Design of Complex Digital FIR Filters in the Chebyshev Sense

A THESIS

Presented to

The Academic Faculty

By

Lina J. Karam

In Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy in Electrical Engineering

Georgia Institute of Technology

March 1995

Copyright © 1995 by Lina J. Karam

Design of Complex Digital FIR Filters in the Chebyshev Sense

Approved:

James H. McClellan, Chairman

Russell M. Mersereau

Ronald W. Schafer

Date approved by Chairman _____

To my parents, Jamil & Fida,

and to my husband, Rida

Acknowledgments

Many many thanks to people from all ranks
Professors, students, family, and friends
Many many thanks for helping me
Overcome every difficulty
Climb the mountains of GT
And earn my PhD

Lina J. Karam

Foremost, I would like to thank very much Professor James H. McClellan for being my advisor. I deeply appreciate his support, encouragement, understanding, and valuable advice.

I thank the members of my defense committee: Professors Earl Barnes, James Herod, Russ Mersereau, and Ron Schafer. I am particularly grateful to Prof. James McClellan, Prof. Russ Mersereau, and Prof. Ron Schafer for serving as readers of this thesis and for always being there whenever I needed their help. I thank very much Prof. Vijay Madisetti for his friendship and constructive comments, Prof. Yorai Wardi for advising me to specialize in Signal Processing, and Prof. Adnan Al-Alaoui for encouraging me to come to Georgia Tech. I would also like to thank Dr. Bill Ribarsky and Prof. James Foley for giving me the opportunity to work in the Graphics, Visualization, and Usability Center during the first two years of my graduate studies. I also appreciate the help of Kay Gilstrap and Stacy Schultz in all administrative matters and the assistance of Lonnie Harvel, Sam Smith, and Dave Webb in problems related to the computing environment.

I sincerely thank the Hariri Foundation for the financial support which made it possible for me to live and study in the United States. I especially thank Dr. David

Thompson at the Hariri Foundation for his advice and assistance.

I sincerely thank ALL the fellow students and graduates whom I have had the chance to meet and interact with. I would especially like to mention (in alphabetical order) Osama Al-Sheikh, Halûk Aydinoğlu, Mohamed Ben Romdhane, Thomas Egolf, Dinei Florencio, Takoi Hamrita, Stephen Martucci, Baldine Paul, Richard Rau, Jonathan Su, and Jack Tumblin. I am particularly grateful to Hani and Nuhad Atrash who have always been willing to help me and have been my family in Atlanta.

I would like to thank my brother Salim for convincing me to specialize in Engineering. The songs we wrote together were a shelter for my mind and for my soul in times of difficulty. I would also like to thank my sisters Nahla and Hoda for their care and concern about me. My warmest thoughts and deepest thanks go to my husband Rida. His endless love, confidence in me, and friendship have empowered me to face the challenges of this world. I cannot find the right words to express how much I am grateful to my parents who revealed to me the importance of education and have always been the source of my ambition. The deep unconditional love of my family is what has made this thesis possible.

Contents

Acknowledgments **iv**

Contents **vi**

List of Tables **x**

List of Figures **xi**

Summary **xxi**

1 Introduction **1**

1.1 Overview 1

1.2 Organization of the Thesis 5

2 Complex Chebyshev Approximation **8**

2.1 Formulation of the Chebyshev Approximation Problem 8

2.2 Existing Optimality Criteria 9

2.3 Solution on Finite Subsets of Points in a Multi-dimensional Space . . 13

2.4 New Simple Optimality Criteria 18

2.4.1 Constrained Chebyshev Approximation Problem 18

2.4.2 Alternation in the Complex case 19

2.5 Summary 23

3 Optimal Complex FIR Filter Design **24**

3.1 Reformulation of the Filter Design Problem 24
3.2 Calculation of the Best Approximation 27
3.2.1 Motivation 28
3.2.2 The Complex Remez Multiple-Exchange Algorithm 33
3.2.3 The Ascent-Descent Multiple-Exchange Algorithm 40
3.2.4 Relation to the Real (Linear-Phase) Filter Design Problem . . . 48
3.3 Multidimensional Chebyshev Approximation 50
3.4 Special Types of Filters 51
3.4.1 Low Delay Filters with Nearly-linear Phase 51
3.4.2 Seismic Migration Filters with Quasi-Quadratic Phase 52
3.5 Design Examples 55
3.5.1 Reduced Delay, Nearly Linear-Phase FIR Filters 57
3.5.2 Exactly Linear-Phase FIR Filters 70
3.5.3 Seismic Migration Filters 73
3.6 Comparison with other existing algorithms 83

4 Multidimensional FIR Filter Design by Transformation 86

4.1 Brief Overview of the McClellan Transformation Method 87
4.2 Identification of Transformable Prototype Filters 90
4.2.1 Transforming an Arbitrary Prototype Filter 90
4.2.2 Transforming Symmetric Complex Prototype Filters 95
4.3 Design of Complex Positive- and Negative-symmetric 2-D Filters: Feasibility and Symmetry Conditions 100
4.3.1 Design of Odd-Length 2-D Filters 101
4.3.2 Design of Even-Length 2-D Filters 103
4.4 Design Examples 110
4.4.1 Example 1: negative centro-symmetric 2-D filter, odd-length prototype 110

4.4.2 Example 2: complex quadrant-symmetric 2-D filter, even-length prototype . . . 111
4.4.3 Example 3 . . . 113
4.5 Summary . . . 127

5 Efficient Design of Families of Filters by Transformation 128
5.1 Families of Filters for 2-D and 3-D Depth Migration . . . 129
5.1.1 Part 1: Subfilter Design . . . 132
5.1.2 Part 2: Prototype Filter Design . . . 134
5.1.3 Part 3: Generating the 2-D and 3-D Depth Migration Filters . . . 135
5.2 Families of Variable-Cutoff Linear-Phase Filters . . . 138
5.2.1 Part 1: Subfilter Design . . . 140
5.2.2 Part 2: Prototype Filter Design . . . 141
5.2.3 Part 3: Realizing the variable-cutoff lowpass filters . . . 142
5.3 Constraints on the Transformation Subfilter . . . 143
5.4 Design Examples . . . 147
5.4.1 Families of migration filters . . . 149
5.4.2 Linear-Phase Variable-Cutoff Filters . . . 162

6 Conclusion 166
6.1 Contributions of the Thesis . . . 166
6.1.1 Chebyshev Approximation Theory . . . 166
6.1.2 Filter Design Theory . . . 168
6.1.3 Chebyshev Approximation Algorithms . . . 169
6.1.4 Design of Filters by Transformation . . . 171
6.2 Directions for Future Research . . . 173

Appendix A Notation and Terminology 175
A.1 Notation . . . 175
A.2 Terminology . . . 176

Appendix B Mathematical Derivations **177**

B.1 Addendum to Chapter 3 177

B.1.1 Derivation of (3.14) 177

B.2 Addendum to Chapter 4 178

B.2.1 Proof of Theorem 4.1 178

B.2.2 Proof of recurrence formula (4.14) 179

B.2.3 Proof of relation (4.15) 180

B.2.4 Proof of relation (4.16) 182

B.2.5 Proof of relation (4.17) 182

Appendix C Descent Method Based on the Kolmogoroff Criterion **184**

Appendix D Code Listing **189**

D.1 Multiple-Exchange Design Algorithm (Chapter 3) 189

D.2 Family Design Procedure (Chapter 5) 214

Appendix E Filter Coefficients Listing **224**

E.1 Nearly-linear Phase 224

E.2 Exactly-linear Phase 225

E.3 Seismic Migration 226

Bibliography **226**

Vita **235**

List of Tables

3.1 Summary of the numerical results. $|\delta|$ is the lower bound on the optimal error and is computed by the complex Remez algorithm. $\|E_{opt}\|$ is the optimal Chebyshev error norm. 56

3.2 Running time comparison with the Burnside-Parks [22] and Preuss-Schulist [19] algorithms run on a SUN SPARC 10 workstation. . 84

4.1 The designed 2-D filter has a rectangular region of support whose size depends on the prototype filter length, the transformation subfilter size, and the transformation technique used. 127

List of Figures

1.1 A filter is identified by its impulse response $h(\mathbf{n})$ or by its frequency response $H(\mathbf{w})$ which is the Discrete-Time Fourier Transform of $h(\mathbf{n})$ [1]. (a) Input-output relation of a linear time-invariant filter in the time domain; the output is the linear convolution of the input with $h(\mathbf{n})$. (b) Input-output relation of a linear time-invariant filter in the frequency domain; $X(\mathbf{w})$ and $Y(\mathbf{w})$ are the Discrete-Time Fourier Transforms [1] of the input $x(\mathbf{n})$ and the output $y(\mathbf{n})$, respectively. 2

1.2 Comparison of the Chebyshev error (top) with the least square error (bottom). While the least square error may randomly fluctuate, the magnitude of the Chebyshev error does not exceed an optimal deviation. 2

2.1 Ascent exchange strategy. 17

2.2 Schematic illustration of the optimal error properties. The light-shaded region represents cases where the optimal complex error alternates. 22

3.1 Ascent structure of the two stages and algorithm. 31

3.2 Block diagram of the multiple-exchange algorithm. $|\delta|$ is the maximal optimal deviation on the sets B_s consisting of $n+1$ points in B. $\|\varepsilon\|$ is the Chebyshev error norm on B. 32

3.3 Illustration of the replacement (3.12): Filled circles represent local maxima of $|\varepsilon(x)|$. Non-filled circles correspond to the candidate points $\{x_i\}_{i=1}^{n+1}$ on which the error $\varepsilon(x)$ is forced to alternate. ρ is one if a local maximum is added at the beginning (end) with the last (first) point x_{n+1} (x_1) removed. It is zero when a local maximum replaces and takes the same index of an adjacent point x_i ($i = 1, \ldots, n+1$). 35

3.4 Vector interpretation of the exchange rule to get a maximum increase in δ_{new}. The index i_o must be chosen so that $e^{j\rho\pi}\text{sgn}\{\lambda'_{i_o}\}\varepsilon(\zeta)$ lies in the same half-plane as δ. A stronger condition on $\varepsilon(\zeta)$ that ensures an increase is that the vector $\mathbf{v}$ also lie in the same half-plane as δ, because $e^{j\rho\pi}\delta_{\text{new}} = \delta + |\lambda'_{i_o}|\mathbf{v}$. 36

3.5 Graph illustrating one iteration of the multiple-exchange complex Remez algorithm. 38

3.6 Block diagram of the complex Remez algorithm. 39

3.7 Simplified model of the earth from [46]. 52

3.8 Example 1–A: nearly linear-phase filter, $N = 32$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). . . 58

3.9 Algorithm progression - Example 1 (nearly linear phase filter $N = 32$). The solid and dashed curves represent the values of $|\delta|$ and $\|E(\omega)\|$ respectively. 59

3.10 Example 1–B: nearly linear-phase filter, $N = 38$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). . . 60

3.11 Example 2: nearly linear-phase filter, $N = 25$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). . . 62

3.12 Example 3: nearly linear-phase filter, $N = 31$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). . . 64

3.13 Example 4: nearly linear-phase filter, $N = 40$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). . . 66

3.14 Example 4: nearly linear-phase filter, $N = 40$. Angle of the optimal error at the extremal points indicating a quasi-alternating behavior. 67

3.15 Example 4: nearly linear-phase filter, $N = 40$. Characteristics of the sub-optimal filter computed by the efficient complex Remez algorithm. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). 68

3.16 Example 4: nearly linear-phase filter, $N = 40$. Angle of the error, which is sub-optimal on B and optimal on a reduced set $B' \subset B$ (see Fig. 3.15(c)), at the extremals on B'. A perfect alternation behavior is exhibited. 69

3.17 Example 1: exactly linear-phase lowpass filter, $N = 32$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). . . 71

3.18 Example 2: exactly linear-phase passband filter, $N = 25$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). . . 72

3.19 Example 3: exactly linear-phase passband filter, $N = 31$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). . . 74

3.20 Example 4: exactly linear-phase passband filter, $N = 40$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). . . 75

3.21 Example 1: seismic migration filter, $N = 19$. (a) Magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Phase error in passband. (d) Magnitude of the error (top), and alternating real rotated error (bottom). . . 76

3.22 Example 2: seismic migration filter, $N = 35$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Phase error in passband. (d) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). 78

3.23 Example 2: seismic migration filter, $N = 36$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Phase error in passband. (d) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). 79

3.24 Example 2: seismic migration filter, $N = 36$. Initial sub-optimal solution computed by the complex Remez method. (a) FIR filter magnitude response in dB. (b) Phase error in passband. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). 80

3.25 Example 3: seismic migration filter, $N = 31$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). (d) Phase error in passband. 82

4.1 The original McClellan transformation method maps 1-D positive-symmetric filters $H(\omega)$ into 2-D positive-symmetric filters $H(\omega_1, \omega_2)$ using real and symmetric transformation functions $F(\omega_1, \omega_2)$. . . 87

4.2 The transformation method breaks down the 2-D design problem into two smaller design problems. 88

4.3 Realization of the 2-D filter by means of a Chebyshev structure. . . . 89

4.4 Realization of the odd-length 2-D filters which are designed by transforming even-length 1-D filters. The coefficients $\{c_n\}$ are obtained from the coefficients $\{h_n\}$ of the prototype 1-D filter as described in Step 2 of the design procedure. 99

4.5 Contours of the subfilters (4.65), (4.71), (4.80), and (4.86). The solid curves correspond to $0 \leq F(\omega_1, \omega_2) \leq 1$ while the dashed curves correspond to $-1 \leq F(\omega_1, \omega_2) < 0$. 107

