformer in the hybrid circuit is set as 3 mH.

The CSA #6 channel is an 180-tap medium-ISI channel as shown in Fig. 5.18. Such a long channel is used in standards [7, 8] for testing the implementation interoperability. The filter length is set as  $N_1 = 180$  and  $N_2 = 0$ . The channel is long and interpolatable such that we can apply the IR interpolation scheme. Fig. 5.25 shows the BER performance of the proposed FSISL, FDISL, and LSL turbo equalizers. Note that 16 reference filters and  $\lambda = 8$  are still used for FSISL and FDISL turbo equalizers. For the FDISL equalizer, the IR interpolation parameters are those obtained in Section 5.4.2. The complexity ratio of FSISL and FDISL equalizer over the LSL equalizer is 9.5% and 3.7%, respectively. Since the channel is with medium-ISI,

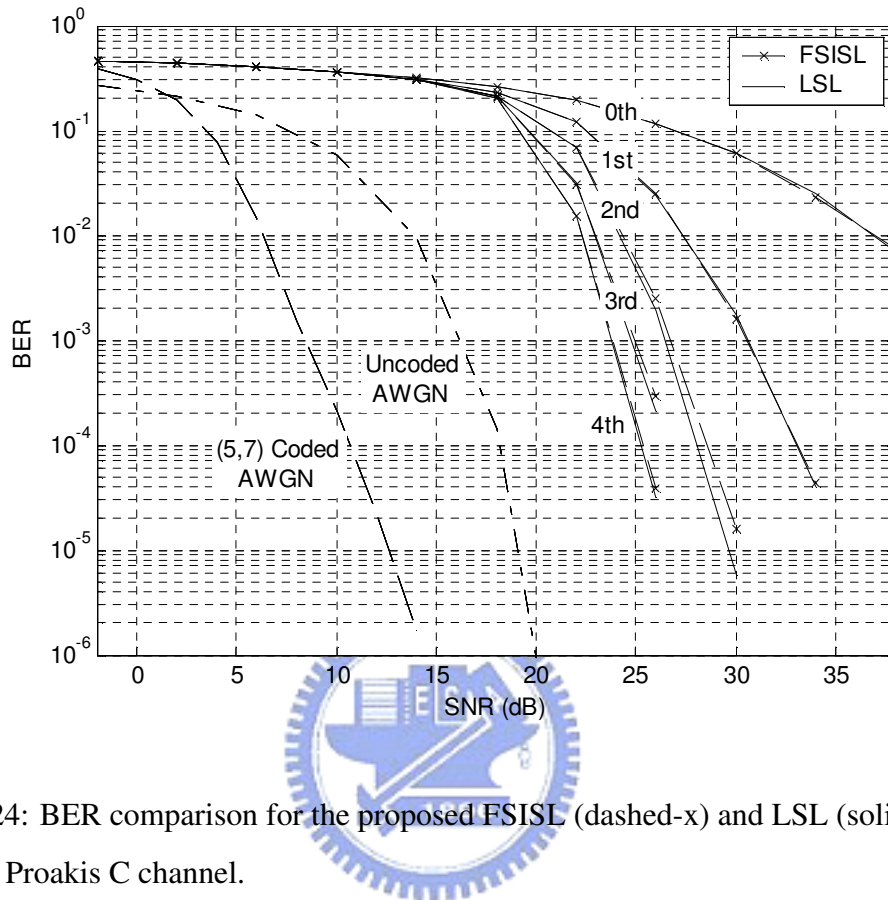


Figure 5.24: BER comparison for the proposed FSISL (dashed-x) and LSL (solid) turbo equalizers over Proakis C channel.

the SNR threshold is at 14 dB. Unfortunately, as we can see the BER oscillation phenomenon occurs. For a turbo equalizer, the BER is usually expected to be improved when more iteration is applied. However, when the oscillation phenomenon occurs, the BER performance will reach a bound after some iterations and become worse if more iterations are applied. Note that even the LSL turbo equalizer suffers from the same problem. The phenomenon is not uncommon and has been reported in [75]. Most previous works used the EXIT chart [67], which is a popular semi-analytical tool analyzing the convergence behavior of the turbo algorithms, to predict the convergence behavior. However, the EXIT chart is not able to consider the BER oscillation phenomenon and it cannot guarantee the convergence.

In what follows, we try to explain the phenomenon. Similar to conventional DFE, the turbo

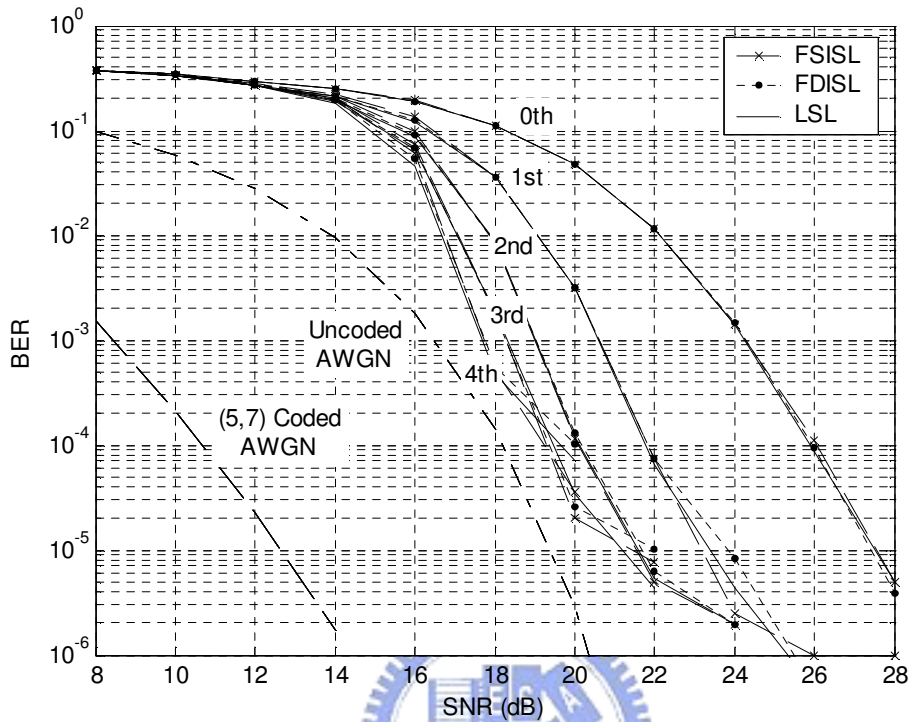


Figure 5.25: BER comparison for the proposed FSISL (dashed-x), FDISL (dotted-point), and LSL (solid) turbo equalizers over SHDSL CSA #6 channel

equalizer also uses decision feedback. The difference lies in that the decision is soft. As a result, there also exists the error propagation problem and this may cause divergence in the worst case. The severity of error propagation depends on the error correcting capability of the SISO decoder and the length of feedback filter. If the error correcting capability is not good enough, the turbo equalizer will become more sensitive when the iteration number is high. In this case, the LLR will become higher and higher and the soft decision will approach the hard decision; a wrong decision will cause severe error propagation (than soft decision). Additional iterations will amplify the error propagation effect. Since the channel is long, errors are easily accumulated in a long feedback path especially when the error bits are uncorrectable to the decoder. In the derivation of all kind of filter-based turbo equalizers, we assume that the distribution of soft



decision is Gaussian and this may not be true. The modelling assumption may also cause the unexpected phenomenon.

We propose a simple method to mitigate the BER oscillation phenomenon. Observe that the output from equalizer is connected to the input of decoder as *a priori* [62]. The soft value is a  $\tanh(\cdot)$  function of the LLR value (see Table 5.1). Thus, as the LLR becomes larger, the soft decision will approach the hard decision. The idea to avoid the oscillation problem is to limit and the maximum soft value. We propose to clip the LLR (equalizer's output) when it is larger than six. The clipping threshold is obtained by trial-and-error. Note that the number of six in LLR will correspond to a probability nearly to one. That means the clipping approach only change the behavior of high LLR values. Thus, clipping in nearly hard decision will result in *softer* inputs to the SISO decoder. This makes the turbo equalizer more *softer* and will alleviate the error feedback problem. Fig. 5.26 shows the BER performance with the clipping approach. The BER oscillation phenomenon no longer exists, but the error floor starts at higher SNR region. With four iterations, the turbo equalizers achieve 8.8 dB performance gain at BER  $10^{-5}$ . From Fig. 5.25 and Fig. 5.26, we can see that the performance and convergence behaviors of proposed turbo equalizers and the LSL turbo equalizer are nearly the same.

#### § 5.4.4 Discussions

From simulations, we found that the performance of filter-based turbo equalizer is close to the MAP equalizer in Proakis A channel. It is shown in [45] that the performance of filter-based turbo equalizer is far from the MAP equalizer in Proakis C channel. In other words, there still have some performance gaps between the filter-based turbo equalizer and trellis-based MAP equalizer for some severe-ISI channels. Fortunately, Proakis C channel is just a hypothetical channel. In real cases, most channels may not have such severe ISI. As we can see, the performance of the filter-based turbo equalizer is still far from Shannon's capacity (e.g., in the case of CSA #6 channel). The convolutional code used here is just a simple error correction code. It is known that the BER performance for a system is influenced by the response, the

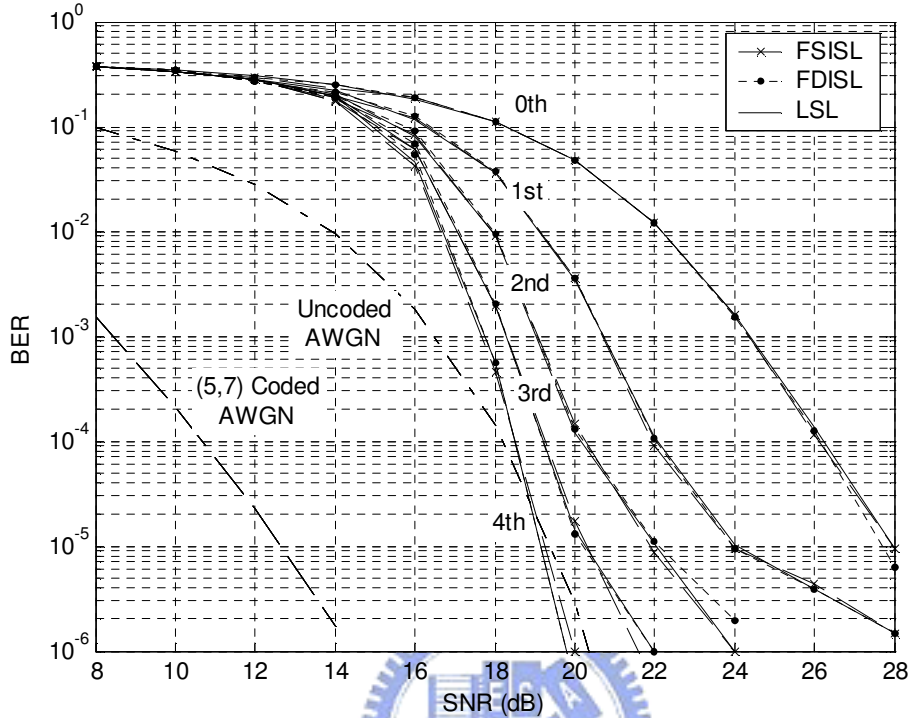


Figure 5.26: BER comparison for the proposed FSISL (dashed-x), FDISL (dotted-point), and LSL (solid) turbo equalizers over SHDSL CSA #6 channel (the LLR is clipped).

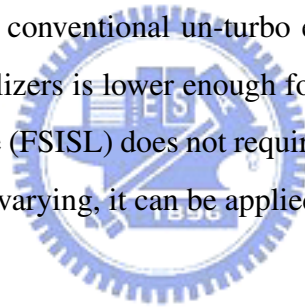
interleaver, and the error correction code. To pursue better performance, we can use a larger interleaver with a larger S-parameter, a better error correction code, or a precoding technique. However, this is out of the scope of this dissertation. Discussion regarding this issue may refer to [46, 49, 76, 77] for details. Other advance coding schemes such as turbo-codes [78, 79] and LDPC codes [80–83] may also help.

## § 5.5 Conclusions

It has been a decade since the turbo decoding scheme first proposed. Its excellent performance enables us to push the transmission toward the Shannon limit. Based on the same turbo prin-

inciple, the turbo equalization was developed to overcome the channel effect. While the performance is significantly improved, the computational complexity is increased dramatically also. Although many works has been devoted to reduce the complexity, the turbo equalizer remains implementable only for short (or sparse) wireless or magnetic channels. For channels with hundreds of taps like DSL, no feasible solutions have been reported yet.

In this dissertation, we propose fast turbo equalizers with implementable complexity. We exploit special properties of the wireline channels and develop low-complexity interpolated structures. For the SHDSL application, we have shown that the complexity of proposed turbo equalizers is an order of magnitude less than the conventional turbo equalizers. Compare to a conventional un-turbo equalizer, we can obtain a 8.8 dB performance gain at BER  $10^{-5}$  (with four iterations). With our doubly interpolated scheme (FDISL), the total computational cost is only three times higher than a conventional un-turbo equalizer. This indicates that the complexity of proposed turbo equalizers is lower enough for real-world implementation. Note that our singly interpolation scheme (FSISL) does not require the channel response having a smooth shape. If the channel is slowly varying, it can be applied to wireless channels too.