4.6 Example 1: contours of the negative centro-symmetric subfilter. . . . 115
4.7 Example 1: prototype filter, $N = 31$ and $\omega_p = \pi/4$. (a) FIR filter magnitude response. (b) FIR filter magnitude response in dB. (c) Magnitude of the error (top), and real alternating error (bottom). 116
4.8 Example 1: frequency response of the transformed 2-D filter, $\omega_p = \pi/4$. 117
4.9 Example 1: prototype filter, $N = 31$ and $\omega_p = \pi/10$. (a) FIR filter magnitude response. (b) FIR filter magnitude response in dB. (c) Magnitude of the error (top), and real alternating error (bottom). 118
4.10 Example 1: frequency response of the transformed 2-D filter, $\omega_p = \pi/10$. 119
4.11 Example 2: contours of the quadrant-symmetric subfilter for designing the complex fan-shaped 2-D filter. 120
4.12 Example 2: prototype filter, $N = 30$ and $\omega_p = \pi/4$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired (dashed) passband phase responses plotted without the half-sample delay. (c) Magnitude of the error (top), and alternating real phase-rotated error (bottom). (d) Phase error in the two bands. 121
4.13 Example 2: two-dimensional fan-shaped phase filter designed by transforming a positive-symmetric even-length prototype with cutoff $\omega_p = \pi/4$. (a) Filter magnitude response. (b) Filter fan-shaped phase response. 122
4.14 Example 3: prototype filter, $N = 30$, $\omega_{p1} = \pi/2$, and $\omega_{p2} = \pi$. (a) FIR filter magnitude response. (b) FIR filter magnitude response in dB. (c) Magnitude of the error (top), and real alternating error (bottom). 123
4.15 Example 3: frequency response of the transformed 2-D filter of size 30×30 ($\omega_{p1} = \pi/2$, and $\omega_{p2} = \pi$). 124

4.16 Example 3: prototype filter, $N = 30$, $\omega_{p1} = \pi/4$, and $\omega_{p2} = \pi/2$. (a) FIR filter magnitude response. (b) FIR filter magnitude response in dB. (c) Magnitude of the error (top), and real alternating error (bottom). 125

4.17 Example 3: frequency response of the transformed 2-D filter of size 30×30 ($\omega_{p1} = \pi/4$, and $\omega_{p2} = \pi/2$). 126

5.1 Ideal hyperbolic contours of the migration filter family. 130

5.2 Chebyshev structure with $\{p_n\}_{n=0}^{L}$ and F representing the prototype coefficients and the transformation subfilter, respectively. 132

5.3 The two-in-one Chebyshev structure for realizing multi-dimensional filters. 136

5.4 Block diagram of the design procedure. 137

5.5 Frequency response of the ideal fan filter. 139

5.6 Ideal subfilter contours (solid lines) and McClellan's first-order transformation for fan filters (dotted curves). 140

5.7 Realization of a variable-cutoff bandpass filter by means of the Chebyshev structures for the variable-cutoff lowpass filter (see Figs. 5.2 and 5.3). The lower and upper cutoff frequencies of the realized bandpass filter are controlled by $\{f_n^l\}_{n=0}^{Q}$ and $\{f_n^u\}_{n=0}^{Q}$, respectively. The highpass transformation coefficients $\{f_n^l\}_{n=0}^{Q}$ are obtained by toggling the signs of the lowpass transformation coefficients which are computed using (5.31) with ω_c set to be the desired lower cutoff frequency of the bandpass filter. 144

5.8 Ideal hyperbolic contours of the migration filter family (solid curves) with the horizontal cross-section (dash-dotted line) representing the migration filter having a cutoff frequency $k_c = 1.4$. The dotted line represents the line into which the prototype filter is mapped. 148

5.9 Migration Filter Family Prototype, $N = 29$. (a) Magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Magnitude of the error (top), and alternating real rotated error (bottom). (d) Phase error in passband. 150

5.10 Comparison of the first-order subfilter contours (solid curves) with the contours of the ideal migration subfilter (5.8)(dashed curves). . . 153

5.11 Migration filter obtained using a first-order subfilter, $k_c = 0.6\pi$. (a) Magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. 154

5.12 Migration filter obtained using a first-order subfilter, $k_c = 0.8\pi$. (a) Magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. 154

5.13 Contours of the second-order subfilter (solid curves) and of the ideal subfilter (dashed curves). (a) Contours resulting from the minimax error criterion. (b) Contours resulting from the least square error criterion. 155

5.14 Passband phase responses of the desired filters (dashed curves) and of the FIR filters (solid curves) obtained by using a non-weighted second-order subfilter. Four cases are distinguished: (a) $0 < k_c \leq 0.45\pi$: obtained contours do not match ideal. (b) $0.45\pi < k_c \leq 0.67\pi$: obtained contours match ideal ones in passband region near the cutoff edge. (c) $0.67\pi < k_c \leq 0.85\pi$: obtained contours match ideal ones in most passband region. (d) $0.85\pi < k_c \leq \pi$: obtained contours match ideal ones near the line $k_2 = 0$. 156

5.15 The non-weighted second-order subfilter provides a good approximation of the continuous set of filters with $0.68\pi \leq k_c \leq 0.85\pi$. FIR filter (solid) and desired filter (dashed) passband phase responses: (a) $k_c = 0.68\pi$. (b) $k_c = 0.75\pi$. (c) $k_c = 0.8\pi$. (d) $k_c = 0.85\pi$. . . 157

5.16 3-D depth migration filter obtained using a non-weighted second-order subfilter, $k_c = 0.8\pi$. (a) Magnitude response. (b) Phase response in the passband. 158

5.17 Contours of the weighted second-order subfilter (solid curves) and of the ideal subfilter (dashed curves). (a) Contours resulting from the weighted minimax error criterion. (b) Contours resulting from the weighted least square error criterion. 159

5.18 A suitably weighted second-order subfilter providing a good approximation of the continuous set of filters with $0.3\pi \leq k_c \leq 0.75\pi$. FIR filter (solid) and desired filter (dashed) passband phase responses: (a) $k_c = 0.3\pi$. (b) $k_c = 0.38\pi$. (c) $k_c = 0.68\pi$. (d) $k_c = 0.75\pi$. 160

5.19 One- and two-dimensional migration filters obtained using a weighted second-order subfilter, $k_c = 0.4\pi$. (a) Magnitude response of the 1-D migration filter. (b) Designed (solid) and desired (dashed) passband phase responses of the 1-D migration filter. (c) Magnitude response of the 2-D migration filter. (d) Phase response of the 2-D migration filter in the passband. 161

5.20 Lowpass Filter Family Prototype, $N = 29$. (a) Magnitude response in dB. (b) Magnitude of the error (top), and alternating real rotated error (bottom). 162

5.21 Frequency response of the designed 1-D, zero-phase, variable-cutoff filter shown for different cutoff frequencies. (a) $\omega_c = 0.1\pi$. (b) $\omega_c = 0.25\pi$. (c) $\omega_c = 0.5\pi$. (d) $\omega_c = 0.75\pi$. 164

5.22 Frequency response of the designed 2-D, zero-phase, variable-cutoff filter shown for different cutoff frequencies. (a) $\omega_c = 0.1\pi$. (b) $\omega_c = 0.25\pi$. (c) $\omega_c = 0.5\pi$. (d) $\omega_c = 0.75\pi$. 165

C.1 Design Example: seismic migration filter, $N = 31$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Magnitude of the weighted subset optimal error (top), and magnitude of the weighted optimal error (bottom). 187

Summary

Algorithms are developed for the efficient design and realization of finite-duration impulse response (FIR) filters with arbitrary magnitude and phase responses. The emphasis is on minimizing and controlling the Chebyshev error norm. For that purpose, new practical characterization theorems and optimality criteria for complex Chebyshev approximation are presented and proved. In particular, the alternation theorem of Chebyshev approximation is extended from the real-only to the complex case.

In the case of one-dimensional filters, it was shown how to reformulate the complex filter design problem so that the second Remez exchange algorithm can be used in computing the optimal Chebyshev approximation to the desired complex-valued frequency response. As a result, a very flexible and efficient multiple-exchange algorithm is presented for designing optimal one-dimensional FIR filters with arbitrary magnitude and phase specifications and, more generally, for computing the best Chebyshev approximation of a complex-valued function. The new algorithm can design both causal and non-causal FIR filters with complex- or real-valued impulse responses. It was also shown how to modify the proposed algorithm for the design of optimal multi-dimensional FIR filters with complex or real frequency responses.

The applicability and limitations of the McClellan transformation method are discussed. It is shown how new types of one-dimensional filters can be transformed into two-dimensional filters, and how all the possible types of symmetries can be designed by the transformation method. The result is that very efficient transformation procedures are developed for the design of complex or real, positive- or negative-

symmetric, two-dimensional filters with a rectangular region of support having either odd- or even-length sides.

Using transformation functions, it is shown how a complete set of filters can be designed by regarding it as a single multi-dimensional filter. Consequently, a very efficient technique is presented for designing a complete set of one- and multi-dimensional filters by transforming a single one-dimensional prototype filter. The new design technique is very fast since it reduces the design of a large set of one- and multi-dimensional filters to the design of a single one-dimensional filter and, possibly, a low-order two-dimensional transformation function. It is shown how the designed filters can be realized efficiently without having to explicitly compute the coefficients of the separate filters in the set. Among the types of filters which can be designed by this transformation technique are depth migration filters and linear-phase filters.

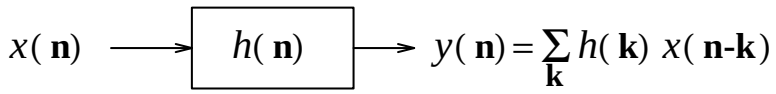
CHAPTER 1

Introduction

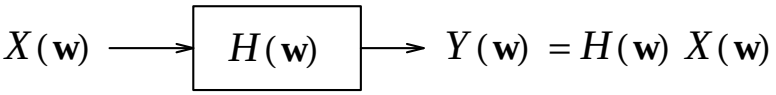
1.1 Overview

Digital signal processing is mainly concerned with the transformation and manipulation of signals and the information they contain. This processing is primarily performed using filtering operators called digital filters. In particular, linear time-invariant digital filters (Fig. 1.1) are widely used for processing all types of digital signals. These filters can be classified into two main categories: finite-duration impulse response (FIR) filters, and infinite-duration impulse response (IIR) filters [1]. Only FIR filters are considered in this thesis.

FIR digital filters possess features that make them attractive for signal processing applications. Among those features are the absence of any stability problems, the possibility of achieving an exact linear-phase, and the efficient realization by hardware or software [2, 3]. This thesis explores the theory of Chebyshev approximation and its application to designing FIR filters, with arbitrary magnitude and phase specifications. The major goal of this thesis is to develop efficient algorithms for the design of complex-valued FIR filters with emphasis on minimizing the Chebyshev norm of the complex approximation error. Complex FIR filters optimum in the Chebyshev sense find a wide range of applications, including seismic migration [4], nearly linear-phase filtering [5], equalization [6], and beamforming [7]. While FIR filters are attractive due to their stability and efficient realization, the Chebyshev norm guarantees that the approximation error will not exceed an optimal deviation (Fig. 1.2). This feature is very desirable, especially in applications where large fluctuations cannot be tolerated.

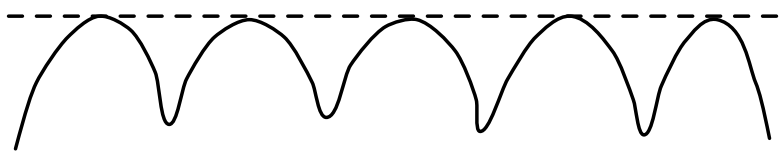


(a) Time or spatial domain.

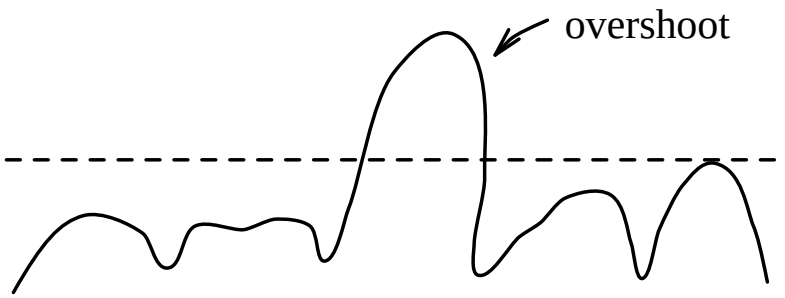


(b) Frequency domain.

Figure 1.1. A filter is identified by its impulse response $h(\mathbf{n})$ or by its frequency response $H(\mathbf{w})$ which is the Discrete-Time Fourier Transform of $h(\mathbf{n})$ [1]. (a) Input-output relation of a linear time-invariant filter in the time domain; the output is the linear convolution of the input with $h(\mathbf{n})$. (b) Input-output relation of a linear time-invariant filter in the frequency domain; $X(\mathbf{w})$ and $Y(\mathbf{w})$ are the Discrete-Time Fourier Transforms [1] of the input $x(\mathbf{n})$ and the output $y(\mathbf{n})$, respectively.



(a) Chebyshev



(b) Least Square

Figure 1.2. Comparison of the Chebyshev error (top) with the least square error (bottom). While the least square error may randomly fluctuate, the magnitude of the Chebyshev error does not exceed an optimal deviation.

The frequency response $H(\omega)$ of a length–N one-dimensional FIR digital filter is in general a complex-valued function of the normalized frequency ω:

$$H(\omega) = \sum_{n=N_1}^{N_2} h(n)e^{-j\omega n}, \qquad N = N_2 - N_1 + 1. \tag{1.1}$$

$\{h(n)\}_{n=N_1}^{N_2}$ in (1.1) represent the filter coefficients (or the filter impulse response) and are generally complex numbers. The complex one-dimensional Chebyshev filter design problem may be briefly stated as follows. Let $D(\omega)$ be a desired continuous complex function defined and continuous on a compact subset $B \subset [-\pi, \pi)$. $D(\omega)$ is to be approximated on B by the frequency response (1.1). The approximation problem consists in finding the filter coefficients $\{h(n)\}_{n=N_1}^{N_2}$ that will minimize the Chebyshev norm

$$\|E(\omega)\| = \max_{\omega \in B} |E(\omega)| \tag{1.2}$$

of the weighted error

$$E(\omega) = W(\omega)[D(\omega) - H(\omega)] \tag{1.3}$$

among all possible choices of $\{h(n)\}_{n=N_1}^{N_2}$. The weighting $W(\omega)$ in (1.3) must be a real, strictly positive and continuous function on B. The same problem formulation applies in the m-dimensional case except that the scalar variables n and ω are replaced by the m-dimensional vectors $\mathbf{n}$ and $\mathbf{w}$, respectively, and B becomes a compact set in the m-dimensional Euclidean space.

Several algorithms have been proposed for designing one-dimensional complex FIR filters in the Chebyshev sense. Cuthbert [8] introduced a design algorithm for non-linear phase FIR filters based on separately approximating the real and imaginary parts of the filter frequency response. Holt et al. [9] followed Cuthbert's design approach but used an iterative procedure based upon the classical Remez exchange algorithm [10, 11] for the two real approximations. Barrodale et al. [12] adapted a linear-programming method to approximate the real and imaginary parts simultaneously. Approximating the real and imaginary parts separately does not, in general, give the optimal solution for the original Chebyshev approximation problem [2, 13].

The deviation could be as much as -3 dB from the true best approximation. Steiglitz [13] approximated the Chebyshev FIR filter design by a linear-programming problem with a specified set of constraints on the magnitude and phase. Glashoff and Roleff [14] restated the complex Chebyshev approximation problem as a linear-programming primal problem in the presence of an infinite number of constraints. They also formulated the dual problem [15] with only a finite number of constraints. An approximate discretized method was presented to solve the primal problem. Independently, Streit and Nuttall [7] introduced a design algorithm which solves the discretized version of the primal problem formulated in [14]. Chen and Parks [5] adapted the approximation method proposed by Glashoff et al. [14] and Nuttall et al. [7] for the complex FIR filter design problem. This method is computationally intensive and solves only an approximate discretized version of the original problem. Tang [16] introduced a single exchange algorithm for the complex Chebyshev problem. His algorithm basically solves the *dual* problem formulated in [14], but only with real parameters. Alkhairy et al. [17] and Schulist [18] adapted Tang's algorithm for the design of FIR filters with real coefficients. Tang's algorithm for the dual problem led to significant improvements in terms of accuracy, speed, and storage requirements, compared to the algorithm proposed for the primal problem in [7, 5].

Most of the aforementioned FIR filter design algorithms are for the design of complex FIR filters with *real-valued* coefficients [8, 9, 13, 5, 17]. This restriction forces the frequency response of the filter to be conjugate symmetric. However, in many applications, the filter coefficients are required to be complex-valued. Such applications include seismic migration, and simultaneous magnitude–phase equalization. Preuss [6] introduced a heuristic simultaneous exchange algorithm for the design of FIR filters with complex-valued coefficients. This algorithm was improved by Schulist [19]. The algorithm by Preuss and Schulist does not always converge [17]. Even when convergence is achieved, the optimality of the solution cannot be determined. Watson [20, 21] introduced an efficient method for solving the dual problem

considered in [14, 16, 17]. Burnside and Parks [22] adapted Watson's method for the complex filter design problem. Their algorithm can handle filters with complex coefficients and has a significantly higher speed of convergence, compared to Tang's algorithm [16, 17]. Iterative reweighted least square (IRLS) error algorithms have been also proposed for the complex Chebyshev filter design problem [23, 24, 25]. These methods are based on Lawson's algorithm which exhibits a slow convergence compared to the other methods mentioned above. Recently, Tseng [26] introduced a relatively efficient implementation of Lawson's algorithm and applied it to the design of complex Chebyshev FIR filters. Potchinkov and Reemtsen [27, 28] formulated and solved the complex Chebyshev filter design problem as a quadratic program. Quadratic programming is generally more computationally expensive than linear programming; however, Potchinkov and Reemtsen reported that a high accuracy can be achieved and that long filters can be designed with this approach.

None of the previous algorithms exploits the characteristics of the complex Chebyshev optimal solutions. Algorithms based on the characterization of optimal solutions, such as the Parks-McClellan algorithm for the one-dimensional real case [29], have proven to be significantly much more efficient than those based on general optimization techniques. One goal of this thesis is to fill in gaps in the theory of Chebyshev approximation and devise such an algorithm for the complex Chebyshev approximation problem. Another main goal of this thesis is to develop efficient novel techniques for the efficient design of multi-dimensional complex FIR filters.

1.2 Organization of the Thesis

This thesis can be divided into three main parts. In the first part, the theory of Chebyshev approximation is reviewed and extended. In the second part, algorithms are developed for computing the optimal Chebyshev approximation of an arbitrary complex-valued function or filter frequency response. In the third part, new tech-

niques are developed for the efficient design and realization of complex FIR filters using transformations.

Chapter 2 is concerned with the theory of Chebyshev approximation. First the general Chebyshev approximation problem is formulated. Existing optimality criteria for complex Chebyshev approximation are then reviewed. These existing optimality criteria are mainly of theoretical interest and do not provide a practical means to compute the optimal Chebyshev approximation. So, new characterization theorems and optimality criteria, which are of theoretical as well as of practical interest, are presented and proved.

In Chapter 3, the filter design problem is reformulated so that it clearly satisfies some important properties identified in Chapter 2. Then, using the theoretical results derived in Chapter 2, it is shown that the very efficient second Remez exchange algorithm [10, 30, 11] can be used in computing the optimal complex Chebyshev approximation. As a result, a robust and efficient multiple-exchange ascent algorithm is presented for designing optimal one-dimensional FIR filters with arbitrary magnitude and phase responses, and, more generally, for computing the best Chebyshev approximation of a complex-valued function. The new approximation algorithm obtains the optimal Chebyshev solution in two stages. The first stage consists of a very efficient multiple-exchange ascent algorithm which is a generalization of the second Remez (Parks-McClellan) algorithm. The second stage consists of a robust multiple-exchange ascent-descent algorithm which makes use of a descent technique suitably adapted for minimizing the non-differentiable Chebyshev error norm. In many cases, the very efficient complex Remez method (first stage) converges to the optimal solution and the algorithm terminates without proceeding into the second stage. In the general case, the complex Remez method converges to the optimal Chebyshev solution over only a subset of the specified domain of approximation. The ascent-descent method is then used to obtain the optimal solution on the desired approximation domain. It is also shown how the ascent-descent algorithm can be extended to solve

the complex multi-dimensional Chebyshev problem. Design examples are presented at the end of Chapter 3 as well as a comparison with other existing algorithms.

The design of multi-dimensional filters using transformations is the topic of Chapter 4. This chapter studies the applicability and limitations of the transformation design methods. One purpose of Chapter 4 is to identify which one-dimensional prototype filters can be transformed, what type of multi-dimensional filters can be designed, and what types of positive and negative symmetries can be achieved. Another purpose of Chapter 4 is to present design methodologies and procedures showing how the identified admissible prototypes can be transformed and how the identified possible symmetries can be obtained. The new presented procedures are used to design complex and real, positive- and negative-symmetric, two-dimensional filters with a rectangular region of support having either odd- or even-length sides.

Chapter 5 introduces the concept of designing a family of filters by regarding it as a single multi-dimensional filter. A new efficient procedure which reduces the design of a large set of filters to the design of a single one-dimensional filter is presented and applied to the design of depth migration filters and variable-cutoff linear-phase filters. It is also shown how the designed one- and multi-dimensional filters can be realized efficiently without having to compute the coefficients of the separate filters in the set.

Chapter 6 consists of the conclusion. There the major contributions of this thesis are highlighted and possible directions for future research are discussed.

Four appendices follow Chapter 6. Appendix A highlights important notations and terms used in the chapters of this thesis. Other notations and terminologies are defined as needed in Chapters 2–5. Appendices B and C contain mathematical proofs and derivations as well as other details that complement the material presented in Chapters 2–5. Appendix D lists the MATLAB scripts of the design algorithms presented in Chapters 3 and 5. Appendix E lists the filter coefficients corresponding to some of the filters designed using the methods presented in this thesis.

CHAPTER 2

Complex Chebyshev Approximation

Complex FIR filter design is a special case of complex Chebyshev approximation, where the approximating basis functions are taken to be complex exponentials and where the approximation domain is defined to be a normalized frequency subset $\mathcal{F} \subseteq [-0.5, 0.5)$. The discussion and theoretical results of this chapter are not restricted to the filter design problem but apply to more general complex Chebyshev approximation problems.

2.1 Formulation of the Chebyshev Approximation Problem

The general complex Chebyshev approximation problem can be stated as follows. Let B be a compact (multi- or one-dimensional) space and let $C(B)$ be the linear space of continuous complex-valued functions $f(\mathbf{x})$ on B (i.e. $\mathbf{x} \in B$). Let V_n be the finite n-dimensional subspace of $C(B)$ consisting of the functions

$$\phi(\mathbf{x}) = \sum_{i=1}^{n} c_i \phi_i(\mathbf{x}) \tag{2.1}$$

where $\{\phi_i(\mathbf{x})\}_{i=1}^{n}$ are the complex basis functions spanning V_n and $\{c_i\}$ are *complex* coefficients. In addition, let $\varepsilon(\mathbf{x}) = f(\mathbf{x}) - \phi(\mathbf{x})$ denote the approximation error. For a given function $f \in C(B)$, the best Chebyshev approximation of $f(x)$ on B with respect to V_n is the element $\phi_o \in V_n$ such that

$$\| \underbrace{f - \phi_o}_{\varepsilon_o} \| \leq \| \underbrace{f - \phi}_{\varepsilon} \| \tag{2.2}$$

for all $\phi \in V_n$. The symbol $\|.\|$ in (2.2) denotes the Chebyshev norm (Appendix A). The error norm can include a real and strictly positive weighting function $W \in C(B)$ by simply replacing $f(\mathbf{x})$ with $W(\mathbf{x})f(\mathbf{x})$ and $\{\phi_i(\mathbf{x})\}_{i=1}^n$ with $\{W(\mathbf{x})\phi_i(\mathbf{x})\}_{i=1}^n$.

An important set in Chebyshev approximation is the set $\mathcal{X}_\varepsilon \subset B$ of extremal points, which consists of the points at which the error function $\varepsilon(\mathbf{x})$ achieves its maximum magnitude (Chebyshev norm), i.e,

$$\mathcal{X}_\varepsilon = \{\mathbf{x} \in B : \ |\varepsilon(\mathbf{x})| = \|\varepsilon\|\}. \tag{2.3}$$

$\mathcal{X}_o$ will be used to denote the set of extremal points of the optimal error $\varepsilon_o(\mathbf{x}) = f(\mathbf{x}) - \phi_o(\mathbf{x})$. The best Chebyshev approximation $\phi_o(\mathbf{x})$ of $f(\mathbf{x})$ on B is also the best Chebyshev approximation of $f(\mathbf{x})$ on the finite set $\mathcal{X}_o$. So, the approximation problem reduces to finding the optimal solution on a finite set of points. However, the needed extremal point set $\mathcal{X}_o$ is not known a-priori.

2.2 Existing Optimality Criteria

Basic theorems that characterize best Chebyshev approximations are presented in this section. Most of the mathematical details, which have been omitted, can be found in the excellent presentations of Meinardus [31], Lorentz [32], Rivlin and Shapiro [33], and Rice [34]. The following fundamental theorem contains the Kolmogoroff criterion for a best Chebyshev approximation [31, 32].

Theorem 2.1 (Kolmogoroff) A function $\phi_o(\mathbf{x}) \in V_n$ is a best Chebyshev approximation of $f(\mathbf{x}) \in C(B)$ if and only if the inequality

$$\min_{\mathbf{x} \in \mathcal{X}_o} \Re e\{\overline{(f(\mathbf{x}) - \phi_o(\mathbf{x}))}\phi(\mathbf{x})\} \leq 0 \tag{2.4}$$

or, equivalently,

$$\max_{\mathbf{x} \in \mathcal{X}_o} \Re e\{\overline{(f(\mathbf{x}) - \phi_o(\mathbf{x}))}\phi(\mathbf{x})\} \geq 0 \tag{2.5}$$

holds for every function $\phi(\mathbf{x}) \in V_n$, where $\mathcal{X}_o$ is the set of extremal points of $\varepsilon_o(x) = f(\mathbf{x}) - \phi_o(\mathbf{x})$ in B. □

For real-valued functions, criterion (2.4) says that $f(\mathbf{x}) - \phi_o(\mathbf{x})$ and $\phi(\mathbf{x})$ cannot possess the same signs for all $\mathbf{x} \in \mathcal{X}_o$, while criterion (2.5) says that $f(\mathbf{x}) - \phi_o(\mathbf{x})$ and $\phi(\mathbf{x})$ cannot be of opposite signs for all $\mathbf{x} \in \mathcal{X}_o$.