# Chapter 6

## Conclusions and Future Works

Since 1948, the Shannon's capacity limit [84] becomes the ultimate goal of every communication engineer and researcher to work with. In 1993, turbo decoding [79] was developed and for the first time the limit was approached as close as 0.7 dB. Since then, the turbo principle is extensively studied and applied to many other areas including turbo equalization. The price behind the success is the high computational complexity. In the real world, there is always a tradeoff between performance and cost. For such reason, turbo related schemes have not found wide applications yet. A conventional communication system usually use a suboptimal design. This is specially true for the wireline system since the channel response is usually very long. In this case, even conventional approaches will suffer from the high computational complexity problem.

This dissertation investigates the complexity reduction issue in wireline communication systems (xDSL). Specifically, we focus on three subjects, echo cancellation, channel equalization, and precoding. For equalization, we study both the conventional DFE and the turbo equalization. The main theme of our methods is to explore the redundancy operations found in conventional approaches. With the interpolation framework, we are able to effectively reduce the computational complexity.

For echo cancellation, we propose an optimal IFIR echo canceller with low-complexity. The

IFIR echo canceller inherits all the numerical stability advantages of the conventional FIR filter while effectively reducing its computational complexity. Optimal interpolation filters for DSL applications are designed using a least-squares method. The theoretical performance of the IFIR echo canceller was derived and the convergence behavior of its adaptive implementation was also analyzed. We have shown that the computational complexity reduction can be as high as 57% in the SHDSL application. Finally, extensive simulations using standard test loops were conducted to demonstrate the effectiveness of the optimal IFIR echo canceller.

For decision feedback equalization and precoding, we propose a low-complexity adaptive IDFE. Similar to the IFIR echo canceller, the IDFE inherits all the numerical stability advantages of the conventional FIR filter while effectively reducing its computational complexity. We have shown that the computational complexity reduction can be as high as 76%. The theoretical performance of the IDFE was derived and the convergence behavior of adaptive IDFE was also analyzed. The IDFE approach was extended to the ITHP yielding a high performance yet very efficient equalization scheme. Simulations shown that the proposed IDFE/ITHP can have the same performance as the conventional DFE/THP. Also, derived theoretical expressions predicting the IDFE performance are accurate.

As to the turbo equalizer, we exploit special properties of the wireline channels and develop two low-complexity interpolated algorithms. These algorithms effectively reduce the computational complexity of the filter-based turbo equalizer. For the SHDSL application, the complexity saving can even up to an order of magnitude. The complexity is almost as low as a conventional un-turbo equalizer. The real-world applicability of turbo equalizer is greatly enhanced.

Based on results in this dissertation, some possible future works are in order. As mentioned in Chapter 5, the total complexity of a turbo equalizer is dominated by the SISO equalizer instead of SISO decoder. Also, a better error code is required for a higher performance target. One of the best codes ever known is the turbo code. Then, it is a straightforward though to include the turbo code into the turbo equalization. Note that the complexity of SISO equalizer may be higher than that of the turbo decoder. Also, the performance of the SISO equalizer

heavily depends on that of the decoder. We may then use less iterations for the equalizer and more for the turbo decoder. The overall complexity can be lowered and the performance can be improved.

It has been shown that the turbo equalizer with precoding is able to improve the performance [76, 77], The precoding structure used here is not the same as the conventional THP. The conventional THP use a modulo operator to limit the output dynamic range. The precoder in [76,77] is in an IIR form and the dynamic range is difficult to control. The THP is simple and effective and it can prevent error propagation. It may be a good subject to study how to include the THP into the turbo equalizer.

The adaptive turbo equalizer proposed in previous works [48–50] is not suitable for long channels due to slow convergence. However, it is still possible to leverage the low-complexity property of the tap weights updating scheme. For example, we may use the non-adaptive scheme to solve the optimal filter weights and they can be served as good initial tap weights. The convergence can then be greatly accelerated and this may be good for time-varying channels. In this work, the channel response and SNR are assume to be known as *a priori*. Though the DSL channel is static, the mismatched or slowly-varying effect can still occur. Joint turbo equalization and channel estimation may provide a solution [85–87]. To this end, the adaptive method may also help.

The frequency-domain turbo equalizer has been studied and applied to the SC-FDE system with a slowly fading wireless channel [53–56, 88]. Primary study shows that our interpolation scheme is applicable such that the computational complexity can be reduced. However, the performance needs to be further verified. Similarly, our scheme can be also applied to the DMT-based DSL systems , such as ADSL, ADSL2 [89], ADSL2+ [90], VDSL [91] and VDSL2 [92]. When cyclic prefix is intentionally shortened to increase the throughput [93–95], the ISI becomes more severe. In such case, the turbo equalization can improve performance even more

The key feature of turbo processing is to process signal with soft values instead of hard

ones. This soft signal characteristic is inherent in analog signal processing. Many analog turbo decoders have been VLSI implemented [96–99]. Contrast to digital implementation, analog implementation resorts to the basic semiconductor physics. Thus, the complexity (usually two order of magnitude less) and power consumption can be reduced dramatically. Since the higher number of iterations is possible (without increasing too much complexity in analog implementation), the performance is expected to be better than digital one. Though analog implementation is still suffering other problems, it deserves for further study. For the turbo equalizer, the problem becomes more involved since we have to consider other more sophisticated operations, such as matrix inversion, filter weights initialization, and filter weights updating [100–102]. As the semiconductor process is going more advanced (now is on nano-meter size), analog implementation is expected to become more and more realistic.





# Appendix A

## Derivation of the Correlation Matrices for DFE

The input vector for the feedforward filter can be expressed as

$$\begin{bmatrix} y_k \\ y_{k-1} \\ \vdots \\ y_{k-N_f+1} \end{bmatrix} = \begin{bmatrix} h_0 & h_1 & \cdots & h_{N_h-1} & 0 & \cdots & 0 \\ 0 & h_0 & h_1 & \cdots & h_{N_h-1} & 0 & \cdots \\ \vdots & & & & & & \vdots \\ 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{N_h-1} \end{bmatrix} \begin{bmatrix} x_k \\ x_{k-1} \\ \vdots \\ x_{k-N_f-N_h+2} \end{bmatrix} + \begin{bmatrix} n_k \\ n_{k-1} \\ \vdots \\ n_{k-N_f+1} \end{bmatrix}. \quad (\text{A.1})$$

Let  $\mathbf{H}$  be the channel matrix and

$$\mathbf{H} = \begin{bmatrix} h_0 & h_1 & \cdots & h_{N_h-1} & 0 & \cdots & 0 \\ 0 & h_0 & h_1 & \cdots & h_{N_h-1} & 0 & \cdots \\ \vdots & & & & & & \vdots \\ 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{N_h-1} \end{bmatrix}_{N_f \times (N_f + N_h - 1)}. \quad (\text{A.2})$$

We then have

$$\mathbf{y}_k = \mathbf{H}\mathbf{x}_k + \mathbf{n}_k \quad (\text{A.3})$$

where  $\mathbf{x}_k = [x_k, x_{k-1}, \cdots, x_{k-N_f-N_h+2}]^T$  is the transmit signal vector,  $\mathbf{y}_k = [y_k, y_{k-1}, \cdots, y_{k-N_f+1}]^T$  is the received signal vector, and  $\mathbf{n}_k = [n_k, n_{k-1}, \cdots, n_{k-N_f+1}]^T$  is the noise vector,

respectively.

$$\mathbf{y}_{(k:k-N_f+1)} = \mathbf{H}\mathbf{x}_{(k:k-N_f-N_h+2)} + \mathbf{n}_{(k:k-N_f+1)} \quad (\text{A.4})$$

Assume that  $x_k$  is white and  $n_k$  is the additive white Gaussian noise with variance  $\sigma_n^2$ . Both are also zero-mean and mutually uncorrelated to each other.

$$\begin{aligned} \mathbf{R}_{\mathbf{y}\mathbf{y}} &= E \{ \mathbf{y}_k \mathbf{y}_k^T \} \\ &= E \{ (\mathbf{H}\mathbf{x}_k + \mathbf{n}_k) (\mathbf{H}\mathbf{x}_k + \mathbf{n}_k)^T \} \\ &= E \{ \mathbf{H}\mathbf{x}_k \mathbf{x}_k^T \mathbf{H}^T + \mathbf{n}_k \mathbf{n}_k^T + \mathbf{H}\mathbf{x}_k \mathbf{n}_k^T + \mathbf{n}_k \mathbf{x}_k^T \mathbf{H}^T \} \\ &= \mathbf{H} E \{ \mathbf{x}_k \mathbf{x}_k^T \} \mathbf{H}^T + E \{ \mathbf{n}_k \mathbf{n}_k^T \} \\ &= \sigma_x^2 \mathbf{H}\mathbf{H}^T + \sigma_n^2 \mathbf{I}_{N_f \times N_f} \end{aligned} \quad (\text{A.5})$$

Also assume  $(N_f + N_h - N_b - \Delta - 2) > 0$ . In other words, the delay value, the feedback filter length, and the feedforward filter length are chosen properly so that the following matrix  $E \{ \mathbf{x}_k \hat{\mathbf{x}}_k^T \}$  exists.

$$\begin{aligned} \mathbf{R}_{\mathbf{y}\mathbf{x}} &= E \{ \mathbf{y}_k \hat{\mathbf{x}}_k^T \} \\ &= E \{ (\mathbf{H}\mathbf{x}_k + \mathbf{n}_k) \hat{\mathbf{x}}_k^T \} \\ &= E \{ \mathbf{H}\mathbf{x}_k \hat{\mathbf{x}}_k^T + \mathbf{n}_k \hat{\mathbf{x}}_k^T \} \\ &= \mathbf{H} E \{ \mathbf{x}_k \hat{\mathbf{x}}_k^T \} \\ &= \sigma_x^2 \mathbf{H} \begin{bmatrix} \mathbf{0}_{N_b \times (\Delta+1)} & \mathbf{I}_{N_b \times N_b} & \mathbf{0}_{N_b \times (N_f+N_h-N_b-\Delta-2)} \end{bmatrix}^T \end{aligned} \quad (\text{A.6})$$

where

$$\begin{aligned}
E \{ \mathbf{x}_k \hat{\mathbf{x}}_k^T \} &= E \left\{ \begin{array}{c} \left[ \begin{array}{c} x_k \\ x_{k-1} \\ \vdots \\ x_{k-N_f-N_h+2} \end{array} \right] \left[ \begin{array}{cccc} x_{k-\Delta-1} & x_{k-\Delta-2} & \cdots & x_{k-\Delta-N_b} \end{array} \right] \\ \times \\ \left[ \begin{array}{c} x_k \\ \vdots \\ x_{k-\Delta-1} \\ \vdots \\ x_{k-\Delta-N_b} \\ \vdots \\ x_{k-N_f-N_h+2} \end{array} \right] \left[ \begin{array}{cccc} x_{k-\Delta-1} & x_{k-\Delta-2} & \cdots & x_{k-\Delta-N_b} \end{array} \right] \\ \times \\ \left[ \begin{array}{c} x_k \\ \vdots \\ x_{k-\Delta-1} \\ \vdots \\ x_{k-\Delta-N_b} \\ \vdots \\ x_{k-N_f-N_h+2} \end{array} \right] \left[ \begin{array}{cccc} x_{k-\Delta-1} & x_{k-\Delta-2} & \cdots & x_{k-\Delta-N_b} \end{array} \right] \end{array} \right\} \\
&= E \left\{ \begin{array}{c} \left[ \begin{array}{c} x_k \\ x_{k-1} \\ \vdots \\ x_{k-N_f-N_h+2} \end{array} \right] \left[ \begin{array}{cccc} x_{k-\Delta-1} & x_{k-\Delta-2} & \cdots & x_{k-\Delta-N_b} \end{array} \right] \\ \times \\ \left[ \begin{array}{c} x_k \\ \vdots \\ x_{k-\Delta-1} \\ \vdots \\ x_{k-\Delta-N_b} \\ \vdots \\ x_{k-N_f-N_h+2} \end{array} \right] \left[ \begin{array}{cccc} x_{k-\Delta-1} & x_{k-\Delta-2} & \cdots & x_{k-\Delta-N_b} \end{array} \right] \\ \times \\ \left[ \begin{array}{c} x_k \\ \vdots \\ x_{k-\Delta-1} \\ \vdots \\ x_{k-\Delta-N_b} \\ \vdots \\ x_{k-N_f-N_h+2} \end{array} \right] \left[ \begin{array}{cccc} x_{k-\Delta-1} & x_{k-\Delta-2} & \cdots & x_{k-\Delta-N_b} \end{array} \right] \end{array} \right\} \\
&= \sigma_x^2 \left[ \mathbf{0}_{N_b \times (\Delta+1)} \quad \mathbf{I}_{N_b \times N_b} \quad \mathbf{0}_{N_b \times (N_f+N_h-N_b-\Delta-2)} \right]^T \quad (\text{A.7})
\end{aligned}$$

The cross-correlation vector can then be derived as

$$\begin{aligned}
\mathbf{p}_{yx} &= E \{ \mathbf{y}_k x_{k-\Delta} \} \\
&= E \{ (\mathbf{H} \mathbf{x}_k + \mathbf{n}_k) x_{k-\Delta} \} \\
&= E \{ \mathbf{H} \mathbf{x}_k x_{k-\Delta} + \mathbf{n}_k x_{k-\Delta} \} \\
&= \mathbf{H} E \{ \mathbf{x}_k x_{k-\Delta} \} \\
&= \sigma_x^2 \mathbf{H} \left[ \mathbf{0}_{1 \times \Delta} \quad 1 \quad \mathbf{0}_{1 \times (N_f+N_h-\Delta-2)} \right]^T. \quad (\text{A.8})
\end{aligned}$$