Convexity considerations lead to another form of Theorem 2.1 [32, 33]. Some basic definitions and useful properties of a convex set are given first [32, pp. 16–23].

Definition 2.1 (Convex Set) A subset S of an m-dimensional Euclidean space E_m is convex if, with every two points $\mathbf{p}_1$, $\mathbf{p}_2 \in S$, it contains the segment consisting of all points $\alpha\mathbf{p}_1 + \beta\mathbf{p}_2$ for which α and β are real positive numbers and $\alpha + \beta = 1$. □

The above definition applies also in the m-dimensional complex space E_m which is isomorphic to the $2m$-dimensional, real Euclidean space.

For each (not necessarily convex) set $S \subset E_m$, there exists a *smallest* convex set S_c that contains S; this set S_c is called the *convex hull* of S. Furthermore, the convex hull of a compact set is compact. The following lemma will be useful [32, pp. 16–23].

Lemma 2.1 The convex hull of a set $S \subset E_m$ consists of all the points $\mathbf{p}$ of the form

$$\mathbf{p} = \sum_{i=1}^{r} w_i \mathbf{p}_i, \quad \mathbf{p}_i \in S,\ w_i > 0,\ i = 1, \ldots, r;\ \sum_{i=1}^{r} w_i = 1, \tag{2.6}$$

where $r \leq 2m + 1$ in the complex case and $r \leq m + 1$ in the real case. □

A new form of Theorem 2.1 is now presented [33, 32].

Theorem 2.2 (Rivlin & Shapiro) A function $\phi_o(\mathbf{x}) \in V_n$ is a best Chebyshev approximation of $f(\mathbf{x}) \in C(B)$ if and only if the origin in E_n belongs to the convex hull of the points

$$z_{\mathbf{x}} = ((f(\mathbf{x}) - \phi_o(\mathbf{x}))\overline{\phi_1(\mathbf{x})}, \ldots, (f(\mathbf{x}) - \phi_o(\mathbf{x}))\overline{\phi_n(\mathbf{x})}), \quad \mathbf{x} \in \mathcal{X}_o \tag{2.7}$$

where $\mathcal{X}_o$ is the set of extremal points of $f(\mathbf{x}) - \phi_o(\mathbf{x})$ in B, and $\mathbf{x}$ runs over all points of $\mathcal{X}_o$. □

The convex hull criterion in Theorem 2.2 is equivalent to the Kolmogoroff criterion given by (2.4–2.5) (consult [32, p. 21] for the proof). Combining Theorem 2.2 with Lemma 2.1, another equivalent characterization is obtained

Theorem 2.3 A function $\phi_o(\mathbf{x}) \in V_n$ is a best Chebyshev approximation to $f(\mathbf{x}) \in C(B)$ if and only if there are r points $\mathbf{x}_1, \ldots, \mathbf{x}_r \in \mathcal{X}_o$ and r positive numbers $w_1, \ldots, w_r$, $\sum w_k = 1$, $r \leq 2n+1$ ($r \leq n+1$ in the real case), for which

$$\sum_{k=1}^{r} w_k [f(\mathbf{x}_k) - \phi_o(\mathbf{x}_k)] \overline{\phi_i(\mathbf{x}_k)} = 0, \quad i = 1, \ldots, n \tag{2.8}$$

or equivalently,

$$\sum_{k=1}^{r} w_k [f(\mathbf{x}_k) - \phi_o(\mathbf{x}_k)] \overline{\phi(\mathbf{x}_k)} = 0, \quad \text{for all } \phi(\mathbf{x}) \in V_n \tag{2.9}$$

where $\mathcal{X}_o$ is the set of extremal points of $f(\mathbf{x}) - \phi_o(\mathbf{x})$ in B. □

Rivlin & Shapiro [33] introduced the useful notion of extremal signature and used it to obtain a characterization of optimal Chebyshev approximations. Before presenting this characterization, some definitions are needed [32].

Definition 2.2 (Finite Support) A function $\mu(\mathbf{x})$ has *finite support* $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_r\} \subseteq B$, where $\{\mathbf{x}_i\}_{i=1}^{r}$ are distinct points in B, if $\mu(\mathbf{x}) \neq 0$ on S and $\mu(\mathbf{x}) = 0$ otherwise. □

Definition 2.3 (Extremal Signature) A *signature* $\sigma(\mathbf{x})$ is a function of finite support $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_r\} \subseteq B$ whose values $\sigma(\mathbf{x}_k)$ on S are (complex or real) signs, i.e., $|\sigma(\mathbf{x}_k)| = 1$. $\sigma(\mathbf{x})$ is an *extremal signature* with respect to V_n if there exists a function μ with support S such that

$$\begin{cases} \operatorname{sign}\{\mu(\mathbf{x}_k)\} &= \sigma(\mathbf{x}_k), \quad k = 1, \ldots, r \\ \displaystyle\sum_{k=1}^{r} \mu(\mathbf{x}_k) \overline{\phi(\mathbf{x}_k)} &= 0, \quad \text{for all } \phi(\mathbf{x}) \in V_n \end{cases} \tag{2.10}$$

□

Now, using the convenient notion of extremal signature, Theorem 2.3 can be restated in a simpler form [32, Chapter 2].

Theorem 2.4 (Rivlin & Shapiro) A function $\phi_o(\mathbf{x}) \in V_n$ is a best Chebyshev approximation to $f(\mathbf{x}) \in C(B)$ if and only if there exists an extremal signature $\sigma(\mathbf{x})$ with support $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_r\} \subset \mathcal{X}_o$ for which $r \leq 2n+1$ ($r \leq n+1$ in the real case) and

$$\operatorname{sign}\{f(\mathbf{x}_k) - \phi_o(\mathbf{x}_k)\} = \sigma(\mathbf{x}_k), \quad k = 1, \ldots, r. \tag{2.11}$$

where $\mathcal{X}_o$ is the set of extremal points of $f(\mathbf{x}) - \phi_o(\mathbf{x})$ in B. □

Important corollaries to the foregoing theorems follow [32, 34].

Theorem 2.5 If $\phi_o(\mathbf{x})$ is a best Chebyshev approximation of $f(\mathbf{x})$ on B, then $\phi_o(\mathbf{x})$ is also a best Chebyshev approximation of $f(\mathbf{x})$ on a certain finite subset $B_s \subset B$, which consists of $r \leq 2n+1$ ($r \leq n+1$ in the real case) points. □

Theorem 2.6 Let $\phi_o(\mathbf{x})$ be a best approximation of $f(\mathbf{x})$ on B. If the extremal set $\mathcal{X}_o$ consists of r points, then $\phi_o(\mathbf{x})$ is an approximation of maximum error norm among all best approximations to $f(\mathbf{x})$ on subsets $B_s \subset B$ containing r points. □

Bounded linear functionals are very useful tools in problems of approximation and minimization [35, chapter 5].

Definition 2.4 A linear functional L on a normed space $C(B)$ is *bounded* if there is a constant M such that $|L(f)| \leq M\|f\|$ for all $f \in C(B)$. The smallest such constant M is called the norm of L and is denoted $\|L\|$. Thus,

$$\|L\| = \inf_M \{M : |L(f)| \leq M\|f\|, \text{all } f \in C(B)\} = \sup_{\|f\| \leq 1} \{|L(f)|, f \in C(B)\}. \tag{2.12}$$

□

The following result can be deduced from the characterization theorems 2.1–2.4 [33, 35] and gives a lower bound on the optimal Chebyshev error norm:

Theorem 2.7 (Duality Theorem) If $f(\mathbf{x}) \in C(B)$ and

$$m = \|f(\mathbf{x}) - \phi_o(\mathbf{x})\| = \min \|f(\mathbf{x}) - \phi(\mathbf{x})\|, \quad \phi_o \text{ and } \phi \in V_n, \tag{2.13}$$

then m equals the maximum of $|L(f)|$ over all bounded linear functionals having norm one (i.e., $\|L\| = 1$) and vanishing on V_n (i.e., $L(\phi) = 0$ for $\phi(\mathbf{x}) \in V_n$), the maximum being attained for some functional L^*. □

A corollary to Theorem 2.7 is the following result which gives a way to construct any linear functional on a finite-dimensional normed space [33]:

Theorem 2.8 Let L be any linear functional on the space $V'_{n+1} = \{f(\mathbf{x}) \cup V_n\}$ of $C(B)$. Then there exist points $\mathbf{x}_1, \ldots, \mathbf{x}_r$ of B, $r \leq n+1$ in the real case, $r \leq 2n+1$ in the complex case, and constants $\lambda_1, \ldots, \lambda_r$ such that

$$\begin{cases} L = \sum_{k=1}^{r} \lambda_k f(\mathbf{x}_k) \\ \sum_{i=1}^{r} |\lambda_k| = \|L\| \end{cases} \tag{2.14}$$

where $\lambda_k \neq 0$ $(k = 1, \ldots, r)$. □

All the above theorems present a characterization of the best Chebyshev approximation but do not provide a means to compute the optimal solutions. In the remainder of this chapter, new results and criteria are derived for computing optimal Chebyshev solutions.

2.3 Solution on Finite Subsets of Points in a Multi-dimensional Space

Theorem 2.5 and 2.7 indicate that a main strategy in Chebyshev approximation is to work on finite subsets B_s of the desired set B and relate the optimal error on B_s to the optimal error on B. The norm of the optimal error on B_s will always be a lower bound to the Chebyshev error norm on B. This section derives the equations for computing optimal Chebyshev approximations in V_n of $f(\mathbf{x})$ on finite sets consisting of r $(\geq n+1)$ points in the m-dimensional (complex or real) space E_m.

Consider the finite set $B_s = \{\mathbf{x}_1, \ldots, \mathbf{x}_r\}$, consisting of the distinct points $\mathbf{x}_k = (x_{k1}, x_{k2}, \ldots, x_{km}) \in E_m$, $k = 1, \ldots, r$. Consider the bounded linear functional

$$L(f) = \sum_{k=1}^{r} \lambda_k f(\mathbf{x}_k) \tag{2.15}$$

normalized such that

$$\begin{cases} L(\phi_j) = \sum_{k=1}^{r} \lambda_k \phi_j(\mathbf{x}_k) = 0, & j = 1, \ldots, n \\ \|L\| = \sum_{k=1}^{r} |\lambda_k| = 1 \end{cases} \tag{2.16}$$

The requirement (2.16) implies that $L(f)$ vanishes on V_n and has norm one. It also implies that not all the parameters $\{\lambda_k\}$ can be zero.

Now, construct the function $\phi_s(\mathbf{x}) = \sum_{j=1}^{n} c_j \phi_j(\mathbf{x})$ by solving the following system of equations

$$\begin{cases} |\lambda_k| \sum_{j=1}^{n} c_j \phi_j(\mathbf{x}_k) + \overline{\lambda_k}\delta = |\lambda_k| f(\mathbf{x}_k), \quad k = 1, \ldots, r & (2.17) \\ \sum_{k=1}^{r} \lambda_k \phi_j(\mathbf{x}_k) = 0, \quad j = 1, \ldots, n & (2.18) \\ \sum_{k=1}^{r} |\lambda_k| = 1 & (2.19) \end{cases}$$

consisting of $r + n + 1$ equations with the $r + n + 1$ unknowns $\{\{\lambda_k\}_{k=1}^{r}; \{c_j\}_{j=1}^{n}; \delta\}$. $\overline{\lambda_k}$ in (2.17) denotes the complex conjugate of λ_k.

From (2.17) and (2.18), it can also be shown that

$$\delta = \sum_{k=1}^{r} \lambda_k f(\mathbf{x}_k) = L(f). \tag{2.20}$$

Note that, for any $\lambda_k \neq 0$, (2.17) can be rewritten as

$$\sum_{j=1}^{n} c_j \phi_j(\mathbf{x}_k) + \frac{\overline{\lambda_k}}{|\lambda_k|}\delta = f(\mathbf{x}_k) \tag{2.21}$$

which gives

$$|f(\mathbf{x}_{k_o}) - \sum_{j=1}^{n} c_j \phi_j(\mathbf{x}_{k_o})| = |\delta|. \tag{2.22}$$

In order for $\phi_s(\mathbf{x})$ to be an optimal solution on B_s (refer to proof of Theorem 2.9), the following additional constraint need to be applied only if some of the parameters $\{\lambda_k\}$, which are obtained by solving (2.17–2.19), are zero:

$$|f(\mathbf{x}_k) - \sum_{j=1}^{n} c_j \phi_j(\mathbf{x}_k)| \leq |\delta|, \quad k = 1, \ldots, r. \tag{2.23}$$

The following theorem gives a way to determine the best approximation $\phi_s(\mathbf{x}) \in V_n$ of $f(\mathbf{x})$ on B_s.

Theorem 2.9 A function $\phi_s(\mathbf{x}) = \sum_{j=1}^{n} c_j \phi_j(\mathbf{x})$, where $\{c_j\}_{j=1}^{n}$ solve (2.17–2.19) and (2.23), is a best Chebyshev approximation of $f(\mathbf{x})$ with respect to V_n on the set B_s.