## Appendix B

# Derivation of the Correlation Matrices for IDFE

The parameters are defined the same as those in Appendix A. Thus,  $\mathbf{R}_{yy} = E \{ \mathbf{y}_k \mathbf{y}_k^T \}$  and  $\mathbf{p}_{yx} = E \{ \mathbf{y}_k x_{k-\Delta} \}$  are the same as (A.5) and (A.8), respectively. Note that the interpolation filter  $\mathbf{g} = [g_{-(M-1)}, g_{-(M-2)}, \dots, g_{(M-1)}]^T$  is defined as non-causal. With this definition, we can obtain concise expressions later. However, the interpolation filter output is still causal as shown below.

$$\tilde{x}_{k-\Delta-(\alpha+1)} = \sum_{i=-(M-1)}^{(M-1)} g_i \hat{x}_{k-\Delta-\alpha-M-i}. \quad (\text{B.1})$$

Without loss of generality, we assume that the interpolated feedback filter length of the IDFE is equal to the feedback filter length of the DFE, i.e.  $N_b = N_g + (N_{b1} - 1)M = (N_{b1} + 1)M - 1$ , where  $N_g = 2M - 1$  is the length of the interpolation filter. Also, assume all the decisions are correct, i.e.,  $\hat{x}_{k-\Delta} = x_{k-\Delta}$ .

$$\begin{bmatrix} \tilde{x}_{k-\Delta-(\alpha+1)} \\ \tilde{x}_{k-\Delta-(\alpha+2)} \\ \vdots \\ \tilde{x}_{k-\Delta-(\alpha+1)-(N_{b1}-1)M} \end{bmatrix} = \begin{bmatrix} g_{-M+1} & g_{-M+2} & \cdots & g_{M-1} & 0 & \cdots & 0 \\ 0 & g_{-M+1} & g_{-M+2} & \cdots & g_{M-1} & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & g_{-M+1} & g_{-M+2} & \cdots & g_{M-1} \end{bmatrix} \begin{bmatrix} x_{k-\Delta-(\alpha+1)} \\ x_{k-\Delta-(\alpha+2)} \\ \vdots \\ x_{k-\Delta-N_b} \end{bmatrix} \quad (\text{B.2})$$

Let

$$\tilde{\mathbf{x}}_{1,k} = [\tilde{x}_{k-\Delta-(\alpha+1)}, \tilde{x}_{k-\Delta-(\alpha+2)}, \dots, \tilde{x}_{k-\Delta-(\alpha+1)-(N_{b1}-1)M}]^T, \quad (\text{B.3})$$

$$\hat{\mathbf{x}}_{1,k} = [x_{k-\Delta-(\alpha+1)}, x_{k-\Delta-(\alpha+2)}, \dots, x_{k-\Delta-N_b}]^T, \quad (\text{B.4})$$

and

$$\mathbf{G} = \begin{bmatrix} g_{-M+1} & g_{-M+2} & \cdots & g_{M-1} & 0 & \cdots & 0 \\ 0 & g_{-M+1} & g_{-M+2} & \cdots & g_{M-1} & 0 & \cdots \\ \vdots & & & & & & \vdots \\ 0 & \cdots & 0 & g_{-M+1} & g_{-M+2} & \cdots & g_{M-1} \end{bmatrix}. \quad (\text{B.5})$$

We then have

$$\tilde{\mathbf{x}}_{1,k} = \mathbf{G}\hat{\mathbf{x}}_{1,k}. \quad (\text{B.6})$$

Let  $\mathbf{x}_{1,k}$  be the  $M$ -downsampled version of  $\tilde{\mathbf{x}}_{1,k}$  and its first sample is  $\tilde{x}_{k-\Delta-\alpha-1}$ ,

$$\mathbf{x}_{1,k} = [\tilde{x}_{k-\Delta-(\alpha+1)}, \tilde{x}_{k-\Delta-(\alpha+1)-M}, \dots, \tilde{x}_{k-\Delta-(\alpha+1)-(N_{b1}-1)M}]^T. \quad (\text{B.7})$$

Then,

$$\begin{bmatrix} \tilde{x}_{k-\Delta-(\alpha+1)} \\ \tilde{x}_{k-\Delta-(\alpha+1)-M} \\ \vdots \\ \tilde{x}_{k-\Delta-(\alpha+1)-(N_{b1}-1)M} \end{bmatrix} = \begin{bmatrix} \underbrace{1, 0, \dots, 0}_{(N_{b1}-1)M} \\ \underbrace{0, \dots, 0, 1, 0, \dots, 0}_M \quad \underbrace{0, \dots, 0}_{(N_{b1}-2)M} \\ \vdots \\ \underbrace{0, \dots, 0, 1}_{(N_{b1}-1)M} \end{bmatrix} \begin{bmatrix} \tilde{x}_{k-\Delta-(\alpha+1)} \\ \tilde{x}_{k-\Delta-(\alpha+2)} \\ \vdots \\ \tilde{x}_{k-\Delta-(\alpha+1)-(N_{b1}-1)M} \end{bmatrix}. \quad (\text{B.8})$$

Define

$$\mathbf{D} = \begin{bmatrix} \underbrace{1, 0, \dots, 0}_{(N_{b1}-1)M} \\ \underbrace{0, \dots, 0, 1, 0, \dots, 0}_M \quad \underbrace{0, \dots, 0}_{(N_{b1}-2)M} \\ \vdots \\ \underbrace{0, \dots, 0, 1}_{(N_{b1}-1)M} \end{bmatrix}. \quad (\text{B.9})$$

Then,

$$\mathbf{x}_{1,k} = \mathbf{D}\tilde{\mathbf{x}}_{1,k} \quad (\text{B.10})$$

$$\mathbf{x}_{1,k} = \mathbf{D}\mathbf{G}\hat{\mathbf{x}}_{1,k} \quad (\text{B.11})$$

$$\begin{bmatrix} \tilde{x}_{k-\Delta-(\alpha+1)} \\ \tilde{x}_{k-\Delta-(\alpha+1)-M} \\ \vdots \\ \tilde{x}_{k-\Delta-(\alpha+1)-(N_{b_1-1})M} \end{bmatrix} = \begin{bmatrix} 1, \underbrace{0, \dots, 0}_{(N_{b_1-1})M} \\ \underbrace{0, \dots, 0}_M, 1, \underbrace{0, \dots, 0}_{(N_{b_1-2})M} \\ \vdots \\ 0, \underbrace{\dots, 0, 1}_{(N_{b_1-1})M} \end{bmatrix} \cdot \begin{bmatrix} g_{-M+1} & g_{-M+2} & \cdots & g_{M-1} & 0 & \cdots & 0 \\ 0 & g_{-M+1} & g_{-M+2} & \cdots & g_{M-1} & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & g_{-M+1} & g_{-M+2} & \cdots & g_{M-1} \end{bmatrix} \cdot \begin{bmatrix} x_{k-\Delta-(\alpha+1)} \\ x_{k-\Delta-(\alpha+2)} \\ \vdots \\ x_{k-\Delta-N_b} \end{bmatrix} \quad (\text{B.12})$$

$$\begin{bmatrix} \tilde{x}_{k-\Delta-(\alpha+1)} \\ \tilde{x}_{k-\Delta-(\alpha+1)-M} \\ \vdots \\ \tilde{x}_{k-\Delta-(\alpha+1)-(N_{b_1-1})M} \end{bmatrix} = \begin{bmatrix} \mathbf{g}^T, \underbrace{0, \dots, 0}_{(N_{b_1-1})M} \\ \underbrace{0, \dots, 0}_M, \mathbf{g}^T, \underbrace{0, \dots, 0}_{(N_{b_1-2})M} \\ \vdots \\ \underbrace{0, \dots, 0, \mathbf{g}^T}_{(N_{b_1-1})M} \end{bmatrix} \cdot \begin{bmatrix} x_{k-\Delta-(\alpha+1)} \\ x_{k-\Delta-(\alpha+2)} \\ \vdots \\ x_{k-\Delta-N_b} \end{bmatrix} \quad (\text{B.13})$$

Let  $\mathbf{M} = \mathbf{D}\mathbf{G}$ . From (B.5), (B.9), and (B.11).

$$\mathbf{M} = \begin{bmatrix} \mathbf{g}^T, \underbrace{0, \dots, 0}_{(N_{b1}-1)M} \\ \underbrace{0, \dots, 0}_M, \mathbf{g}^T, \underbrace{0, \dots, 0}_{(N_{b1}-2)M} \\ \vdots \\ \underbrace{0, \dots, 0}_{(N_{b1}-1)M}, \mathbf{g}^T \end{bmatrix}, \quad (\text{B.14})$$

$$\mathbf{x}_{1,k} = \mathbf{M}\hat{\mathbf{x}}_{1,k}. \quad (\text{B.15})$$

Then, we can have  $\mathbf{R}_{\mathbf{x}_1\mathbf{x}_1} = E\{\mathbf{x}_{1,k}\mathbf{x}_{1,k}^T\}$  as

$$\begin{aligned} \mathbf{R}_{\mathbf{x}_1\mathbf{x}_1} &= E\{\mathbf{x}_{1,k}\mathbf{x}_{1,k}^T\} \\ &= E\{\mathbf{M}\hat{\mathbf{x}}_{1,k}\hat{\mathbf{x}}_{1,k}^T\mathbf{M}^T\} \\ &= \mathbf{M}E\{\hat{\mathbf{x}}_{1,k}\hat{\mathbf{x}}_{1,k}^T\}\mathbf{M}^T \\ &= \sigma_x^2\mathbf{M}\mathbf{M}^T, \end{aligned} \quad (\text{B.16})$$

$\mathbf{R}_{\mathbf{y}\mathbf{x}_1} = E\{\mathbf{y}_k\mathbf{x}_{1,k}^T\}$  as

$$\begin{aligned} \mathbf{R}_{\mathbf{y}\mathbf{x}_1} &= E\{\mathbf{y}_k\mathbf{x}_{1,k}^T\} \\ &= E\{(\mathbf{H}\mathbf{x}_k + \mathbf{n}_k)\hat{\mathbf{x}}_{1,k}^T\mathbf{M}^T\} \\ &= E\{\mathbf{H}\mathbf{x}_k\hat{\mathbf{x}}_{1,k}^T\mathbf{M}^T + \mathbf{n}_k\hat{\mathbf{x}}_{1,k}^T\mathbf{M}^T\} \\ &= \mathbf{H}E\{\mathbf{x}_k\hat{\mathbf{x}}_{1,k}^T\}\mathbf{M}^T \\ &= \sigma_x^2\mathbf{H} \begin{bmatrix} \mathbf{0}_{(N_b-\alpha)\times(\Delta+1)} & \mathbf{I}_{(N_b-\alpha)\times(N_b-\alpha)} & \mathbf{0}_{(N_b-\alpha)\times(N_f+N_h-(N_b-\alpha)-\Delta-2)} \end{bmatrix}^T \mathbf{M}^T, \end{aligned} \quad (\text{B.17})$$



$\mathbf{R}_{\mathbf{y}\mathbf{x}_2} = E \{ \mathbf{y}_k \mathbf{x}_{2,k}^T \}$  as

$$\begin{aligned}
\mathbf{R}_{\mathbf{y}\mathbf{x}_2} &= E \{ \mathbf{y}_k \mathbf{x}_{2,k}^T \} \\
&= E \{ (\mathbf{H}\mathbf{x}_k + \mathbf{n}_k) \mathbf{x}_{2,k}^T \} \\
&= E \{ \mathbf{H}\mathbf{x}_k \mathbf{x}_{2,k}^T + \mathbf{n}_k \mathbf{x}_{2,k}^T \} \\
&= \mathbf{H} E \{ \mathbf{x}_k \mathbf{x}_{2,k}^T \} \\
&= \sigma_x^2 \mathbf{H} \begin{bmatrix} \mathbf{0}_{N_{b2} \times (\Delta+1)} & \mathbf{I}_{N_{b2} \times N_{b2}} & \mathbf{0}_{N_{b2} \times (N_f + N_h - N_{b2} - \Delta - 2)} \end{bmatrix}^T, \tag{B.18}
\end{aligned}$$

$\mathbf{R}_{\mathbf{x}_1\mathbf{x}_2} = E \{ \mathbf{x}_{1,k} \mathbf{x}_{2,k}^T \}$  as

$$\begin{aligned}
\mathbf{R}_{\mathbf{x}_1\mathbf{x}_2} &= E \{ \mathbf{x}_{1,k} \mathbf{x}_{2,k}^T \} \\
&= E \{ \mathbf{M} \hat{\mathbf{x}}_{1,k} \mathbf{x}_{2,k}^T \} \\
&= \mathbf{M} E \{ \hat{\mathbf{x}}_{1,k} \mathbf{x}_{2,k}^T \} \\
&= \sigma_x^2 \mathbf{M} \begin{bmatrix} \mathbf{0}_{(M-1) \times \alpha} & \mathbf{I}_{(M-1) \times (M-1)} \\ \mathbf{0}_{(N_b - M + 1) \times \alpha} & \mathbf{0}_{(N_b - M + 1) \times (M-1)} \end{bmatrix}, \tag{B.19}
\end{aligned}$$

and  $\mathbf{R}_{\mathbf{x}_2\mathbf{x}_2} = E \{ \mathbf{x}_{2,k} \mathbf{x}_{2,k}^T \}$  as

$$\begin{aligned}
\mathbf{R}_{\mathbf{x}_2\mathbf{x}_2} &= E \{ \mathbf{x}_{2,k} \mathbf{x}_{2,k}^T \} \\
&= \sigma_x^2 \mathbf{I}_{N_{b2} \times N_{b2}}. \tag{B.20}
\end{aligned}$$



## Appendix C

# Derivation of the BCJR Algorithm for Optimal Equalization

The original BCJR algorithm was derived [61] for decoding. However, the algorithm derived here is for the equalization purpose. Both probabilistic and logarithmic forms will be derived in detail. Some of the derivation closely follows the lecture notes in [103].

Consider a discrete-equivalent ISI channel with real-valued response denoted as  $\mathbf{h} = [h_0, h_1, \dots, h_\mu]^T$ , where  $\mu$  is the channel *memory* length. The received signal  $y_k$  corrupted by additive noise can be described as follows:

$$y_k = \sum_{i=0}^{\mu} h_i x_{k-i} + n_k \quad (\text{C.1})$$

where  $x_k$ , with a unit variance, denotes the transmit signal and  $n_k$ , with a variance  $\sigma_n^2$ , denotes the additive white Gaussian noise (AWGN). Both  $x_k$  and  $n_k$  are also assume to have zero means and to be independent each other.

Suppose that the transmitter sends a block of  $K$  data symbols  $\mathbf{x} = [x_0, x_1, \dots, x_{K-1}]^T$ , drawn independently and uniformly from a discrete signal constellation set  $X$ . We assume that one of the symbols in  $X$  is identified as the *idle* symbol, and that idle symbols are transmitted outside the transmit block (i.e., time  $k < 0$  and  $k > K - 1$ ). A block of received signals is

collected to form a vector  $\mathbf{y} = [y_0, y_1, \dots, y_{K+\mu-1}]^T$ . The received signal  $y_k$  outside the time interval  $k < 0$  and  $k \geq K + \mu$  are independent of  $\mathbf{x}$  and can be ignored since they correspond to *idle* symbols.

The relationship of the channel input and output can be represented with a trellis diagram. Let  $[x_{k-1}, x_{k-2}, \dots, x_{k-\mu}]$  denote the state at time  $k$ . Thus, there are  $Q = M^\mu$  possible states and these states are labelled from state 0 through state  $Q - 1$ , with state 0 reserved for the idle state. Usually, the idle state is defined as the state with all mapping bits the convolutional encoder are equals to all zeros. Let  $s_k$  be the state label in time instant  $k$ . Fig. 5.3 defines the state transition diagram. One generic state transition (or branch), denoted as a two-tuple  $(p, q)$ , is a connection between these states from the state  $p$  at time  $k$  to state  $q$  at time  $k + 1$ . Meanwhile,  $x_{(p,q)} / v_{(p,q)}$  ( $x_k = x_{(p,q)} \in X$ ,  $v_k = \sum_{i=0}^{\mu} h_i x_{k-i}$ ) denotes the associated input/output pair. The signal  $v_k$  is the  $k$ -th channel output signal (without noise).

The optimal equalizer for the ISI channel is an equalizer that minimizes the probability of error for each symbol decision or equivalently maximizes *a posteriori* probability (APP)  $\Pr(x_k | \mathbf{y})$ . For example, the equalizer calculates the *a posteriori* probability  $\Pr(x_0 | \mathbf{y})$  for each possible  $x_0 \in X$ , and make a decision maximizing  $\Pr(x_0 | \mathbf{y})$ . This process is then repeated for all  $K$  transmitted symbols. The APP for the  $k$ -th symbol  $x_k$  is related to the *a posteriori* transition probabilities by

$$\Pr(x_k = \chi | \mathbf{y}) = \sum_{(p,q) \in B_\chi} \Pr(s_k = p; s_{k+1} = q | \mathbf{y}), \quad (\text{C.2})$$

where  $B_\chi$  denote the set of integer pairs  $(p, q)$  for which a state transition from  $p$  to  $q$  corresponds to an input symbol of  $\chi$ ,  $\chi \in X$ . In the context of the trellis diagram, the APPs can be easily computed once the *a posteriori* state transition probabilities  $\Pr(s_k = p; s_{k+1} = q | \mathbf{y})$  are known for each state transition (or branch). The BCJR algorithm provides a computationally efficient method for finding these state transition probabilities.

The key of the BCJR algorithm is to decompose the *a posteriori* transition probability for a transition at time  $k$  into three separable factors: the first one depending only on the past

observations  $\mathbf{y}_{i < k} = \{y_i : i < k\}$ , the second one depending on only the present observation  $y_k$ , and the third one depending only on the future observations  $\mathbf{y}_{i > k} = \{y_i : i > k\}$ . Using Bayes' rule and chain rule [104], we can accomplish this through the following straightforward substitutions.

$$\begin{aligned}
 \Pr(s_k = p; s_{k+1} = q | \mathbf{y}) &= p(s_k = p; s_{k+1} = q; \mathbf{y}) / p(\mathbf{y}) \\
 &= p(s_k = p; s_{k+1} = q; \mathbf{y}_{i < k}; y_k; \mathbf{y}_{i > k}) / p(\mathbf{y}) \\
 &= p(\mathbf{y}_{i > k} | s_k = p; s_{k+1} = q; \mathbf{y}_{i < k}; y_k) p(s_k = p; s_{k+1} = q; \mathbf{y}_{i < k}; y_k) \cdot \\
 &\quad 1/p(\mathbf{y}), \tag{C.3}
 \end{aligned}$$

where  $p(\cdot)$  stands for the PDF. Because of the Markov property of the finite-state machine channel model, knowledge of the state at time  $k + 1$  supersedes knowledge of the state at time  $k$ , and it also supersedes knowledge of  $y_k$  and  $\mathbf{y}_{i < k}$ . So, (C.3) is reduced to:

$$\begin{aligned}
 \Pr(s_k = p; s_{k+1} = q | \mathbf{y}) &= p(\mathbf{y}_{i > k} | s_{k+1} = q) p(s_k = p; s_{k+1} = q; \mathbf{y}_{i < k}; y_k) / p(\mathbf{y}) \\
 &= p(\mathbf{y}_{i > k} | s_{k+1} = q) p(s_{k+1} = q; y_k | s_k = p; \mathbf{y}_{i < k}) p(s_k = p; \mathbf{y}_{i < k}) \cdot \\
 &\quad 1/p(\mathbf{y}). \tag{C.4}
 \end{aligned}$$

Exploiting the Markov property again, we can simplify (C.4) to

$$\begin{aligned}
 \Pr(s_k = p; s_{k+1} = q | \mathbf{y}) &= \underbrace{p(\mathbf{y}_{i > k} | s_{k+1} = q)}_{\alpha_k(p)} \underbrace{p(s_{k+1} = q; y_k | s_k = p)}_{\gamma_k(p, q)} \underbrace{p(s_k = p; \mathbf{y}_{i < k})}_{\beta_{k+1}(q)} \cdot 1/p(\mathbf{y}) \\
 &= \alpha_k(p) \cdot \gamma_k(p, q) \cdot \beta_{k+1}(q) / p(\mathbf{y}), \tag{C.5}
 \end{aligned}$$

where  $\alpha_k(p)$  is a probability (density) measure for state  $p$  at time  $k$  that depends only on the past observations  $\mathbf{y}_{i < k}$ ,  $\beta_{k+1}(q)$  is a probability measure for state  $q$  at time  $k + 1$  that depends only on the future observations  $\mathbf{y}_{i > k}$ , and  $\gamma_k(p, q)$  is a probability measure connecting state transition from  $p$  at time  $k$  to state  $q$  at time  $k + 1$  and it depends only on the present ( $k$ -th) observation

$y_k$ . From (C.5), the APP now is given by

$$\begin{aligned}\Pr(x_k = \chi | \mathbf{y}) &= \sum_{(p,q) \in B_X} \Pr(s_k = p; s_{k+1} = q | \mathbf{y}) \\ &= \frac{1}{p(\mathbf{y})} \sum_{(p,q) \in B_X} \alpha_k(p) \cdot \gamma_k(p, q) \cdot \beta_{k+1}(q).\end{aligned}\quad (\text{C.6})$$

The  $\gamma_k(p, q)$  named as branch metric is the fundamental probability measure for the BCJR algorithm. Once the branch metric is calculated, other probability measures  $\alpha_k(p)$  and  $\beta_{k+1}(q)$  can be obtained easily. The BCJR algorithm makes a forward pass and a backward pass to obtain these probabilities and we will describe this later. The branch metric is further decomposed as

$$\begin{aligned}\gamma_k(p, q) &= p(s_{k+1} = q; y_k | s_k = p) \\ &= p(y_k | s_k = p; s_{k+1} = q) \Pr(s_{k+1} = q | s_k = p).\end{aligned}\quad (\text{C.7})$$

For the channel under consideration, the first term in (C.7) equals

$$p(y_k | s_k = p; s_{k+1} = q) = \frac{1}{2\pi\sigma_n^2} \exp\left\{-\frac{|y_k - v_{(p,q)}|^2}{2\sigma_n^2}\right\}.\quad (\text{C.8})$$

On the other hand, the second term in (C.7) equals

$$\begin{aligned}\Pr(s_{k+1} = q | s_k = p) &= \Pr(x_k = x_{(p,q)}; s_k = p) / \Pr(s_k = p) \\ &= \Pr(s_k = p | x_k = x_{(p,q)}) \Pr(x_k = x_{(p,q)}) / \Pr(s_k = p) \\ &= \Pr(s_k = p) \Pr(x_k = x_{(p,q)}) / \Pr(s_k = p) \\ &= \Pr(x_k = x_{(p,q)}).\end{aligned}\quad (\text{C.9})$$

Note that (C.9) is just the *a priori* probability for the  $k$ -th symbol. From (C.8) and (C.9), (C.7) now is given by

$$\gamma_k(p, q) = \frac{1}{2\pi\sigma_n^2} \exp\left\{-\frac{|y_k - v_{(p,q)}|^2}{2\sigma_n^2}\right\} \Pr(x_k = x_{(p,q)}).\quad (\text{C.10})$$

Comparing the branch metric of the BCJR with the Viterbi Algorithm (VA) [40, 105], we found that there is an extra factor for the *a priori* probability  $\Pr(x_k = x_{(p,q)})$ . For some applications,

all symbols are equally likely (i.e.,  $\Pr(x_k = x_{(p,q)})$  is a constant and independent of  $x_{(p,q)}$ ). The BCJR metric is equivalent to the Viterbi metric. However, there are other applications such as channel decoding and turbo applications in which all symbols are not equally likely. Exploiting the knowledge of *a priori* probability can improve the system performance accordingly.

We now introduce some vector representation describing the BCJR more compactly. Define a  $1 \times Q$  row vector  $\alpha_k$  as the collection of  $\alpha_k(p)$  values at time  $k$  (one for each of the  $Q$  states and  $p = 0, 1, \dots, Q - 1$ ):

$$\alpha_k = [\alpha_k(0), \alpha_k(1), \dots, \alpha_k(Q - 1)]. \quad (\text{C.11})$$

Similarly, define a  $1 \times Q$  column vector  $\beta_k$  as the collection of  $\beta_k(q)$  values at time  $k$  (one for each of the  $Q$  states and  $q = 0, 1, \dots, Q - 1$ ):

$$\beta_k = [\beta_k(0), \beta_k(1), \dots, \beta_k(Q - 1)]^T. \quad (\text{C.12})$$

Finally, define a  $Q \times Q$  matrix for the  $k$ -th stage of the trellis according to  $(\Gamma_k)_{p,q} = \gamma_k(p, q)$ .

Then, the forward probabilities  $\alpha_{k+1}$  can be calculated recursively as

$$\begin{aligned} \alpha_{k+1}(q) &= p(s_{k+1} = q; \mathbf{Y}_{i < k+1}) \\ &= p(s_{k+1} = q; y_k; \mathbf{Y}_{i < k}) \\ &= \sum_{p=0}^{Q-1} p(s_{k+1} = q; y_k; s_k = p; \mathbf{Y}_{i < k}) \\ &= \sum_{p=0}^{Q-1} p(s_{k+1} = q; y_k | s_k = p; \mathbf{Y}_{i < k}) p(s_k = p; \mathbf{Y}_{i < k}) \\ &= \sum_{p=0}^{Q-1} p(s_{k+1} = q; y_k | s_k = p) p(s_k = p; \mathbf{Y}_{i < k}) \\ &= \sum_{p=0}^{Q-1} r_k(p, q) \alpha_k(p), \end{aligned} \quad (\text{C.13})$$

or equivalently, in a vector form as

$$\alpha_{k+1} = \alpha_k \Gamma_k. \quad (\text{C.14})$$

Similarly, the backward probability  $\beta_k$  is given by

$$\begin{aligned}
\beta_{k+1}(p) &= p(\mathbf{y}_{i>k-1} | s_k = p) \\
&= p(\mathbf{y}_{i>k}; y_k | s_k = p) \\
&= \sum_{q=0}^{Q-1} p(\mathbf{y}_{i>k}; y_k; s_{k+1} = q | s_k = p) \\
&= \sum_{q=0}^{Q-1} p(\mathbf{y}_{i>k} | y_k; s_{k+1} = q; s_k = p) p(y_k; s_{k+1} = q | s_k = p) \\
&= \sum_{q=0}^{Q-1} p(\mathbf{y}_{i>k} | s_{k+1} = q) p(y_k; s_{k+1} = q | s_k = p) \\
&= \sum_{q=0}^{Q-1} \beta_{k+1}(q) r_k(p, q),
\end{aligned} \tag{C.15}$$

or equivalently, in a vector form as:

$$\boldsymbol{\beta}_k = \boldsymbol{\Gamma}_k \boldsymbol{\beta}_{k+1}. \tag{C.16}$$


The BCJR algorithm in the probabilistic form can be summarized as follows.