□

Proof: Without loss of generality, let $\{\lambda_k\}_{k=1}^{r_o} \neq 0$ ($r_o \leq r$) by reordering the points $\{\mathbf{x}_k\}$. Thus, from (2.17) and (2.23),

$$|f(\mathbf{x}_k) - \phi_s(\mathbf{x}_k)| = |\delta|, \quad k = 1, \ldots, r_o \tag{2.24}$$

$$|f(\mathbf{x}_k) - \phi_s(\mathbf{x}_k)| \leq |\delta|, \quad k = r_o + 1, \ldots, r \tag{2.25}$$

So, $\{\mathbf{x}_1, \ldots, \mathbf{x}_{r_o}\} \subseteq \mathcal{X}_o$, where $\mathcal{X}_o$ is the set of extremal points of $f(\mathbf{x}) - \phi_s(\mathbf{x})$ on B_s, and $|\delta|$ is the norm of the optimal Chebyshev error on B_s. In what follows, the proof is similar to the one used by Meinardus [31, p. 19] for the more restricted case with $r = n + 1$. If the function ϕ_s were not a best approximation of f on B_s, then by Theorem (2.1) there would exist a function $\phi \in V_n$ such that

$$\Re e\{\overline{(f(\mathbf{x}_k) - \phi_s(\mathbf{x}_k))}\phi(\mathbf{x}_k)\} > 0, \quad \text{for all } \mathbf{x}_k \in \mathcal{X}_o. \tag{2.26}$$

Inequality (2.26) and (2.21) imply that

$$\Re e\{\lambda_k \hat{\phi}(\mathbf{x}_k)\} > 0, \quad k = 1, \ldots, r_o \tag{2.27}$$

where $\hat{\phi} = \overline{\delta}\phi$ and $\hat{\phi} \in V_n$. It follows that

$$\Re e\{L(\hat{\phi})\} > 0 \tag{2.28}$$

and thus

$$L(\hat{\phi}) \neq 0 \tag{2.29}$$

in contradiction with (2.18). □

Note that several optimal solutions may be possible. A unique optimal solution results if the basis functions $\{\phi_j\}_{j=1}^n$ satisfy the Haar condition [32, 34].

Definition 2.5 (Haar Condition) A set of n functions $\{\phi_j\}_{j=1}^n$ is said to satisfy the Haar condition on B if every non-trivial linear combination of these functions vanishes at no more than $n-1$ points of B. □

Another consequence of the Haar condition is that the set of extremal points $\mathcal{X}_o$ consists of at least $n+1$ distinct points [31]. The case where the Haar condition is satisfied and B_s consists of *exactly* $n+1$ points is important because, in that case, the above (constrained) nonlinear system of equations reduces to two independent linear system of equations with no constraints. The optimal solution can then be uniquely determined by simply solving two linear system of equations as it was also shown by Meinardus [31]. In fact, the best approximation ϕ_s of f with respect to V_n (satisfying the Haar condition) on any set B_s of $n+1$ points $\{\mathbf{x}_k\}_{i=1}^{n+1}$ can be constructed as follows [31]:

$$f(\mathbf{x}_k) - \phi_s(\mathbf{x}_k) = \delta \frac{\overline{\lambda_k}}{|\lambda_k|} = \delta e^{j\theta_k}, \quad k = 1, \ldots, n+1 \tag{2.30}$$

where $\overline{\lambda}$ denotes the complex conjugate of λ, and the λ_k are first computed by solving

$$\begin{cases} \sum_{k=1}^{n+1} \lambda_k \phi_j(\mathbf{x}_i) &= 0, \quad j = 1, \ldots, n \\ \sum_{k=1}^{n+1} |\lambda_k| &= 1, \quad \lambda_1 \text{ real and } > 0 \end{cases} \tag{2.31}$$

Without loss of generality, λ_1 can be taken to be real and positive since its phase can be absorbed into δ [31, p. 18]. The first equation in (2.31) is a system of n equations in $n+1$ unknowns. The Haar condition implies that its solution space is rank one [30, 31], and this solution can be scaled according to the second equation in (2.31). It follows that the λ_k are uniquely determined by virtue of the Haar

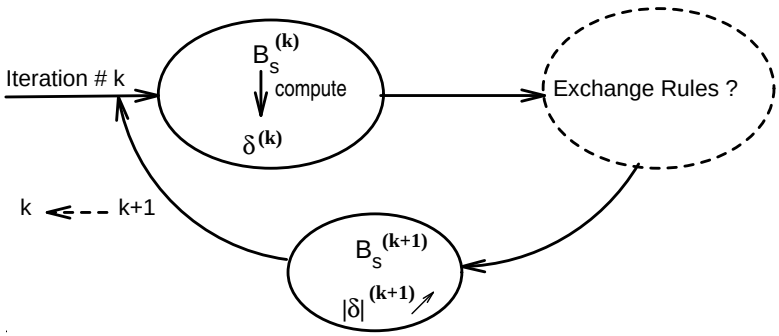


Figure 2.1. Ascent exchange strategy.

condition and that $\lambda_k \neq 0$, $k = 1, \ldots, n+1$ [31]. Representing $\phi_s(\mathbf{x})$ in (2.30) as a linear combination of $\{\phi_j(\mathbf{x})\}_{j=1}^{n}$ leads to a uniquely solvable linear system with the complex basis coefficients and complex δ as unknowns.

With $\varepsilon_o(\mathbf{x})$ denoting the optimal error on B, the optimal error $|\delta|$ on $B_s \subset B$ is related to $\varepsilon_o(\mathbf{x})$ as follows [31]:

$$|\delta| \leq \|\varepsilon_o(\mathbf{x})\|. \tag{2.32}$$

The approximation problem on B is solved by finding the set B_s of $n+1$ points $\{\hat{x}_i\}_{i=1}^{n+1}$ for which $|\delta|$ equals its upper bound $\|\varepsilon_o(\mathbf{x})\|$ [31, 29]. Given an initial set B_s and its corresponding δ, the idea is to iteratively construct new sets B_s with increasing $|\delta|$. A new set is formed by exchanging points of the current set with new points taken from B, as it is illustrated by Fig. 2.1. Exchange rules need then to be derived in order to ensure an increase of $|\delta|$ at each iteration. This strategy is at the core of exchange algorithms, such as the second Remez or Parks-McClellan algorithm [30, 29] for real one-dimensional Chebyshev approximation. However, in the general complex case, it can be seen (Equation 2.30) that δ depends on the phases of the parameters $\{\lambda_k\}$ which are not known a-priori and which need to be computed for all the possible sets $B_s \subset B$. Due to this dependency, it is thus difficult to derive efficient point-exchange rules.

The following section eliminates this dependency by presenting an extension of

the alternation property [31, 34] and other related characterization theorems to the complex case.

2.4 New Simple Optimality Criteria

A simple and exploitable characterization of optimal Chebyshev solutions is very desirable since it makes possible the development of efficient algorithms for computing the best Chebyshev approximation of a function. While such a characterization does not seem possible for the general complex Chebyshev approximation problem formulated in section 2.1, some constraints are imposed resulting in a class of problems for which an efficient characterization is presented.

2.4.1 Constrained Chebyshev Approximation Problem

The constrained Chebyshev approximation problem can be stated like the unconstrained problem of section 2.1 with the following two additional constraints:

1. The set B is constrained to be a compact set of the real line (one-dimensional). So, the vector $\mathbf{x} \in B$ will be replaced by the scalar element x.

2. The basis functions $\{\phi_j\}_{j=1}^n$ of the approximating space V_n are constrained to be real-valued and to satisfy the Haar condition on B. It is also assumed that the basis functions are continuous on a real interval $[a, b]$ $(a < b)$ containing B.

Some important remarks are in order:

Remark 1 It is important to note that no constraints are imposed on the basis coefficients $\{c_j\}_{j=1}^n$ which are in general complex-valued. So, the approximating space V_n consists of complex-valued functions.

Remark 2 No constraints are imposed on the complex-valued function $f(x)$ that is to be approximated, except that its domain B should be a compact real one-dimensional space. This latter requirement follows from constraint 1.

Remark 3 The 1-D complex FIR filter design problem is shown in Chapter 3 to satisfy the constraints listed in this section.

Remark 4 With the coefficients $\{c_j\}_{j=1}^{n}$ being complex (see *Remark 1*), any set of complex basis functions can be converted into a set of real-valued basis functions by simply taking (possibly a subset of) the real and imaginary parts of the original basis functions. However, this straightforward conversion does not work in general. In fact, the reformulation of the 1-D filter design problem in terms of real-valued basis functions required factoring out a linear-phase term and expressing the filter impulse response $h(n)$ as the sum of a positive-symmetric function and a negative-symmetric function (refer to Chapter 3).

2.4.2 Alternation in the Complex case

The results of this section apply to the constrained Chebyshev problem formulated in section 2.4.1. In addition, if $f(x)$ is a real-valued function, $\{\lambda_k\}_{k=1}^{n+1}$ in (2.30) are known to be real and alternating in sign [31, 30]; so there is no need to solve (2.31) to find the phases of the λ_k's. The following lemma generalizes this result to the complex case, where $f(x)$ is an arbitrary complex function.

Lemma 2.2 Let $\{x_k\}_{k=1}^{n+1}$ be a set of $n+1$ distinct points in B arranged in increasing order. Let $\{\phi_j(x)\}_{j=1}^{n}$ be the real-valued basis functions of V_n. The parameters $\{\lambda_k\}_{k=1}^{n+1}$ given by (2.31) are real-valued and have alternating signs. □

Proof: From (2.31), it is clear that $\{\lambda_k\}_{k=1}^{n+1}$ are real because the basis functions $\{\phi_j(x)\}_{j=1}^{n}$ are real. It is still required to show that the λ_k alternate in sign. Since V_n satisfies the Haar condition and has real-valued basis functions, there exists a real-valued function $\phi(x) \in V_n$ vanishing at all the points $\{x_k\}_{k=1}^{n+1}$ with the exception of two successive points x_{k_o} and x_{k_o+1} and such that $\phi(x_{k_o}) = 1$. Let $\phi(x_{k_o+1}) = \beta$, where β is a real number. From the Haar condition, $\phi(x)$ cannot vanish at more than

$n-1$ points. Therefore, $\phi(x_{k_o})$ and $\phi(x_{k_o+1})$ must have the same sign, which implies that $\beta > 0$. From (2.18), it follows that

$$\sum_{k=1}^{n+1} \lambda_k \phi(x_k) = \lambda_{k_o} + \lambda_{k_o+1}\beta = 0$$

or

$$\lambda_{k_o} = -\beta \lambda_{k_o+1}$$

which implies that λ_{k_o} and λ_{k_o+1} have opposite signs. The lemma is proved. □

Lemma 2.2 and (2.30) lead to the following theorem.

Theorem 2.10 The best approximation on a set of $n+1$ distinct points $\{x_k\}_{k=1}^{n+1}$ has an error that assumes its maximum value at all the points with an alternating phase shift of π. □

From Theorem 2.10, it can be concluded that the best approximation ϕ_s of f with respect to V_n on any ordered set of $n+1$ points $\{x_k\}_{k=1}^{n+1}$ can be constructed as follows

$$f(x_k) - \phi_s(x_k) = (-1)^{k+1}\delta, \; k = 1, \ldots, n+1. \tag{2.33}$$

For convenience, the term *alternation* will be used to denote a phase shift of π when going from one extremal point to another in an increasing order.

The next theorem states that the alternation property is a sufficient condition to have a best approximation on B.

Theorem 2.11 Let $\phi \in V_n$. If there exist (at least) $n+1$ points $\{x_k\}_{k=1}^{n+1} \subset B$ arranged in increasing order such that the error function $\varepsilon(x) = f(x) - \phi(x)$ satisfies

$$\begin{cases} |\varepsilon(x_k)| &= \|f-\phi\|, \quad k = 1, \ldots, n+1 \\ \varepsilon(x_i) + \varepsilon(x_{i+1}) &= 0, \qquad\quad i = 1, \ldots, n \end{cases} \tag{2.34}$$

Then ϕ is the best approximation of f with respect to V_n on B. □

Proof: (2.34) states that the error alternates on $n+1$ extremal points $\{x_k\}_{k=1}^{n+1}$. Using Theorems 2.3 and 2.4, it is sufficient to prove that it is possible to find $n+1$ positive

numbers $w_1, \ldots, w_{n+1}$, $\sum_{i=1}^{n+1} w_i = 1$, such that

$$\sum_{k=1}^{n+1} w_k[f(x_k) - \phi(x_k)]\overline{\phi_j(x_k)} = 0, \quad j = 1, \ldots, n \tag{2.35}$$

where $\{\phi_j\}_{j=1}^{n}$ are the basis functions of V_n. Note that it is sufficient to find only a *subset* of the extremal points satisfying (2.35) [32, p. 22]. Multiplying both sides of the first equation in (2.31) by $\varepsilon(x_1)$, and recalling that $\lambda_k = (-1)^{k+1}|\lambda_k|$, we get

$$\sum_{k=1}^{n+1} |\lambda_k|\varepsilon(x_1)(-1)^{k+1}\phi_j(x_k) = 0, \quad j = 1, \ldots, n$$

or

$$\sum_{i=1}^{n+1} |\lambda_i|\varepsilon(x_k)\phi_j(x_k) = 0, \quad j = 1, \ldots, n$$

where $\sum_{k=1}^{n+1} |\lambda_k| = 1$ from (2.31). Setting $w_k = |\lambda_k|$ and noting that the basis functions $\{\phi_j\}_{j=1}^{n}$ are real, it can be readily seen that finding the positive numbers $\{w_k\}$ of (2.35) is equivalent to finding the coefficients $\{\lambda_k\}$ in (2.31) which exist and are unique by virtue of the Haar condition [31]. The theorem is proved. □

Theorem 2.11 is of particular practical importance. For instance, suppose ϕ is not the best approximation of f on the set B and there exists a set $B' \subset B$ on which $\varepsilon(x) = f(x) - \phi(x)$ satisfies (2.34). With $D = \{x_k\}_{k=1}^{n+1}$, $B' = \{x \in B : |\varepsilon(x)| \leq \max_{x \in D} |\varepsilon(x)|\}$. One can then conclude that $\phi(x)$ is the best approximation of $f(x)$ with respect to V_n on B'. $\phi(x)$ is refered to as the subset best approximation of $f(x)$. In addition, it can be deduced using (2.32) that

$$\sigma_1 = \max_{x \in B'} |\varepsilon(x)| \leq \|\varepsilon_o(x)\| \leq \|\varepsilon(x)\| = \sigma_2 \tag{2.36}$$

where $\|\varepsilon_o(x)\|$ denotes the optimal error on B. If $B' \approx B$ or the ratio σ_2/σ_1 is near 1, then for practical purposes the approximation $\phi(x)$ is acceptable as an excellent approximation on B [31, p. 15] [36, p. 640].