## § C.1 The BCJR Algorithm (Probabilistic Form)

1. Calculate branch metrics:  $\gamma_k(p, q)$  for  $p = 0, 1, \dots, Q - 1, q = 0, 1, \dots, Q - 1, k = 0, 1, \dots, K + \mu - 1$ .
2. Calculate the forward/backward state probabilities:

$$\boldsymbol{\alpha}_{k+1} = \boldsymbol{\alpha}_k \boldsymbol{\Gamma}_k, \quad k = \{0, 1, \dots, K + \mu - 1\}, \tag{C.17}$$

$$\boldsymbol{\beta}_k = \boldsymbol{\Gamma}_k \boldsymbol{\beta}_{k+1}, \quad k = \{K + \mu - 1, K + \mu - 2, \dots, 0\}, \tag{C.18}$$

where the  $\boldsymbol{\alpha}_0$  and  $\boldsymbol{\beta}_{K+\mu}$  is initialized as  $[1, 0, \dots, 0]^T$  (zero state) or  $[\frac{1}{M}, \frac{1}{M}, \dots, \frac{1}{M}]^T$  (unknown state).



3. Calculate *a posteriori* probabilities (APPs):

$$\Pr(x_k = \chi | \mathbf{y}) = \frac{1}{p(\mathbf{y})} \sum_{(p,q) \in B_\chi} \alpha_k(p) \cdot \gamma_k(p, q) \cdot \beta_{k+1}(q), \quad (\text{C.19})$$

where  $B_\chi$  is the subset of state transition for input symbol  $\chi \in X$ ,  $X = \{\chi_0, \chi_1, \dots, \chi_{Q-1}\}$ . This quantity is calculated for every  $\chi$  in  $X$ , and for each input symbol  $x_0$  through  $x_{K+\mu-1}$  ( $\mathbf{x} = \{x_0, x_1, \dots, x_{K+\mu-1}\}$ ). Note that  $p(\mathbf{y})$  is a constant term and is ignorable.

4. Select the maximum APP as the estimated value (hard decision output):

$$\hat{x}_k = \max \{ \Pr(x_k = \chi | \mathbf{y}) \}. \quad (\text{C.20})$$

In practice, the quantity  $p(\mathbf{y})$  in the denominator of (C.19) can be ignored, since it is common to all APPs and will not influence the maximization procedure. Nevertheless, a simple way to calculate  $p(\mathbf{y})$  is described below. Considering the *a posteriori* joint PDF of (C.3), we have

$$\Pr(s_k = p; s_{k+1} = q | \mathbf{y}) = \alpha_k(p) \cdot \gamma_k(p, q) \cdot \beta_{k+1}(q) / p(\mathbf{y}) \quad (\text{C.21})$$

Summation of all conditional PDFs must be equal to one. i.e.,

$$\sum_{q=0}^{Q-1} \alpha_k(p) \cdot \gamma_k(p, q) \cdot \beta_{k+1}(q) / p(\mathbf{y}) = 1, \quad (\text{C.22})$$

or equivalently,

$$\begin{aligned} p(\mathbf{y}) &= \sum_{p=0}^{Q-1} \sum_{q=0}^{Q-1} \alpha_k(p) \cdot \gamma_k(p, q) \cdot \beta_{k+1}(q) \\ &= \boldsymbol{\alpha}_k \boldsymbol{\Gamma}_k \boldsymbol{\beta}_{k+1} \\ &= \boldsymbol{\alpha}_k \boldsymbol{\beta}_k. \end{aligned} \quad (\text{C.23})$$

That means  $p(\mathbf{y})$  is equal to the inner product  $\boldsymbol{\alpha}_k \boldsymbol{\beta}_k$  for any time  $k$ . Observe that (C.23) is valid for any time  $k$ , including  $k = 0$  and  $k = K + \mu - 1$ . Thus, (C.23) is reduced to a constant as

$$p(\mathbf{y}) = \beta_0(0) = \alpha_{K+\mu}(0) \quad (\text{C.24})$$

The property of (C.24) is useful when realizing a BCJR algorithm. In run-time, we can check if  $p(\mathbf{y})$  is a constant.

The above BCJR algorithm is expressed with a probabilistic form, i.e., in terms of probability and probability density function. It provides simple and closed-form expressions for describing the BCJR algorithm but it suffers from numerical *underflow* problem during simulations and implementation. For the forward state probability in (C.13), the next state probability  $\alpha_{k+1}(q)$  is the weighted sum of the current state probability  $\alpha_k(p)$  and the branch metric  $\gamma_k(p, q)$ . The current state probability is *always* smaller and equal to one and branch metric is *usually* smaller than one (for a low-to-medium SNR condition, the  $\gamma_k(p, q)$  has a low-and-wide Gaussian distribution and its probability density is usually smaller than one). Thus, the value of  $\alpha_{k+1}(q)$  becomes smaller and smaller toward the zero for every forward recursion. Finally, the underflow problem occurs. Note that for a small block length, the underflow problem is not severe. However, if the block is large as a well-performed BCJR algorithm required to achieve good performance, the problem becomes significant.

Similarly, from (C.15) it can be shown that the backward state probability also suffers from the underflow problem. To solve the underflow problem, the state probabilities can be normalized to one at every time instance. That is

$$\alpha_k = \frac{\alpha_k}{\sum_{p=0}^{Q-1} \alpha_k(p)}, \beta_k = \frac{\beta_k}{\sum_{p=0}^{Q-1} \beta_k(p)}. \quad (\text{C.25})$$

Note that the constant property in (C.23) is not hold any more if the normalization is applied.

Though a numerical stability problem is obtained, the implementation cost for the BCJR algorithm is still high due to the high-complex nonlinear exponential operation in  $\gamma_k(p, q)$  and the multiplier-dominated operation in (C.13), (C.15) and (C.19). These shortcomings can be mitigated if the algorithm is implemented in the logarithm domain.

## § C.2 The BCJR Algorithm (Logarithmic Form or Log-MAP)

1. Calculate logarithmic branch metrics:

$$\ln \gamma_k(p, q) = \frac{-|y_k - v_{(p,q)}|^2}{2\sigma_n^2} + K_\gamma, \quad (\text{C.26})$$

where  $K_\gamma = \ln(\Pr(x_k = a_{(p,q)})) - \ln(2\pi\sigma_n^2)$ , for  $p = 0, 1, \dots, Q-1, q = 0, 1, \dots, Q-1, k = 0, 1, \dots, K + \mu - 1$ .

2. Calculate the forward/backward state probabilities:

$$\ln \alpha_{k+1}(q) = \ln \sum_{p=0}^{Q-1} \exp[\ln r_k(p, q) + \ln \alpha_k(p)], \quad (\text{C.27})$$

$$\ln \beta_k(p) = \ln \sum_{q=0}^{Q-1} \exp[\ln r_k(p, q) + \ln \beta_{k+1}(q)], \quad (\text{C.28})$$

where  $\ln \alpha_0$  and  $\ln \beta_{K+\mu}$  is initialized as  $[0, -\infty, \dots, -\infty]^T$  (zero state) or  $[-\ln(M), -\ln(M), \dots, -\ln(M)]^T$  (unknown state).

3. Perform normalization to avoid numerical *overflow*:

$$\alpha_k(p) = \alpha_k(p) - \max\{\alpha_k(p)\}, \quad (\text{C.29})$$

$$\beta_k(p) = \beta_k(p) - \max\{\beta_k(p)\}. \quad (\text{C.30})$$

This step limits the maximum probability density value at one.

4. Calculate *a posteriori* probabilities (APPs):

$$\ln \Pr(x_k = \chi | \mathbf{y}) \triangleq \ln \sum_{(p,q) \in B_\chi} \exp[\ln \alpha_k(p) + \ln \gamma_k(p, q) + \ln \beta_{k+1}(q)]. \quad (\text{C.31})$$

where  $B_\chi$  is the subset of state transition for input symbol  $\chi \in X, X = \{\chi_0, \chi_1, \dots, \chi_{Q-1}\}$ . This quantity is calculated for every  $\chi$  in  $X$ , and for each input symbol  $x_0$  through  $x_{K+\mu-1}$  ( $\mathbf{x} = \{x_0, x_1, \dots, x_{K+\mu-1}\}$ ). Note that the constant term  $-\ln p(\mathbf{y})$  is omitted in (C.31) for simplicity.

5. Select the maximum APP LLR as the estimated value (hard decision output):

$$\hat{x}_k = \max \{ \ln \Pr(x_k = \chi | \mathbf{y}) \}. \quad (\text{C.32})$$

For simplicity, the exponential and logarithmic operations in (C.27), (C.28), and (C.31) are usually calculated with the Jacobian formula [106]:

$$\begin{aligned} \ln(e^{\delta_1} + e^{\delta_2}) &= \max(\delta_1, \delta_2) + \ln(1 + e^{-|\Delta|}) \\ &= \max(\delta_1, \delta_2) + f_c(\Delta) \\ &\triangleq \max^*(\delta_1, \delta_2), \end{aligned} \quad (\text{C.33})$$

where

$$f_c(\Delta) = \ln(1 + e^{-|\Delta|}) \quad (\text{C.34})$$

is a correction term and  $\Delta \triangleq \delta_2 - \delta_1$ .

Generally, the expression  $\ln(e^{\delta_1} + e^{\delta_2} + \dots + e^{\delta_n})$  is computed recursively (and exactly) as

$$\begin{aligned} \ln(e^{\delta_1} + e^{\delta_2} + \dots + e^{\delta_n}) &= \max(\delta, \delta_n) + \ln(1 + e^{-|\Delta|}) \\ &= \max(\delta, \delta_n) + f_c(\Delta) \\ &\triangleq \max^*(\delta, \delta_n), \end{aligned} \quad (\text{C.35})$$

where  $\delta = \ln(e^{\delta_1} + e^{\delta_2} + \dots + e^{\delta_{n-1}})$ . The  $\max^*(\cdot)$  is the so-called  $\max^*$  operator which stands for exact computation of  $\ln(e^{\delta_1} + e^{\delta_2})$  instead of approximation with  $\ln(e^{\delta_1} + e^{\delta_2}) \approx \max(\delta_1, \delta_2)$ . Equations (C.33) and (C.35) are very useful for LLR calculations.

In spite of the correction term in (C.34), the BCJR algorithm in logarithmic domain successfully transforms the complex multiplication operations in original BCJR algorithm into simpler addition operations. In this dissertation, both the SISO equalizer and the SISO decoder are implemented in this form. In order to implement the BCJR in a lower complexity, the correction term is further approximated by many ways, such as a constant, a linear function, or a simplified operation [107]. However, performance loss may occur. One extreme case is to omit the correction term and the algorithm is called Max-log-MAP. This algorithm is summarized below for comparison.

### § C.3 The BCJR algorithm (Max-log-MAP)

To obtain a lowest complexity, the correction term in (C.34) is omitted. Thus, the forward/backward state probabilities in (C.27), (C.28) and logarithmic APPs (C.31) are simplified as

$$\ln \alpha_{k+1}(q) = \max_{(s_k, s_{k+1}=q)} \{ \ln \gamma_k(p, q) + \ln \alpha_k(p) \}, \quad (\text{C.36})$$

where  $(s_k, s_{k+1} = q)$  stands for all possible state transitions from any state  $s_k$  at time  $k$  to state  $s_{k+1} = q$  at time  $k + 1$ .

$$\ln \beta_k(p) = \max_{(s_k=p, s_{k+1})} \{ \ln \gamma_k(p, q) + \ln \beta_{k+1}(q) \}, \quad (\text{C.37})$$

where  $(s_k = p, s_{k+1})$  stands for all possible state transitions from any state  $s_{k+1}$  at time  $k + 1$  to state  $s_k = p$  at time  $k$ .

$$\ln \Pr(x_k = \chi | \mathbf{y}) \stackrel{\Delta}{=} \max_{(p, q) \in B_\chi} \{ \ln \alpha_k(p) + \ln \gamma_k(p, q) + \ln \beta_{k+1}(q) \}. \quad (\text{C.38})$$

This algorithm simply consists of addition and comparison operations only. Since  $\max(\cdot)$  instead of the summation operation is applied, the performance degradation usually occurs.



# Appendix D

## Derivations of Soft-input and Soft-output for BPSK and 4-PAM

Assume that bits  $c_{k,j}$ 's within a codeword are independent. The *a priori* symbol probability is then given by

$$\Pr(x_k = \chi_i) = \prod_{j=1}^{\rho} \frac{1}{2} (1 + u_{i,j} \tanh(L(c_{i,j})/2)), \quad (\text{D.1})$$

where  $u_{i,j} = (-1)^{c_{k,j}}$ ,  $\rho$  is the codeword length in bits.

### § D.1 2-PAM (BPSK)

For a 2-PAM, the signal constellation equals  $\chi = \{\chi_0 = +1, \chi_1 = -1\}$ . Since  $L(x_k) \triangleq \ln \frac{\Pr(x_k=+1)}{\Pr(x_k=-1)}$ , the soft-input is given by

$$\begin{aligned} \bar{x}_k &= \sum_{i=0}^1 \chi_i \Pr(x_k = \chi_i) \\ &= \Pr(x_k = +1) - \Pr(x_k = -1) \\ &= \frac{e^{L(x_k)}}{1 + e^{L(x_k)}} - \frac{1}{1 + e^{L(x_k)}} \\ &= \tanh(L(x_k)/2). \end{aligned} \quad (\text{D.2})$$

The associated reliability information is given by

$$\begin{aligned}
v_k &\triangleq \mathbb{E}\{x_k^2\} - |\bar{x}_k|^2 \\
&= \sum_{i=0}^{M-1} |\chi_i|^2 \Pr(x_k = \chi_i) - |\bar{x}_k|^2 \\
&= 1 - |\bar{x}_k|^2.
\end{aligned} \tag{D.3}$$

Note that the last equality in (D.3) is also valid for M-PSK with a unit-power normalization but may not valid for M-PAM and M-QAM when  $M \geq 2$ .

Given an optimal filter  $\mathbf{f}_k$ , we have

$$\rho_{k,i} \triangleq \frac{|\hat{x}_k - \chi_i \cdot \mathbf{f}_k^T \mathbf{s}|^2}{\mathbf{f}_k^T \mathbf{s} (1 - \mathbf{f}_k^T \mathbf{s})}. \tag{D.4}$$

The extrinsic information for 2-PAM is given by

$$\begin{aligned}
L_e(c_{k,1}) &= \ln \frac{e^{\left(\frac{-\rho_0}{2}\right)}}{e^{\left(\frac{-\rho_1}{2}\right)}} \\
&= \frac{-\rho_0 + \rho_1}{2} \\
&= -\frac{|\hat{x}_k - \mathbf{f}_k^T \mathbf{s}|^2}{2\mathbf{f}_k^T \mathbf{s} (1 - \mathbf{f}_k^T \mathbf{s})} + \frac{|\hat{x}_k + \mathbf{f}_k^T \mathbf{s}|^2}{2\mathbf{f}_k^T \mathbf{s} (1 - \mathbf{f}_k^T \mathbf{s})} \\
&= \frac{2\hat{x}_k}{(1 - \mathbf{f}_k^T \mathbf{s})}.
\end{aligned} \tag{D.5}$$

## § D.2 4-PAM

For 4-PAM ( $\rho = 2$ ), define  $L_1 \triangleq \tanh(L(c_{i,1})/2)$ ,  $L_2 \triangleq \tanh(L(c_{i,2})/2)$ . The related parameters for calculating the soft-input are listed in Table D.1.

Thus, the soft-input for 4-PAM is given by

$$\begin{aligned}
\bar{x}_k &= \sum_{i=0}^{M-1} \chi_i \Pr(x_k = \chi_i) \\
&= \frac{2L_1 + L_2}{\sqrt{5}}.
\end{aligned} \tag{D.6}$$



Table D.1: Parameters for calculating the soft-input (4-PAM)

| Constellation<br>( $x_k = \chi_i$ ) | Informationbits<br>$[c_{i,1}, c_{i,2}]$ | Sign<br>$[u_{i,1}, u_{i,2}]$ | Probabilities<br>$\Pr(x_k = \chi_i)$       |
|-------------------------------------|---|------------------------------|--|
| $+3/\sqrt{5}$                       | 00                                      | ++                           | $\frac{1}{2}(1 + L_1)\frac{1}{2}(1 + L_2)$ |
| $+1/\sqrt{5}$                       | 01                                      | +−                           | $\frac{1}{2}(1 + L_1)\frac{1}{2}(1 - L_2)$ |
| $-1/\sqrt{5}$                       | 10                                      | −+                           | $\frac{1}{2}(1 - L_1)\frac{1}{2}(1 + L_2)$ |
| $-3/\sqrt{5}$                       | 11                                      | −−                           | $\frac{1}{2}(1 - L_1)\frac{1}{2}(1 - L_2)$ |

For given LLRs, we can calculate the average power of  $x_k$  as

$$\begin{aligned} \mathbb{E}\{x_k^2\} &= \sum_{i=0}^{M-1} |\chi_i|^2 \Pr(x_k = \chi_i) \\ &= \frac{5 + 4L_1L_2}{5} \end{aligned} \quad (\text{D.7})$$

From (D.7) and (D.6), the associated reliability information equals

$$\begin{aligned} v_k &\triangleq \mathbb{E}\{x_k^2\} - |\bar{x}_k|^2 \\ &= \left(\frac{5 + 4L_1L_2}{5}\right) - \left(\frac{2L_1 + L_2}{K}\right)^2 \\ &= 1 - \frac{4L_1^2 + L_2^2}{5}. \end{aligned} \quad (\text{D.8})$$

Given an optimal filter  $\mathbf{f}_k$ , we have

$$\begin{aligned} \rho_{k,i} &\triangleq \frac{|\hat{x}_k - \mu_{k,i}|^2}{\mathbf{f}_k^T \mathbf{s} (1 - \mathbf{f}_k^T \mathbf{s})} \\ &= \frac{|\hat{x}_k - \chi_i \cdot \mathbf{f}_k^T \mathbf{s}|^2}{\mathbf{f}_k^T \mathbf{s} (1 - \mathbf{f}_k^T \mathbf{s})} \\ &= \frac{|\hat{x}_k - \chi_i \kappa|^2}{\kappa'}, \end{aligned} \quad (\text{D.9})$$

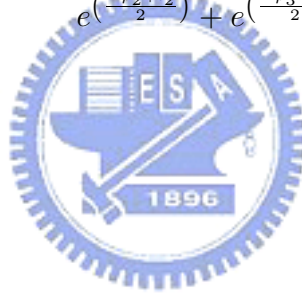
where  $\kappa \triangleq \mathbf{f}_k^T \mathbf{s}$  and  $\kappa' \triangleq \kappa(1 - \kappa)$  are constants. The related parameters for calculating the extrinsic information for 4-PAM are listed in Table D.2. Define  $l_j = L(c_{k,j})$ , the extrinsic

Table D.2: Parameters for calculating the extrinsic information (4-PAM)

| Constellation<br>( $x_k = \chi_i$ ) | Informationbits<br>$[c_{i,1}, c_{i,2}]$ | Sign<br>$[u_{i,1}, u_{i,2}]$ | $\mu_i \triangleq \mu_{k,i}$         | $\rho_i \triangleq \rho_{k,i}$                            |
|-------------------------------------|---|------------------------------|--------------------------------------|---|
| $+3/\sqrt{5}$                       | 00                                      | ++                           | $\mu_0 \triangleq 3\kappa/\sqrt{5}$  | $\rho_0 \triangleq \frac{ \hat{x}_k - \mu_0 ^2}{\kappa'}$ |
| $+1/\sqrt{5}$                       | 01                                      | +-                           | $\mu_1 \triangleq +\kappa/\sqrt{5}$  | $\rho_1 \triangleq \frac{ \hat{x}_k - \mu_1 ^2}{\kappa'}$ |
| $-1/\sqrt{5}$                       | 10                                      | -+                           | $\mu_2 \triangleq -\kappa/\sqrt{5}$  | $\rho_2 \triangleq \frac{ \hat{x}_k - \mu_2 ^2}{\kappa'}$ |
| $-3/\sqrt{5}$                       | 11                                      | --                           | $\mu_3 \triangleq -3\kappa/\sqrt{5}$ | $\rho_3 \triangleq \frac{ \hat{x}_k - \mu_3 ^2}{\kappa'}$ |

information for 4-PAM, is given by

$$L_e(c_{k,1}) = \ln \frac{e^{\left(\frac{-\rho_0 + l_2}{2}\right)} + e^{\left(\frac{-\rho_1 - l_2}{2}\right)}}{e^{\left(\frac{-\rho_2 + l_2}{2}\right)} + e^{\left(\frac{-\rho_3 - l_2}{2}\right)}}. \quad (\text{D.10})$$



# Appendix E

## Complexity Analysis

In this appendix, we analyze the complexity for all kind of equalizers in detail. Since a complete turbo equalizer includes a decoder, we also analyze the complexity of the BCJR decoder. The BCJR algorithm considered here is implemented with a logarithmic form.

Assume that the channel is time-invariant. Denote the system parameters as: the channel memory  $\mu = L-1$  (channel length  $L$ ), the filter length  $N$ , (total tap weights length of feedforward and feedback filter), the interpolation complexity ratio  $R$ , the block length  $K \gg N$ , the total number of iteration  $P$ . The complexity is in terms of the number of multiplication ( $\times$ ), addition ( $+$ ), division ( $\div$ ) operations. The nonlinear function  $\tanh(\cdot)$  used in the soft-input converter is implemented by 512 entries of lookup table [55, pp. 124]. The table lookup is implemented by a binary search; and the complexity of a comparison operation used in binary search is assumed equivalently to one addition operation. Thus, the complexity of table lookup is included in addition operations.

The detail complexity of the proposed FDISL equalizer is tabulated in Table E.1. Note that the complexity for calculating  $v_k$  and  $\mathbf{y}_n - \bar{\mathbf{y}}_n$  is ignored. The optimal filter is implemented as

$$\tilde{\mathbf{f}}_i = w\mathbf{f}_1 + (1 - w)\mathbf{f}_2 = w(\mathbf{f}_1 - \mathbf{f}_2) + \mathbf{f}_2, \quad (\text{E.1})$$

in which the complexity equals one multiplication and two additions per tap. Since the block size is greater than the filter length, i.e.,  $N/K \ll 1$ , the complexity for computing the optimal

filter tap weights reduces to only one table lookup and one division operations (which is used to find the weighting factor  $w$ ). This is the major reason why the proposed algorithms can reduce the complexity dramatically. There is no need to update the channel response  $\mathbf{h}$  since the

Table E.1: Complexity of FDISL equalizer

| Equalizer type              | FDISL                    |  |        |
|-----------------------------|--------------------------|--|--------|
|                             | $\times$                 | $+$  | $\div$ |
| $\mathbf{f}_i^n$ –update    | $\frac{N}{K}$            | $\frac{2N}{K} + 4$                               | 1      |
| $\mathbf{h}$ –update        | -                        | -  | -      |
| $\mathbf{f}_i^n$ –filtering | $\frac{N}{R}$            | $\frac{(N-1)}{R}$                                | -      |
| $\mathbf{h}$ –filtering     | $\frac{N}{R}$            | $\frac{(N-1)}{R}$                                | -      |
| <b>Subtotal</b>             | $\frac{2N}{R}$           | $\frac{(2N-2)}{R}$                               | -      |
| $\bar{x}_k$ –soft input     | $2^{M+1}$                | $2^M + 10M - 1$                                  | -      |
| $L_e(c_{k,j})$ –soft output | $2^{M+1}$                | $(M^2 - M + 1)2^M$                               | -      |
| <b>Total</b>                | $\frac{2N}{R} + 2^{M+2}$ | $\frac{(2N-2)}{R} + (M^2 - M + 2)2^M + (9M + 3)$ | 1      |

channel is time-invariant. From our simulations, a table with 16 reference optimal filters is good enough. Thus, the complexity for (E.1) is four additions for the table lookup. The complexity for computing  $\tanh(\cdot)$  via the table lookup equals  $9M$  additions included in the complexity of the soft input conversion. At initialization, reference optimal filters are calculated and store in the table. Since this only perform once, the complexity is ignorable.

For the LSL equalizer, the detailed complexity is tabulated in Table E.2. Note that the optimal filter is obtained with the FFT approximation method in [55, pp. 111].

For the BCJR equalizer, the detail complexity is tabulated in Table E.3. The system parameters are the same as filter-based turbo equalizer, thus, the total number of state equals  $M^\mu = M^{(L-1)}$  and total numbers of state transition (branch) equal  $M^L$ . Because  $\max^*(x, y) =$

Table E.2: Complexity of LSL equalizer

| Equalizer type              | LSL   |   |         |
|-----------------------------|---|---|---------|
| Function                    | ×   | +   | ÷       |
| $\mathbf{f}_i^n$ –update    | $\frac{6L \log_2 L + 3L + 2}{K}$                | $\frac{4L \log_2 L + 9L + 1}{K}$                                    | $L + 2$ |
| $\mathbf{h}$ –update        | -   | -   | -       |
| $\mathbf{f}_i^n$ –filtering | $N$   | $N - 1$   | -       |
| $\mathbf{h}$ –filtering     | $N$   | $N - 1$   | -       |
| <b>Subtotal</b>             | $\frac{6L \log_2 L + 3L + 2}{K} + 2N$           | $\frac{4L \log_2 L + 9L + 1}{K} + (2N - 2)$                         | $L + 2$ |
| $\bar{x}_k$ -soft input     | $2^{M+1}$                                       | $2^M + 10M - 1$   | -       |
| $L_e(c_{k,j})$ -soft output | $2^{M+1}$                                       | $(M^2 - M + 1)2^M$  | -       |
| <b>Total</b>                | $\frac{6L \log_2 L + 3L + 2}{K} + 2N + 2^{M+2}$ | $\frac{4L \log_2 L + 9L + 1}{K} + (2N - 2) + (M^2 - M + 2)2^M + 9M$ | $L + 2$ |

$\max(x, y) + \ln(1 + e^{|x-y|}) = \ln(e^x + e^y)$  which consists of two operations  $\max(x, y)$  and a correction factor  $\ln(1 + e^{|x-y|})$ , is intensively used in the BCJR, the complexity considered here is in terms of  $\max^*$  operations. Later, the complexity of  $\max^*$  is equivalent to a two eight-bit additions [107]. Note that the normalization cost is already included in forward and backward path metric calculation.

For the BCJR decoder, the detail complexity is tabulated in Table E.4. The decoder considered here is a binary half-rate convolutional code with a memory size  $\eta$ . Thus, the total number of states equals  $2^\eta$  and the total number of state transition equal  $2^{\eta+1}$ . Note that the decision is made on the last iteration. Its complexity is divided by the total number of iterations accordingly. Assume that a division costs 40 FLOPS [55, pp. 108]. The complexity summary for different turbo equalizers in terms of is tabulated in Table E.5. A FLOP [73, pp. 19] is defined as a real number floating point operation (i.e., a floating point addition or a multiplication, , with some indexing). Note that the complexity of the decoder is evaluated in terms of (/bit/iteration).