The alternation property is desirable since it makes possible the development of simple and efficient algorithms for computing the best Chebyshev approximation of

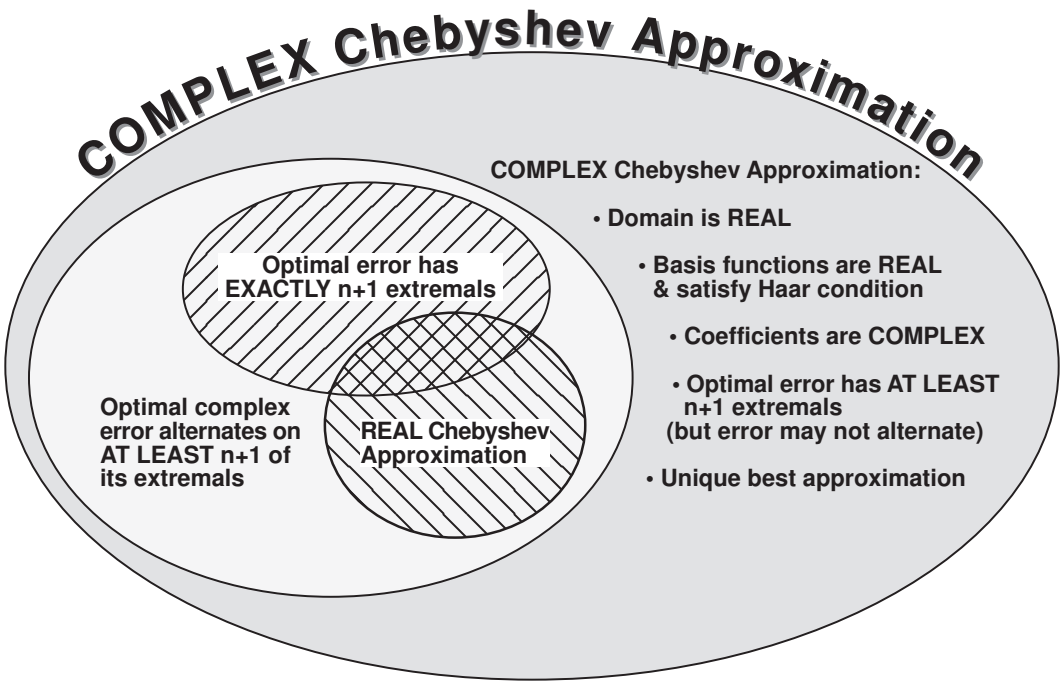


Figure 2.2. Schematic illustration of the optimal error properties. The light-shaded region represents cases where the optimal complex error alternates.

a function. However, the alternation property is not necessary for optimality in the complex case. Theorem 2.12 gives a sufficient condition under which the error of the best approximation alternates. First, a fundamental lemma is given below.

Lemma 2.3 Let $\mathcal{X}_\varepsilon$ be the set of extremal points of $\varepsilon(x) = f(x) - \phi(x)$ on B. If ϕ is the best approximation of f with respect to V_n on B, then ϕ is also the best approximation of f on the set $\mathcal{X}_\varepsilon$.

The proof of Lemma 2.3 follows directly from the Kolmogoroff Criterion (Theorem 2.1).

Theorem 2.12 Let $\varepsilon(x) = f(x) - \phi(x)$, where $\phi \in V_n$. If $\varepsilon(x)$ has exactly $n+1$ extremal points $\{x_k\}_{k=1}^{n+1}$ on B then the following equivalence holds:

$\phi(x)$ is the best approximation of $f(x)$ on B

if and only if $\varepsilon(x_k) + \varepsilon(x_{k+1}) = 0$, $k = 1, \ldots, n$. □

Proof: The "if" part (sufficiency) follows from Theorem 2.11. It is required to prove the "only if" part (necessity) under the assumption that $\varepsilon(x)$ has exactly $n+1$ points. So, given that ϕ is the best approximation, the goal is to prove that the error $\varepsilon(x)$ alternates on $\mathcal{X}_\varepsilon$, where $\mathcal{X}_\varepsilon$ is the set of extremal points $\{x_k\}_{k=1}^{n+1}$. By Lemma 2.3, ϕ is also the best approximation on $\mathcal{X}_\varepsilon$. Since $\mathcal{X}_\varepsilon$ has exactly $n+1$ points, (2.33) can be used to construct an alternating best approximation on $\mathcal{X}_\varepsilon$. It is also known that the best approximation on D is unique due to the Haar condition. The theorem is proved. □

Figure 2.2 illustrates some of the results presented in this section. The light-shaded set corresponds to Chebyshev approximations with an alternating optimal error.

2.5 Summary

The theory of Chebyshev approximation was reviewed and extended. New characterization theorems and optimality criteria were presented and proved. In particular, the alternation theorem [31, 30] was extended from the real-only to the complex case (Theorems 2.10, 2.11, and 2.12).

CHAPTER 3

Optimal Complex FIR Filter Design

This chapter presents new algorithms for designing optimal Chebyshev FIR filters with arbitrary magnitude and phase responses and, more generally, for computing the best Chebyshev approximation of an arbitrary complex-valued function. Although the emphasis is on the one-dimensional Chebyshev problem (section 2.4.1), the general multidimensional complex Chebyshev problem is also treated in this Chapter. The reformulation of the one-dimensional approximation problem in terms of real-valued basis functions and complex-valued coefficients is an important insight which is exploited in developing a new one-dimensional approximation algorithm.

3.1 Reformulation of the Filter Design Problem

The new optimality criteria derived in Chapter 2 apply to the constrained Chebyshev problem formulated in section 2.4.1. Recall that, for a constrained Chebyshev problem, the desired complex function is to be approximated on a compact set B, with respect to a subspace V_n satisfying the Haar condition on B and spanned by *real-valued* basis functions continuous on a real interval containing B. In this section, it is shown that these constraints are satisfied by the complex FIR filter design problem.

Let $D(\omega)$ be a desired complex frequency response. $D(\omega)$ is to be approximated by an FIR filter having a frequency response $H(\omega)$ and an impulse response $h(k)$, $k = N_1, \ldots, N_2$, of length $N = N_2 - N_1 + 1$. Note that, when N_1 is negative, the filter is noncausal. Rewriting $D(\omega)$ and $H(\omega)$ as

$$D(\omega) = e^{-j\frac{N_1+N_2}{2}\omega} A(\omega) \tag{3.1}$$

and

$$H(\omega) = e^{-j\frac{N_1+N_2}{2}\omega} H_{nc}(\omega) \tag{3.2}$$

the design problem reduces to the equivalent problem of approximating $A(\omega)$ by $H_{nc}(\omega)$.

- When N *is odd*,

$$H_{nc}(\omega) = \sum_{k=0}^{(N-1)/2} \alpha_k \cos k\omega \; + \sum_{k=1}^{(N-1)/2} \beta_k \sin k\omega \tag{3.3}$$

where

$$\begin{cases} \alpha_0 &= h_e(0) \\ \alpha_k &= 2h_e(k), \;\; k \neq 0 \\ \beta_k &= -2j\,h_o(k) \\ h_e(k) &= [h(N_{mid}+k) + h(N_{mid}-k)]/2 \\ h_o(k) &= [h(N_{mid}+k) - h(N_{mid}-k)]/2 \\ N_{mid} &= (N_1+N_2)/2 \end{cases}$$

For a space to satisfy the Haar condition, it is necessary and sufficient that its basis functions satisfy the Haar condition [30, p. 74]. In this case, V_n $(n = N)$ is the N-dimensional subspace spanned by the real basis functions $\{\{1\} \cup \{\cos l\omega, \sin l\omega\}_{l=1}^{(N-1)/2}\}$ which satisfy the Haar condition [30, p. 78] on any compact set $B \subset [-\pi, \pi)$. Furthermore, the basis functions are continuous on $[-\pi, \pi]$.

- When N *is even*,

$$H_{nc}(\omega) = \sum_{k=0}^{(N-2)/2} [\alpha_k \cos (k+\frac{1}{2})\omega \; + \; \beta_k \sin (k+\frac{1}{2})\omega] \tag{3.4}$$

where

$$\begin{cases} \alpha_k &= 2h_e(k) \\ \beta_k &= -2j\,h_o(k) \\ h_e(k) &= [h(N_{mid}+k) + h(N_{mid}-k-1)]/2 \\ h_o(k) &= [h(N_{mid}+k) - h(N_{mid}-k-1)]/2 \\ N_{mid} &= (N_1+N_2+1)/2 \end{cases}$$

In this case, V_n $(n = N)$ is the N-dimensional subspace spanned by the real basis functions $\{\cos(l-0.5)\omega, \sin(l-0.5)\omega\}_{l=1}^{N/2}\}$ which can be shown to satisfy the Haar condition on $B \subset [-\pi, \pi]$.

A closed interval $[-\pi, \pi]$ can be used when N is even because $H_{nc}(\omega)$ is periodic with a period equal to 4π and not 2π. It follows that the error at the points $\omega = -\pi$ and $\omega = \pi$ may not be identical (see Example 1 of section 3.5). Note that the resulting FIR filter $H(\omega)$ given by (3.2) is periodic with a period of 2π as required.

If $A(\omega)$ is (anti-)symmetric its best approximation $H_{nc}(\omega)$ is also (anti-)symmetric [31, p. 27]. In these cases, the number of required basis functions can be reduced. When $A(\omega) = A(-\omega)$, the coefficients $\{\beta_k\}$ are 0. Therefore, (3.3) and (3.4) reduce to a sum consisting of only the cos() basis functions. The subspace V_n, spanned by the cos() basis functions, satisfies the Haar condition on B where

$$\begin{cases} n = (N+1)/2, & B \subseteq [0, \pi], \quad \text{when } N \text{ odd} \\ n = N/2, & B \subset [0, \pi), \quad \text{when } N \text{ even.} \end{cases} \tag{3.5}$$

Similarly, when $A(\omega) = -A(-\omega)$, the coefficients $\{\alpha_k\}$ are 0. Therefore, (3.3) and (3.4) reduce to a sum consisting of the sin() basis functions. The subspace V_n, spanned by the sin() basis functions, satisfies the Haar condition on B where

$$\begin{cases} n = (N-1)/2, & B \subset (0, \pi), \quad \text{when } N \text{ odd} \\ n = N/2, & B \subset (0, \pi], \quad \text{when } N \text{ even.} \end{cases} \tag{3.6}$$

The above new formulation clearly shows that the complex one-dimensional filter design problem satisfies the Haar condition. Consequently, contrary to what has been previously reported in the literature [17, 27], the optimal solution is unique and no degeneracy cases should be expected.

It should be noted that the basis functions spanning the approximating space V_n change depending whether the filter length N is odd or even. Therefore, completely different approximation problems result when the filter length is increased by an

odd increment. However, increasing the filter length by an even increment simply increases the dimensionality of the space V_n. In this latter case, the number of the basis functions increases but the form of the basis is not changed. Consequently, it can be concluded that incrementing the filter length by an even number results in a smaller optimal deviation. This is not necessarily true when the filter length is incremented by an odd number.

The remainder of this chapter switches to a notation that matches the discussion and theorems of Chapter 2. Thus, $f(x)$ corresponds to $A(\omega)$, $\phi(x)$ to $H_{nc}(\omega)$, x to ω, and $\{\phi_i(x)\}$ to the appropriate set of trigonometric basis functions.

3.2 Calculation of the Best Approximation

This section presents a new exchange ascent algorithm for computing the best Chebyshev approximation of an arbitrary complex-valued function $f(x)$ with respect to an n-dimensional subspace V_n on a desired compact set B of the real line. The proposed approximation algorithm consists of two stand-alone exchange methods combined in a pipelined fashion: a very efficient multiple-exchange ascent algorithm which is a generalization of the second Remez (Parks-McClellan) algorithm to the complex case, and a robust multiple-exchange ascent-descent algorithm which makes use of a descent technique suitably adapted for minimizing the non-differentiable Chebyshev error norm.

Although very efficient in terms of storage and speed of convergence, the complex Remez multiple-exchange method may not converge to the optimal Chebyshev solution on the desired set B. The exchange ascent-descent method is more computationally expensive than the complex Remez method but it always results in the desired optimal Chebyshev approximation. Furthermore, as discussed in section 3.2.3, the ascent-descent method can be easily generalized to solve the complex multi-dimensional Chebyshev problem. A robust and efficient approximation algo-

rithm results by pipelining the two exchange procedures. In this way, the efficiency of the complex Remez multiple-exchange method is exploited while guaranteeing convergence to the optimal solution on the domain of interest.