Table E.3: Complexity of BCJR equalizer

| Equalizer type                    | BCJR equalizer   |  |                               |
|-----------------------------------|------------------|--|-------------------------------|
|                                   | ×                | +                                      | max*                          |
| Function                          | ×                | +                                      | max*                          |
| Branch metric                     | $2 \cdot 2^{ML}$ | $2^{ML}$                               | -                             |
| Forward path metric               | -                | $(2^M + 1) \cdot 2^{M(L-1)}$           | $(2^M - 1) \cdot 2^{M(L-1)}$  |
| Backward path metric              | -                | $(2^M + 1) \cdot 2^{M(L-1)}$           | $(2^M - 1) \cdot 2^{M(L-1)}$  |
| $L_e(c_{k,j})$ : extrinsic output | -                | $2 \cdot 2^{ML}$                       | $(2^M - 1) \cdot 2^{M(L-1)}$  |
| Total                             | $2 \cdot 2^{ML}$ | $5 \cdot 2^{ML} + 2 \cdot 2^{M(L-1)}$  | $3(2^M - 1) \cdot 2^{M(L-1)}$ |
| Equivalent total                  | $2 \cdot 2^{ML}$ | $11 \cdot 2^{ML} - 4 \cdot 2^{M(L-1)}$ | -                             |

Meanwhile, the complexity of equalizer is in terms of (/symbol/iteration). For a system with  $1/r$ -rate convolutional code and  $M$ -ary modulation scheme,  $\log_2 M$  coded bits are mapped to a symbol, i.e.,

$$1 \text{ information bit} = \frac{r}{\log_2 M} \text{ symbol.} \quad (\text{E.2})$$

Thus, the total complexity per information bit per iteration equals

$$C_{total} = C_D + \frac{r}{\log_2 M} C_E, \quad (\text{E.3})$$

where  $C_D$  is the complexity of the decoder,  $C_E$  is the complexity of the equalizer per information bit per iteration. The total complexity per information bit equals  $P \cdot C_{total}$ . From (E.3), we can see that a higher-order modulation scheme is preferred for complexity consideration especially when the complexity is dominated by the equalizer. The complexity of the interleaver and deinterleaver, which is one order less than the decoder or equalizer [55, pp. 127], is ignored here for simplicity.

The complexities for different equalizers without SISO conversion are also tabulated in Table E.6. Note that the complexity ratio  $R \leq 1$  of IR interpolation is summarized as follows

Table E.4: Complexity of BCJR decoder

| Decoder                           | BCJR decoder |   |                                  |
|-----------------------------------|--------------|---|----------------------------------|
| Function                          | ×            | +   | max*                             |
| Branch metric                     | -            | $2 \cdot 2^\eta$                                      | -                                |
| Forward path metric               | -            | $3 \cdot 2^\eta$                                      | $2^\eta$                         |
| Backward path metric              | -            | $3 \cdot 2^\eta$                                      | $2^\eta$                         |
| $L_e(b_{m,l})$ : extrinsic output | -            | $8 \cdot 2^\eta + 2$                                  | $4 \cdot 2^\eta$                 |
| $L(a_m)$ : APP                    | -            | $\frac{4 \cdot 2^\eta + 1}{P}$                        | $\frac{2 \cdot 2^\eta}{P}$       |
| $\hat{a}_m$ : decision            | -            | 1   | -                                |
| Total                             | -            | $(16 + \frac{4}{P}) \cdot 2^\eta + (3 + \frac{1}{P})$ | $(6 + \frac{2}{P}) \cdot 2^\eta$ |
| Equivalent total                  | -            | $(28 + \frac{8}{P}) \cdot 2^\eta + (3 + \frac{1}{P})$ | -                                |

Table E.5: Complexity summary of the turbo equalizers

| SISO function   | FLOPS/symbol/iteration   |
|-----------------|--|
| BCJR decoder    | $(28 + \frac{8}{P}) \cdot 2^\eta + (3 + \frac{1}{P})$                                |
| BCJR equalizer  | $(13 \cdot 2^M - 4)2^{M(L-1)}$   |
| OSL equalizer   | $10L \log_2 L + 42L + 4N + 81 + 2^{M+2} + (M^2 - M + 2)2^M + 9M$                     |
| LSL equalizer   | $\frac{10L \log_2 L + 12L + 3}{K} + 40L + 4N - 78 + 2^{M+2} + (M^2 - M + 2)2^M + 9M$ |
| FDISL equalizer | $R(4N - 2) + 44 + 2^{M+2} + (M^2 - M + 2)2^M + 9M$                                   |

[15]:

$$\begin{aligned}
 R &= \frac{\left( \underbrace{\left\lfloor \frac{N-t}{U} \right\rfloor}_{IFIR} + \underbrace{t}_{FIR} + \underbrace{(s-1)U}_{overlapped} + \underbrace{(2sU-1)}_{interpolation\ filter} \right)}{N} \\
 &= \frac{\left( \left\lfloor \frac{N-t}{U} \right\rfloor + t + (3s-1)U - 1 \right)}{N}, \tag{E.4}
 \end{aligned}$$

where  $t$  is the cutting point,  $U$  is the interpolation factor, and  $s$  is the span of interpolation filter.

Table E.6: Complexity of SISO equalizers without SISO conversion

| Equalizer type | FLOPS/symbol/iteration                             |
|----------------|--|
| OSL            | $10L \log_2 L + 52L + 4N + 81$                     |
| LSL            | $\frac{10L \log_2 L + 12L + 3}{K} + 40L + 4N + 78$ |
| FDISL          | $R(4N - 2) + 44$                                   |





# Bibliography

- [1] W. Y. Chen, *DSL Simulation Techniques and Standards Development for Digital Subscriber Line Systems*. Indianapolis, IN: Macmillan Technical Publishing, 1998.
- [2] N. Verhoeckx, H. van den Elzen, F. Sniijders, and P. van Gerwen, "Digital echo cancellation for baseband data transmission," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, no. 6, pp. 768–781, Dec. 1979.
- [3] W. Y. Chen, J. L. Dixon, and D. L. Waring, "High bit rate digital subscriber line echo cancellation," *IEEE J. Select. Areas Commun.*, vol. 9, no. 6, pp. 848–860, Aug. 1991.
- [4] J. J. Werner, "The HDSL environment [high bit rate digital subscriber line]," *IEEE J. Select. Areas Commun.*, vol. 9, no. 6, pp. 785–800, Aug. 1991.
- [5] *High Bit-rate Digital Subscriber Line (HDSL) Transmission System on Metallic Local Lines*, ETSI TS 101 135, Sep. 2000.
- [6] *High Bit Rate Digital Subscriber Line (HDSL) Transceivers*, ITU-T Recommendation G.991.1, Oct. 1998.
- [7] *High Bit Rate Digital Subscriber Line - 2nd Generation (HDSL2)*, ANSI Standard T1.418 Issue 2, 2002.
- [8] *Single-Pair High-Speed Digital Subscriber Line (SHDSL) Transceivers*, ITU-T Recommendation G.991.2, Feb. 2001.

- [9] *Access Transmission System on Metallic Access Cables; Symmetrical Single Pair High Bitrate Digital Subscriber Line (SDSL); Part 1: Functional Requirements*, ETSI TS 101 524-1, Apr. 2000.
- [10] *Access Transmission System on Metallic Access Cables; Symmetrical Single Pair High Bitrate Digital Subscriber Line (SDSL); Part 2: Transceiver Requirements*, ETSI TS 101 524-2, June 2000.
- [11] A. N. Kaelin, A. G. Lindgren, and G. S. Moschytz, "Simplified adaptive IIR filters based on optimized orthogonal prefiltering," *IEEE Trans. Circuits Syst. II: Analog and Digital Signal Processing*, vol. 42, no. 5, pp. 326–333, May 1995.
- [12] G. W. Davidson and D. D. Falconer, "Reduced complexity echo cancellation using orthonormal functions," *IEEE Trans. Circuits Syst.*, vol. 38, no. 1, pp. 20–28, Jan. 1991.
- [13] T. Aboulnasr, A. Abousaada, and W. Steenaart, "Efficient implementation of echo cancellers for ISDN applications," in *IEEE Int. Conf. Acoust., Speech, Signal Processing ICASSP'89*, vol. 2, May 1989, pp. 1380–1383.
- [14] A. Abousaada, T. Aboulnasr, and W. Steenaart, "An echo tail canceller based on adaptive interpolated FIR filtering," *IEEE Trans. Circuits Syst. II: Analog and Digital Signal Processing*, vol. 39, no. 7, pp. 409–416, July 1992.
- [15] S.-S. Lin and W.-R. Wu, "A low-complexity adaptive echo canceller for xDSL applications," *IEEE Trans. Signal Processing*, vol. 52, no. 5, pp. 1461–1465, May 2004.
- [16] —, "A low complexity adaptive interpolated FIR echo canceller," in *Proc. IEEE Int. Symp. Circuits and Syst. ISCAS'01*, vol. 4, May 2001, pp. 438–441.
- [17] Y. Neuvo, C. Y. Dong, and S. K. Mitra, "Interpolated finite impulse response filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, no. 3, pp. 563–570, June 1984.

- [18] N. J. Fliege, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*. New York, NY: John Wiley & Sons, 1994.
- [19] S. Haykin, *Adaptive Filter Theory*, 2nd ed. London: Prentice Hall International, 1991.
- [20] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear estimation*. Upper Saddle River, NJ: Prentice Hall, 2000.
- [21] T. Starr, J. M. Cioffi, and P. Silverman, *Understanding digital subscriber line technology*. Upper Saddle River, NJ: Prentice Hall, 1999.
- [22] *Standards Project for Interfaces Relating to Carrier to Customer Connection of Asymmetrical Digital Subscriber Line (ADSL) Equipment*, ANSI Standard T1.413 Issue 2, Dec. 1998.
- [23] *Test procedures for digital subscriber line (DSL) transceivers*, ITU-T Recommendation G.996.1, Feb. 2001.
- [24] C. E. Belfiore and J. H. Park, Jr., "Decision feedback equalization," *Proc. IEEE*, vol. 67, pp. 1143–1156, Aug. 1979.
- [25] P. M. Crespo and M. L. Honig, "Pole-zero decision feedback equalization with a rapidly converging adaptive IIR algorithm," *IEEE J. Select. Areas Commun.*, vol. 9, no. 6, pp. 817–829, Aug. 1991.
- [26] G. Young, "Reduced complexity decision feedback equalization for digital subscriber loops," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 810–816, Aug. 1991.
- [27] N. Al-Dhahir, A. H. Sayed, and J. M. Cioffi, "Stable pole-zero modeling of long FIR filters with application to the MMSE-DFE," *IEEE Trans. Commun.*, vol. 45, no. 5, pp. 508–513, May 1997.



- [28] S. Ariyavisitakul and L. J. Greenstein, "Reduced-complexity equalization techniques for broadband wireless channels," *IEEE J. Select. Areas Commun.*, vol. 15, no. 1, pp. 5–15, Jan. 1997.
- [29] I. J. Fevrier, S. B. Gelfand, and M. P. Fitz, "Reduced complexity decision feedback equalization for multipath channels with large delay spreads," *IEEE Trans. Commun.*, vol. 47, no. 6, pp. 927–937, June 1999.
- [30] A. A. Rontogiannis and K. Berberidis, "Efficient decision feedback equalization for sparse wireless channels," *IEEE Trans. Wireless Commun.*, vol. 2, no. 3, pp. 570–581, May 2003.
- [31] N. Al-Dhahir and J. M. Cioffi, "Fast computation of channel-estimate based equalizers in packet data transmission," *IEEE Trans. Signal Processing*, vol. 43, no. 11, pp. 2462–2473, Nov. 1995.
- [32] M. Tomlinson, "New automatic equalizer employing modulo arithmetic," *Electron. Lett.*, vol. 7, pp. 138–139, Mar. 1971.
- [33] H. Harashima and H. Miyakawa, "Matched-transmission technique for channels with intersymbol interference," *IEEE Trans. Commun.*, vol. 20, no. 4, pp. 774–780, Aug. 1972.
- [34] R. F. H. Fischer and J. B. Huber, "Comparison of precoding schemes for digital subscriber lines," *IEEE Trans. Commun.*, vol. 45, no. 3, pp. 334–343, Mar. 1997.
- [35] N. Al-Dhahir and J. M. Cioffi, "MMSE decision-feedback equalizers: finite-length results," *IEEE Trans. Inform. Theory*, vol. 41, no. 4, pp. 961–975, July 1995.
- [36] S.-S. Lin and W.-R. Wu, "An optimal interpolated FIR echo canceller for digital subscriber lines," *IEICE Trans. Commun.*, vol. 87-B, no. 12, pp. 3584–3592, Dec. 2004.

- [37] E. Baccarelli, A. Fasano, and M. Biagi, "A novel tunable-complexity turbo-soft detector for high-throughput HDSL applications over ISI channels with crosstalk," *IEEE J. Select. Areas Commun.*, vol. 20, no. 2, pp. 372–383, Feb. 2002.
- [38] R. Koetter, A. Singer, and M. Tüchler, "Turbo equalization," *IEEE Signal Processing Mag.*, vol. 20, no. 1, pp. 67–80, Jan. 2004.
- [39] C. Douillard, M. Jézéquel, A. P. C. Berrou, P. Didier, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," *European Trans. Telecommun.*, vol. 6, no. 5, pp. 507–511, Sep.-Oct. 1995.
- [40] J. Hagenauer and P. Hoeher, "Viterbi algorithm with soft-decision outputs and its applications," in *Proc. IEEE Global Telecommun. Conf.*, Nov. 1989, pp. 1680–1686.
- [41] G. Bauch and V. Franz, "A comparison of soft-in/soft-out algorithms for "Turbo-detection"," in *Proc. Int. Conf. Telecommun. ICT'98*, June 1998, pp. 259–263.
- [42] X. Wang and H. Poor, "Iterative (turbo) soft interference cancellation and decoding for coded CDMA," *IEEE Trans. Commun.*, vol. 47, no. 7, pp. 1046–1061, July 1999.
- [43] D. Reynolds and X. Wang, "Low complexity turbo-equalization for diversity channels," *Signal Processing*, vol. 81, no. 5, pp. 989–995, May 2001.
- [44] M. Tüchler, "Iterative equalization and decoding using priors," Master's thesis, Univ. of Illinois, Urbana-Champaign, IL, May 2000.
- [45] M. Tüchler, A. Singer, and R. Koetter, "Minimum mean squared error MMSE equalization using priors," *IEEE Trans. Signal Processing*, vol. 50, no. 3, pp. 673–683, Mar. 2002.
- [46] M. Tüchler, R. Koetter, and A. Singer, "Turbo equalization: Principles and new results," *IEEE Trans. Commun.*, vol. 50, no. 5, pp. 754–767, May 2002.