The resulting composite exchange algorithm computes the optimal Chebyshev approximation in two stages: the complex Remez procedure as the first stage followed by the ascent-descent method as the second stage. In the many cases where the complex Remez method converges to the optimal solution on B, the algorithm terminates without proceeding into the second stage. Compared to existing techniques [22, 19], the proposed Chebyshev approximation algorithm exhibits improved memory requirements and better convergence characteristics, and is significantly more efficient when the optimal solution is obtained by the multiple-exchange complex Remez stage.

The individual exchange methods, forming the first and second stages of the algorithm, are described in sections 3.2.2 and 3.2.3, respectively. Filter Design examples are presented in section 3.5 to illustrate the performance of the proposed algorithm.

3.2.1 Motivation

From Theorems 2.5 and 2.6, the optimal Chebyshev solution can be found by computing the best approximations on all possible subsets B_s of r points and selecting the one with the maximum error norm. This exhaustive search strategy is not efficient nor convenient for practical use. As discussed in section 2.3, a good strategy would be to iteratively construct new subsets B_s with monotonically increasing error norms $\|\varepsilon_s\|$. For that purpose, two main issues must be addressed in developing the approximation algorithm:

1. Finding an efficient way to compute the best approximation $\phi_s(x)$ on a given subset B_s of r points ($r \geq n+1$).

2. Devising a good strategy to construct a new subset B_s with an increasing optimal error norm $\|\varepsilon_s\|$.

While in the real case it is sufficient to consider subsets of $r = n + 1$ points, the minimal subset size r is not known a-priori in the complex case and can take any value between $n + 1$ and $2n + 1$. It is desirable, whenever possible, to keep the size of the subsets B_s small since the computational complexity increases with the size of B_s.

The case where $r = n + 1$ points is important because, in that case, it was shown (section 2.4) that the best approximations on subsets of $n + 1$ points can be simply computed by solving a linear system of equations (2.33). So, the first issue is directly resolved. In addition, by exploiting the alternation property of the optimal error on B_s, efficient exchange rules can be derived and the second issue is easily resolved. These exchange rules are derived in section 3.2.2 resulting in a very efficient multiple-exchange ascent algorithm which iteratively constructs best approximations on subsets of $n + 1$ points with monotonically increasing error norms. This complex Remez multiple-exchange algorithm is based on the optimality criteria derived in section 2.4 and converges to the optimal Chebyshev solution on B when the optimal error $\varepsilon_o(x)$ satisfies (2.34). In the general case, the computed solution is optimal over a reduced set $B' \subset B$. The reason why the best Chebyshev approximation on B is not always obtained can be seen by considering the linear system of equations (2.33) which computes the best approximation on a subset B_s of $n+1$ points. Note that the magnitude of δ in (2.33) is equal to the norm of the optimal error on B_s, i.e. $\|\varepsilon_s\| = |\delta|$. Thus, the complex Remez multiple-exchange method iteratively generates a sequence of increasing error norms $|\delta|$. From (2.20), δ is a complex linear functional of the form

$$\delta = L_{n+1}(f) = \sum_{k=1}^{n+1} \lambda_k f(x_k). \tag{3.7}$$

On the other hand, Theorems 2.7 and 2.8 indicate that the optimal error norm $\|\varepsilon_o\|$ on B is an upper bound for *all* linear functionals of the form

$$L_r(f) = \sum_{k=1}^{r} \lambda_k f(x_k), \quad n + 1 \leq r \leq 2n + 1 \tag{3.8}$$

where $\{x_k\}_{k=1}^{r}$ are any r distinct points in B, and where $\{\lambda_k\}$ are such that $L_r(f)$ has norm one and vanishes on V_n. The maximum is always attained for some functional

$$L_{r_o}^{*}(f) = \sum_{k=1}^{r_o} \lambda_k^{*} f(x_k^{*}), \quad n+1 \leq r_o \leq 2n+1. \tag{3.9}$$

Consequently, the max of $|\delta|$ may be less than $\|\varepsilon_o\|$ unless one of the following conditions holds:

- $r_o = n+1$.
- $r_o > n+1$ and (at least) $n+1$ of the elements of the ordered set $\{\lambda_k^{*}\}_{k=1}^{r_o}$ in (3.9) have alternating signs (Theorem 2.34).

When the complex Remez multiple-exchange method results in a solution optimal only on a reduced set $B' \subset B$, the max of $|\delta|$ is less than the upper bound $\|\varepsilon_o\|$. In that case, subsets B_s of more than $n+1$ points ($r > n+1$) need to be considered in order to compute the optimal solution on the desired set B.

When $r > n+1$, both issues addressed above are harder to resolve. Theorem 2.9 presents a way to compute the best approximation on a subset B_s with more than $n+1$ points; however, this method is mainly of theoretical interest and is not very convenient for practical use. Moreover, a simple and efficient point-exchange strategy, where the size of B_s is kept minimal and constant, does not seem possible when $r > n+1$. In this case, a multiple-exchange ascent method is presented (section 3.2.3) for constructing a sequence of best approximations on subsets of r points ($r > n+1$) with monotonically increasing error norms (ascent strategy). The best approximations on the subsets B_s are computed by means of a descent method suitably adapted for minimizing the non-differentiable Chebyshev error norm. Other general optimization techniques (e.g. simplex method of linear programming [20, 22]), which are suitable for computing the best Chebyshev approximations on finite subsets of more than $n+1$ points, can also be used instead of the descent method in the second stage of the proposed two-stage algorithm.

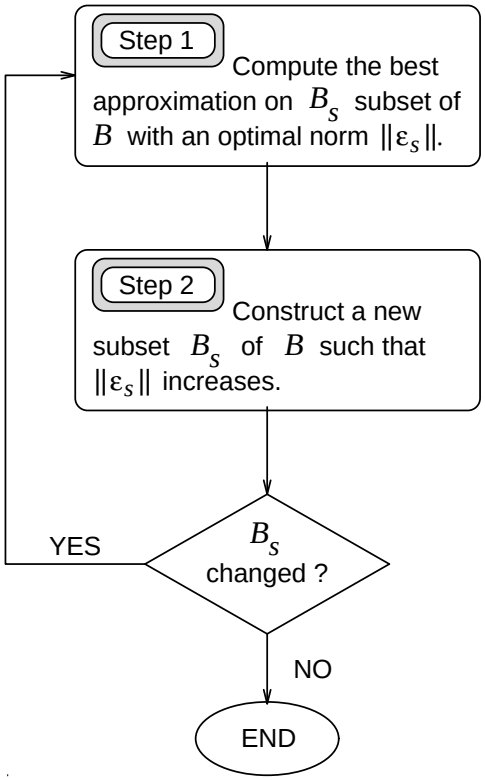


Figure 3.1. Ascent structure of the two stages and algorithm.

From the above discussion, it can be concluded that the two multiple-exchange stages (complex Remez and ascent-descent) have the same basic ascent structure. They both consist of the two main steps shown in Fig. 3.1, and they only differ in the way these steps are implemented. The new approximation algorithm, obtained by combining the two exchange strategies, is thus a multiple-exchange ascent algorithm with the same structure (Fig. 3.1). This algorithm starts by constructing a sequence of best approximations on subsets of $n+1$ points (minimum possible size) using the very efficient complex Remez algorithm (section 3.2.2) and terminates if $\max|\delta| = \|\varepsilon_o\|$. Otherwise, the algorithm continues constructing the sequence of best approximations with increasing error norms on subsets of more than $n+1$ points by means of the multiple-exchange ascent-descent method. A block diagram of the approximation algorithm is outlined in Fig. 3.2.

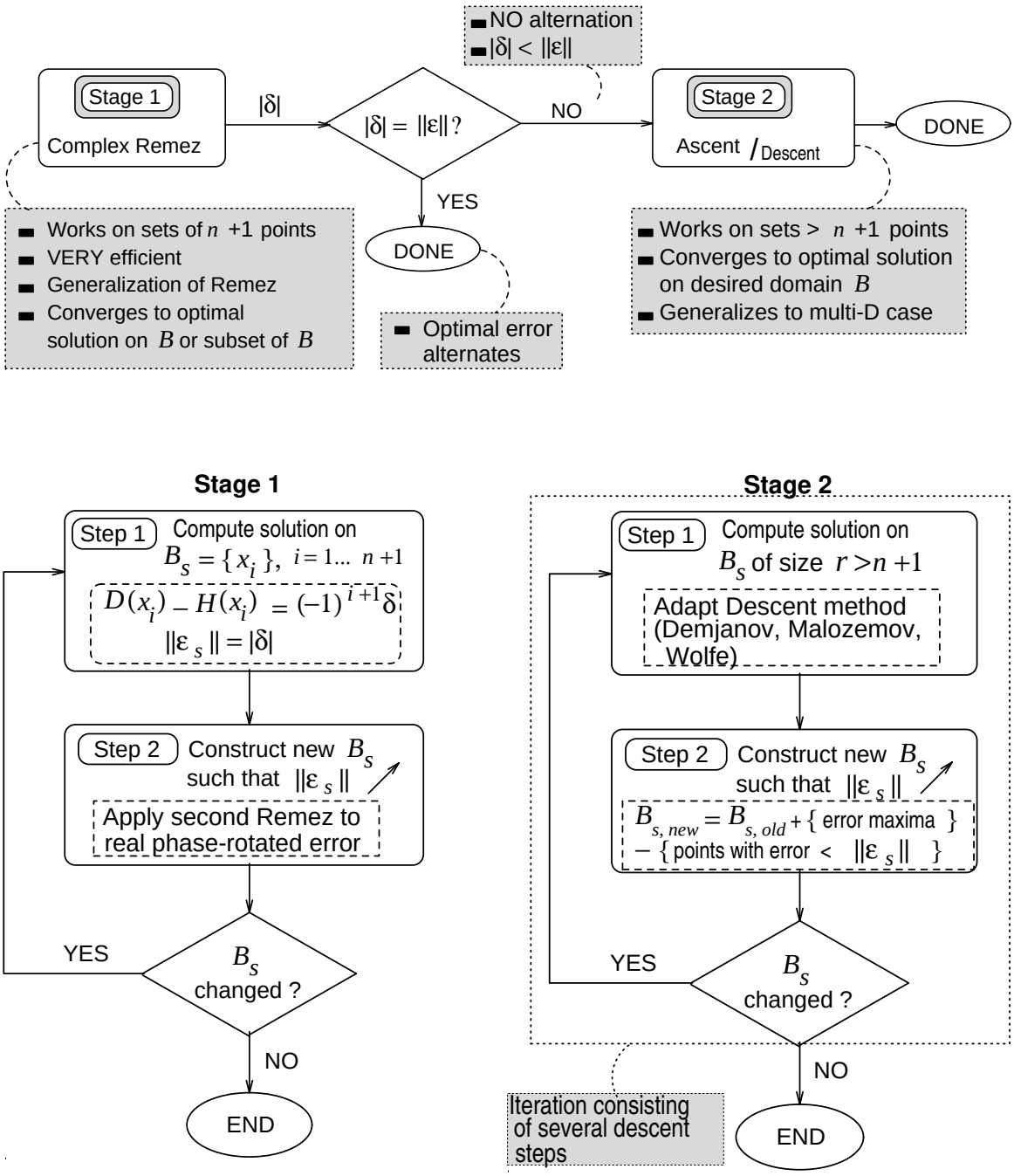


Figure 3.2. Block diagram of the multiple-exchange algorithm. $|\delta|$ is the maximal optimal deviation on the sets B_s consisting of $n+1$ points in B. $\|\varepsilon\|$ is the Chebyshev error norm on B.

It should be noted that, in order to compute the best approximation, the continuous domain B is represented by a dense set of discrete points. Therefore, from Theorem 2.6, the approximation algorithm must yield an approximation of maximum norm in a finite number of iterations since there is a finite number of distinct subsets B_s containing r $(n+1 \leq r \leq 2n+1)$ points in the discrete set B.

An interesting analogy of the proposed two-stage algorithm with the first and second algorithms of Remez can be made. It should be noted that both Remez algorithms are used for solving real one-dimensional Chebyshev approximation problems satisfying the Haar condition (constrained real Chebyshev problems). The two real Remez algorithms involve the solution of a sequence of discrete problems [30]: at each iteration, a finite discrete subset B_s is defined and the best Chebyshev approximation is computed on B_s. In the second algorithm of Remez, the successive subsets B_s contain exactly $n+1$ points: an initial subset of $n+1$ points is replaced by $n+1$ local maxima of the current real error function. In the first algorithm of Remez, the initial point set contains at least $n+1$ points, and these points are supplemented at each iteration by the global maximum of the current approximation error. As section 3.2.2 shows, the complex Remez stage (stage 1) of the new proposed algorithm is a generalization of the second Remez algorithm to the complex case and reduces to it when real-valued functions are approximated. On the other hand, the second stage of the proposed algorithm can be compared to the first Remez algorithm in that the size of the constructed subsets B_s is variable and is greater than $n+1$ (except at the initial iteration). A main difference between the second stage and the first Remez algorithm is that the second stage is based on a multiple-exchange strategy while the first algorithm of Remez is a single-exchange method.

3.2.2 The Complex Remez Multiple-Exchange Algorithm

The existing complex Chebyshev approximation algorithms are mainly based on general-purpose optimization techniques. None exploits directly the characteristics of

the complex Chebyshev optimal solutions. Algorithms based on the characterization of optimal solutions, such as the Parks-McClellan algorithm for the real case [29], have proven to be much more efficient than those based on general optimization techniques. One result of this thesis is to devise such an algorithm for the complex case.