- [47] J. M. Cioffi, G. P. Dudevoir, M. V. Eyuboglu, and G. D. Forney, Jr., “MMSE decision-feedback equalizers and coding — Part I: Equalization results,” *IEEE Trans. Commun.*, vol. 43, no. 10, pp. 2582–2594, Oct. 1995.
- [48] C. Laot, “Égalisation autodidacte et turbo-égalisation - application aux canaux sélectifs en fréquence,” Ph.D. dissertation, Univ. de Rennes I, Rennes, France, July 1997, (in French).
- [49] C. Laot, A. Glavieux, and J. Labat, “Turbo-equalization: Adaptive equalization and channel decoding jointly optimized,” *IEEE J. Select. Areas Commun.*, vol. 19, no. 9, pp. 1744–1752, Sep. 2001.
- [50] C. Langlais, “Étude et amélioration d’une technique de réception itérative: Turbo-égalisation,” Ph.D. dissertation, Univ. de Rennes I, Rennes, France, Nov. 2002, (in French).
- [51] D. Raphaeli and A. Saguy, “Linear equalizer for turbo equalization - a new optimization criterion for determining the equalizer taps,” in *Proc. 2nd Int. Symp. on Turbo Codes & Related Topics*, Brest, France, 4-7 Sep. 2000, pp. 371–374.
- [52] —, “Reduced complexity APP for turbo equalization,” in *Proc. IEEE Int. Conf. Commun. ICC’02*, vol. 3, New York, NY, 28 Apr.-2 May 2002, pp. 1940–1943.
- [53] M. Tüchler and J. Hagenauer, “Turbo equalization using frequency domain equalizers,” in *Proc. 38th Annual Allerton Conf. Commun., Control and Computing*, Monticello, IL, 4-6 Oct. 2000.
- [54] —, “Linear time and frequency domain turbo equalization,” in *Proc. IEEE 53rd Veh. Technol. Conf.*, vol. 2, May 2001, pp. 1449–1453.

- [55] R. Le Bidan, "Turbo-equalization for bandwidth-efficient digital communications over frequency-selective channels," PhD. dissertation, Univ. de Rennes I, Rennes, France, Nov. 2003.
- [56] F. Pancaldi and G. M. Vitetta, "Block channel equalization in the frequency domain," *IEEE Trans. Commun.*, vol. 53, no. 3, pp. 463–471, Mar. 2005.
- [57] A. Dejonghe and L. Vandendorpe, "A comparison of bit and symbol interleaving in MMSE turbo-equalization," in *Proc. IEEE Int. Conf. Commun. ICC'03*, vol. 4, Anchorage, Alaska, 11-15 May 2003, pp. 2928–2932.
- [58] —, "Turbo-equalization for multilevel modulation: An efficient low-complexity scheme," in *Proc. IEEE Int. Conf. Commun. ICC'02*, vol. 3, New York, NY, 28 Apr.-2 May 2002, pp. 1863–1867.
- [59] S. Dolinar and D. Divsalar, "Weight distributions for turbo-codes using random and non-random permutations," *JPL TDA Progress Report*, vol. 42-122, Jet Propulsion Laboratory, Pasadena, CA, Aug. 1995.
- [60] J. Hagenauer, "The turbo principle: Tutorial introduction and state of the art," in *Proc. Int. Symp. on Turbo Codes & Related Topics*, Brest, France, 3-5 Sep. 1997, pp. 1–11.
- [61] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284–287, Mar. 1974.
- [62] G. Colavolpe, G. Ferrari, and R. Raheli, "Extrinsic information in iterative decoding: A unified view," *IEEE Trans. Commun.*, vol. 49, no. 12, pp. 2088–2094, Dec. 2001.
- [63] A. Gersho and T. L. Lim, "Adaptive cancellation of intersymbol interference for data transmission," *Bell Sys. Tech. J.*, vol. 60, no. 11, pp. 1997–2021, Nov. 1981.

- [64] M. S. Mueller and J. Salz, "A unified theory of data-aided equalization," *Bell Sys. Tech. J.*, vol. 60, no. 11, pp. 2023–2038, Nov. 1981.
- [65] J. G. Proakis, "Adaptive nonlinear filtering techniques for data transmission," in *Proc. IEEE Symp. on Adaptive Processes, Decision and Control*, 1970, pp. XV.2.1–XV.2.5.
- [66] R. Otnes and M. Tüchler, "Block SISO linear equalizers for turbo equalization in serial-tone HF modems," in *Proc. Norwegian Signal Processing Symp. (NORSIG)*, Trondheim, Norway, pp. 93-98, Oct 2001.
- [67] S. ten Brink, "Convergence of iterative decoding," *Electron. Lett.*, vol. 35, no. 13, pp. 1117–1119, June 1999.
- [68] S. J. Lee, A. C. Singer, and N. R. Shanbhag, "Analysis of linear turbo equalizer via EXIT chart," in *IEEE Global Telecommun. Conf. GLOBECOM'03*, vol. 4, 1-5 Dec. 2003, pp. 2237–2242.
- [69] J. G. Proakis, *Digital Communications*, 3rd ed. New York, NY: McGraw-Hill, 1995.
- [70] R. L. Burden and J. D. Faires, *Numerical Analysis*, 7th ed. Pacific Grove, CA: Brooks/Cole, 2001.
- [71] B. Nicoletti and L. Mariana, "A computer algorithm for optimal curve fitting," *IEEE Trans. Automat. Contr.*, pp. 90–92, Feb. 1971.
- [72] T. Starr, J. M. Cioffi, and P. Silverman, *Understanding Digital Subscriber Line Technology*. Englewood Cliffs, NJ: Prentice-Hall, 1998.
- [73] G. H. Golub and C. F. Van Loan, *Matrix Computation*, 2nd ed. Baltimore, MD: The Johns Hopkins University Press, 1989.
- [74] G. Caire, G. Taricco, and E. Biglieri, "Bit-interleaved coded modulation," *IEEE Trans. Inform. Theory*, vol. 44, no. 3, pp. 927–946, May 1998.



- [75] O. Y. Takeshita, O. M. Collins, P. C. Massey, and D. J. Costello, Jr., "On the frame-error rate of concatenated turbo codes," *IEEE Trans. Commun.*, vol. 49, no. 4, pp. 602–608, Apr. 2001.
- [76] K. Narayanan, "Effect of precoding on the convergence of turbo equalization for partial response channels," *IEEE J. Select. Areas Commun.*, vol. 19, no. 4, pp. 686–698, Apr. 2001.
- [77] I. Lee, "The effect of a precoder on serially concatenated coding systems with an ISI channel," *IEEE Trans. Comm.*, vol. 49, no. 7, pp. 1168–1175, July 2001.
- [78] D. Raphaeli and Y. Zarai, "Combined turbo equalization and turbo decoding," *IEEE Commun. Lett.*, vol. 2, no. 4, pp. 107–109, Apr. 1998.
- [79] C. Berrou, A. Glavieux, , and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. IEEE Int. Conf. Commun. ICC'93*, vol. 2, Geneva, Switzerland, 23-26 May 1993, pp. 1064–1070.
- [80] K. R. Narayanan, X. Wang, and G. Yue, "LDPC code design for turbo equalization," in *Proc. IEEE Inform. Theory Workshop ITW'02*, Bangalore, India, 20-25 Oct. 2002, pp. 57–60.
- [81] S. Ölçer and M. Keskinöz, "Performance of MMSE turbo equalization using outer LDPC coding for magnetic recording channels," in *Proc. IEEE Int. Conf. Commun. ICC'04*, vol. 2, 20-24 June 2004, pp. 645–650.
- [82] H. Song and B. Vijaya Kumar, "Low-density parity check codes for partial response channels," *IEEE Signal Processing Mag.*, vol. 21, no. 1, pp. 56–66, 2004.
- [83] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: MIT Press, 1963.
- [84] C. E. Shannon, "A mathematical theory of communication," *Bell Sys. Tech. J.*, vol. 27, pt. I, pp. 379-423, 1948; pt. II, pp. 623-656, July-Oct. 1948.

- [85] R. Otnes and M. Tüchler, "Soft iterative channel estimation for turbo equalization: Comparison of channel estimation algorithms," in *Proc. IEEE Int. Conf. on Commun. Systems ICCS'02*, vol. 1, Singapore, 25-28 Nov. 2002, pp. 72–76.
- [86] R. Otnes, "Improved receivers for digital high frequency communications: Iterative channel estimation, equalization, and decoding (adaptive turbo equalization)," Ph.D. dissertation, Norwegian Univ. of Science and Technology, Trondheim, Norway, Nov. 2002.
- [87] R. Otnes and M. Tüchler, "Iterative channel estimation for turbo equalization of time-varying frequency-selective channels," *IEEE Trans. Wireless Commun.*, vol. 3, no. 6, pp. 1918–1923, Nov. 2004.
- [88] T. Hwang and Y. Li, "Iterative cyclic prefix reconstruction for coded single-carrier systems with frequency-domain equalization (SC-FDE)," in *IEEE Veh. Technol. Conf. VTC'03-Spring*, vol. 3, 22-25 Apr. 2003, pp. 1841–1845.
- [89] *Asymmetric digital subscriber line transceivers 2 (ADSL2)*, ITU-T Recommendation G.992.3, Jan. 2005.
- [90] *Asymmetric Digital Subscriber Line (ADSL) transceivers - Extended bandwidth ADSL2 (ADSL2+)*, ITU-T Recommendation G.992.5, Jan. 2005.
- [91] *Very high speed digital subscriber line transceivers*, ITU-T Recommendation G.993.1, June 2004.
- [92] *Very high speed digital subscriber line transceivers 2*, ITU-T Recommendation G.993.2.
- [93] S. Trautmann and N. J. Fliege, "Perfect equalization for DMT systems without guard interval," *IEEE J. Select. Areas Commun.*, vol. 20, no. 5, pp. 987–996, June 2002.
- [94] C.-J. Park and G.-H. Im, "Efficient DMT/OFDM transmission with insufficient cyclic prefix," *IEEE Commun. Lett.*, vol. 8, no. 9, pp. 576–578, Sep. 2004.

- [95] ———, “Efficient cyclic prefix reconstruction for coded OFDM systems,” *IEEE Commun. Lett.*, vol. 8, no. 5, pp. 274–276, May 2004.
- [96] D. Vogrig, A. Gerosa, A. Neviani, A. G. i Amat, G. Montorsi, and S. Benedetto, “A 0.35- $\mu\text{m}$  CMOS analog turbo decoder for the 40-bit rate 1/3 UMTS channel code,” *IEEE J. Solid-State Circuits*, vol. 40, no. 3, pp. 753–762, March 2005.
- [97] F. Lustenberger, “On the design of analog vlsi iterative decoders,” Ph.D. dissertation, ETH, Zurich, Switzerland, 2000.
- [98] J. Dai, “Design methodology for analog vlsi implementations of error control decoders,” Ph.D. dissertation, Univ. Utah, Salt Lake, 2001.
- [99] J. Hagenauer, E. Offer, C. Méasson, and M. Mörz, “Decoding and equalization with analog non-linear networks,” *ETT European Trans. Telecommun.*, vol. 10, pp. 659–680, Nov.-Dec. 1999.
- [100] J. Hagenauer, M. Mörz, and E. Offer, “Analog turbo-networks in VLSI: The next step in turbo decoding and equalization,” in *Proc. 2nd Int. Symp. on Turbo Codes & Related Topics*, Brest, France, 4-7 Sep. 2000, pp. 209–218.
- [101] N. Correal, J. Heck, and M. Valenti, “An analog turbo decoder for an (8,4) product code,” in *IEEE 45th Midwest Symp. on Circuits and Syst. MWSCAS’02*, vol. 3, 4-7 Aug. 2002, pp. 632–635.
- [102] J. Hagenauer, M. Mörz, and A. Schaefer, “Analog decoders and receivers for high speed applications,” in *Int. Zurich Seminar on Broadband Commun.*, vol. 3, 19-21 Feb. 2002, pp. 1–8.
- [103] J. R. Barry, “The BCJR algorithm for optimal equalization,” School of Electrical and Computer Engineering, Georgia Institute of Technology, Mar. 2000, [Online]. Available: <http://users.ece.gatech.edu/~barry/>.

- [104] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, 3rd ed. New York, NY: McGraw-Hill, 1991.
- [105] G. D. Forney, Jr., "The Viterbi algorithm," *Proc. IEEE*, vol. 61, pp. 268–278, 1973.
- [106] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proc. IEEE Int. Conf. Commun. ICC'95*, vol. 2, Seattle, WA, 18-22 June 1995, pp. 1009–1013.
- [107] W. J. Gross and P. G. Gulak, "Simplified MAP algorithm suitable for implementation of turbo decoders," *Electron. Lett.*, vol. 34, no. 16, pp. 1577–1578, Aug. 1998.