Based on the results presented in section 2.4, this section derives an ascent multiple-exchange algorithm for computing the best Chebyshev approximation of a complex-valued function $f(x)$ with respect to an n-dimensional subspace V_n on a given compact set B of the real line. The algorithm converges to the optimal solution on the whole set B if the error of the best approximation satisfies (2.34) on B (from Theorem 2.12, it is sufficient, but not necessary, for the error to have exactly $n+1$ extremal points). In the general case, the computed solution is the best approximation of f on a reduced set B' of B. The new algorithm is a generalization of the Parks-McClellan algorithm, so that arbitrary magnitude and phase responses can be approximated. Both causal and non-causal filters with complex or real-valued impulse responses can be designed.

The algorithm is based on (2.33) and an ascent Remez multiple-exchange method [10, 31, 29] that will be described in this section. Equation (2.33) is repeated below for convenience:

$$\varepsilon(x_i) = f(x_i) - \phi(x_i) = (-1)^{i+1}\delta\,, \quad i = 1, \ldots, n+1 \tag{3.10}$$

where f is the function to be approximated with respect to V_n, $\phi \in V_n$ is the candidate best approximation, $\{x_i\}_{i=1}^{n+1}$ are the candidate extremal points, and δ is the optimal complex Chebyshev error on $\{x_i\}_{i=1}^{n+1}$.

The optimal error must have at least $n+1$ extremal points by virtue of the Haar condition (see Definition 2.5). Instead of directly minimizing the maximal magnitude of the error function, one attempts to solve the dual problem by finding the set of $n+1$ extremal points for which $|\delta|$ attains its maximum value. The best approximation is obtained if $|\varepsilon(x)| \leq |\delta|$ for all x in B since, as stated in (2.32), $|\delta|$ is a lower bound of the optimal error norm.

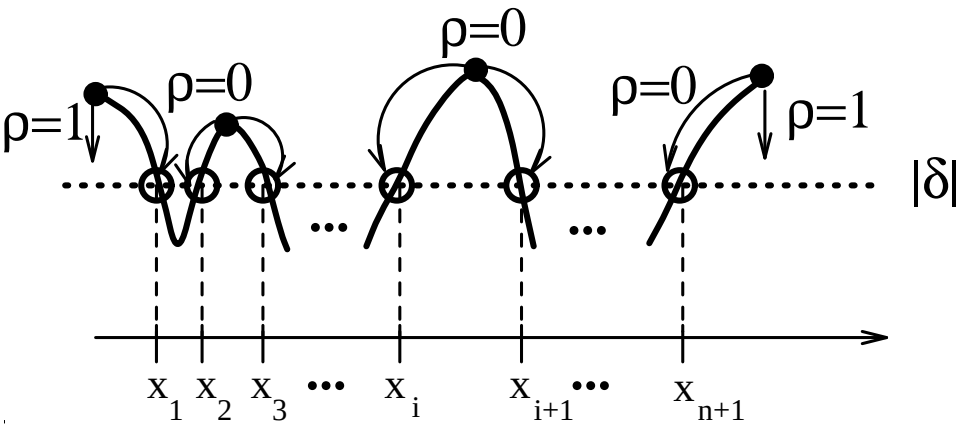


Figure 3.3. Illustration of the replacement (3.12): Filled circles represent local maxima of $|\varepsilon(x)|$. Non-filled circles correspond to the candidate points $\{x_i\}_{i=1}^{n+1}$ on which the error $\varepsilon(x)$ is forced to alternate. ρ is one if a local maximum is added at the beginning (end) with the last (first) point x_{n+1} (x_1) removed. It is zero when a local maximum replaces and takes the same index of an adjacent point x_i ($i = 1, \ldots, n+1$).

The algorithm begins with an initial guess of $n+1$ points, and exchanges points such that $|\delta|$ is increased at each iteration. At the k$^{\text{th}}$ iteration, there are $n+1$ points $\{x_i\}_{i=1}^{n+1}$ on which the error is forced, by solving (3.10), to have magnitude $|\delta|$ with an alternating phase shift of π. If there is a point ζ such that

$$|\varepsilon(\zeta)| > |\delta| \tag{3.11}$$

then it is desired to find a new set of candidate extremal points such that $|\delta|$ increases at the next iteration. However, note that, even if $|\delta|$ reaches its maximum value, it may be less than $\|\varepsilon_o\|$ which is an upper bound for all bounded linear functionals (Theorem 2.7 and section 3.2.1). This certainly occurs when the optimal error does not satisfy (2.34) on the set B and, in that case, the algorithm converges to the optimal Chebyshev approximation over only a subset of the desired bands. The following discussion derives the replacement conditions that would ensure an increase of $|\delta|$ at each iteration.

For simplicity, consider a single exchange method where a new candidate set

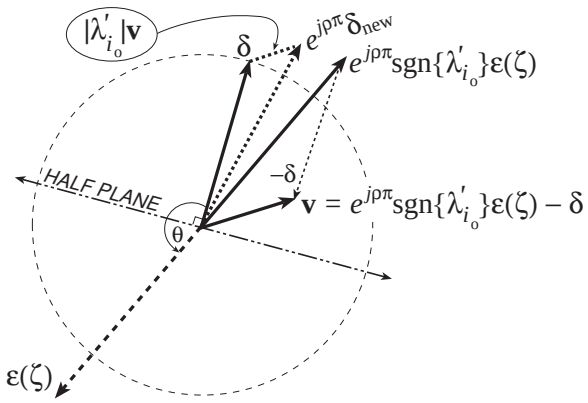


Figure 3.4. Vector interpretation of the exchange rule to get a maximum increase in δ_{new}. The index i_o must be chosen so that $e^{j\rho\pi}\text{sgn}\{\lambda'_{i_o}\}\varepsilon(\zeta)$ lies in the same half-plane as δ. A stronger condition on $\varepsilon(\zeta)$ that ensures an increase is that the vector $\mathbf{v}$ also lie in the same half-plane as δ, because $e^{j\rho\pi}\delta_{\text{new}} = \delta + |\lambda'_{i_o}|\mathbf{v}$.

$\{x'_i\}_{i=1}^{n+1}$ is formed by replacing one point x_{i_o} of the current set $\{x_i\}_{i=1}^{n+1}$ by a point $\zeta \in B$. One objective is to develop a rule for selecting the points x_{i_o} and ζ. Let $\varepsilon(x) = f(x) - h(x)$ be the error at the *current* iteration, $\theta_{\varepsilon(x)}$ be its phase, and $x'_{i_o} = \zeta$. The replacement is such that

$$\theta_{\varepsilon(x'_i)} = \theta_{\varepsilon(x_i)} + \rho\pi,\ i = 1, \ldots, n+1 \quad \text{and} \quad i \neq i_o \tag{3.12}$$

where ρ is either zero or one. As illustrated by Fig. 3.3, ρ is one if ζ is added at the beginning (end) with the last (first) point x_{n+1} (x_1) removed. It is zero when ζ takes the same index as the adjacent point to be replaced. Constructing the linear functional δ_{new} corresponding to the new set $\{x'_i\}_{i=1}^{n+1}$ as in (3.7) and (2.31)

$$\delta_{\text{new}} = \sum_{i=1}^{n+1} \lambda'_i f(x'_i) = \sum_{i=1}^{n+1} \lambda'_i \varepsilon(x'_i) \tag{3.13}$$

one gets after some simplifications (Appendix B)

$$\delta_{\text{new}} = e^{j\rho\pi}\left(\delta(1 - |\lambda'_{i_o}|) + |\lambda'_{i_o}|(\text{sgn}\{\lambda'_{i_o}\}e^{j\rho\pi}\varepsilon(\zeta))\right). \tag{3.14}$$

A vector interpretation of (3.14) is given in Fig. 3.4, which shows that it is necessary for the vectors $v_1 = \delta$ and $v_2 = \text{sgn}\{\lambda'_{i_o}\}e^{j\rho\pi}\varepsilon(\zeta)$ to be in the same half plane in order

to get a maximum increase of δ_{new}. This can always be done by selecting the index i_o such that

$$e^{j\rho\pi}(-1)^{i_o+1}\cos\theta > 0 \tag{3.15}$$

where θ is the angle between the vectors δ and $\varepsilon(\zeta)$, and $\text{sgn}\{\lambda'_{i_o}\}$ has been replaced with its known value $(-1)^{i_o+1}$. Condition (3.15) represents the rule for selecting the point for exchange x_{i_o}.

While, for filters with real frequency responses (including filters with complex coefficients), (3.11) and (3.15) are necessary and sufficient for increasing $|\delta|$, they are only necessary conditions for general complex filters. So, now, a condition is developed for selecting the point ζ that would ensure an increase of $|\delta|$ at the next iteration. Equation (3.14) can be rewritten as:

$$\delta_{\text{new}} = e^{j\rho\pi}\left(\delta + |\lambda'_{i_o}|\left(\text{sgn}\{\lambda'_{i_o}\}e^{j\rho\pi}\varepsilon(\zeta) - \delta\right)\right). \tag{3.16}$$

The requirement $|\delta_{\text{new}}|^2 > |\delta|^2$ reduces to

$$\begin{aligned} 2\,\Re e\left\{\delta^*\left(\text{sgn}\{\lambda'_{i_o}\}e^{j\rho\pi}\varepsilon(\zeta) - \delta\right)\right\} + \\ |\lambda'_{i_o}|\left|\text{sgn}\{\lambda'_{i_o}\}e^{j\rho\pi}\varepsilon(\zeta) - \delta\right|^2 > 0 \end{aligned} \tag{3.17}$$

where δ^* denotes the complex conjugate of δ.

A stronger condition that is independent of λ'_{i_o} and is sufficient to guarantee an increase in δ is

$$\Re e\left\{\delta^*\left((-1)^{i_o+1}e^{j\rho\pi}\varepsilon(\zeta) - \delta\right)\right\} \geq 0 \tag{3.18}$$

Equation (3.18) can be reduced to a practical rule as follows. Letting the phases of δ and $\varepsilon(\zeta)$ be denoted as θ_δ and θ_ε, respectively, it follows that

$$\Re e\left\{\left((-1)^{i_o+1}e^{j\rho\pi}e^{-j\theta_\delta}|\varepsilon(\zeta)|e^{j\theta_\varepsilon} - |\delta|\right)\right\} \geq 0 \tag{3.19}$$

or

$$(-1)^{i_o+1}(-1)^{\rho}|\varepsilon(\zeta)|\cos(\theta_\varepsilon - \theta_\delta) \geq |\delta|. \tag{3.20}$$

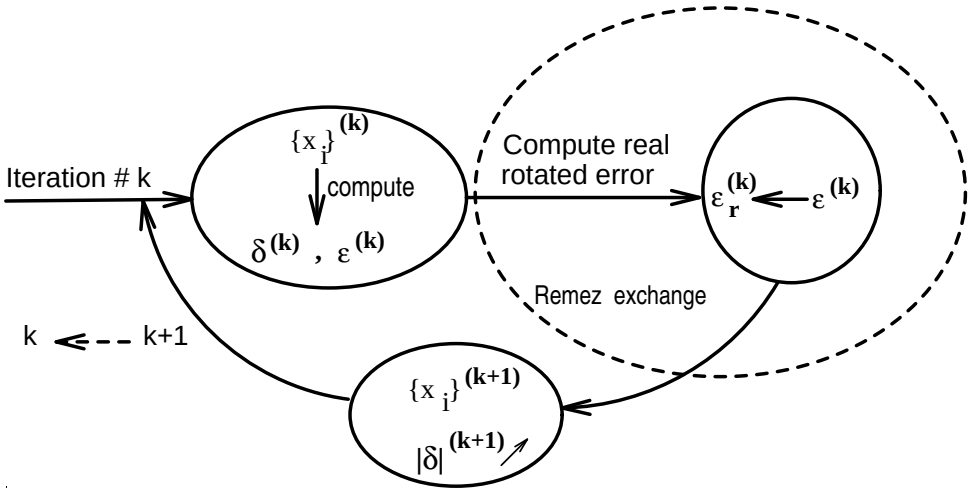


Figure 3.5. Graph illustrating one iteration of the multiple-exchange complex Remez algorithm.

Note that (3.20) also implies that the index i_o must be chosen as in (3.15). Now, if one defines a real "phase-rotated" error

$$\varepsilon_r(x) = \Re e\left\{e^{-j\theta_\delta}\varepsilon(x)\right\} \tag{3.21}$$

then the sufficient condition (3.18) on ζ becomes

$$|\varepsilon_r(\zeta)| \geq |\delta| \tag{3.22}$$

and the point for exchange is x_{i_o} satisfying

$$e^{j\rho\pi}\mathrm{sgn}\{\varepsilon_r(x_{i_o})\} = \mathrm{sgn}\{\varepsilon_r(\zeta)\}. \tag{3.23}$$

Equations (3.15) and (3.23) are equivalent because $\mathrm{sgn}\{\varepsilon_r(x_{i_o})\} = \mathrm{sgn}\{\lambda'_{i_o}\}$ and $\mathrm{sgn}\{\varepsilon_r(\zeta)\} = \mathrm{sgn}\{\cos\theta\}$.

The replacement conditions (3.22) and (3.23) are similar to the conditions of the real second Remez exchange algorithm [10, 29]. Thus, as shown in Fig. 3.5, the Remez multiple-exchange algorithm is applied to the real phase-rotated error $\varepsilon_r(x)$ in order to determine the new set of candidate extremal points.

A block diagram of the approximation algorithm is given in Fig. 3.6. Note

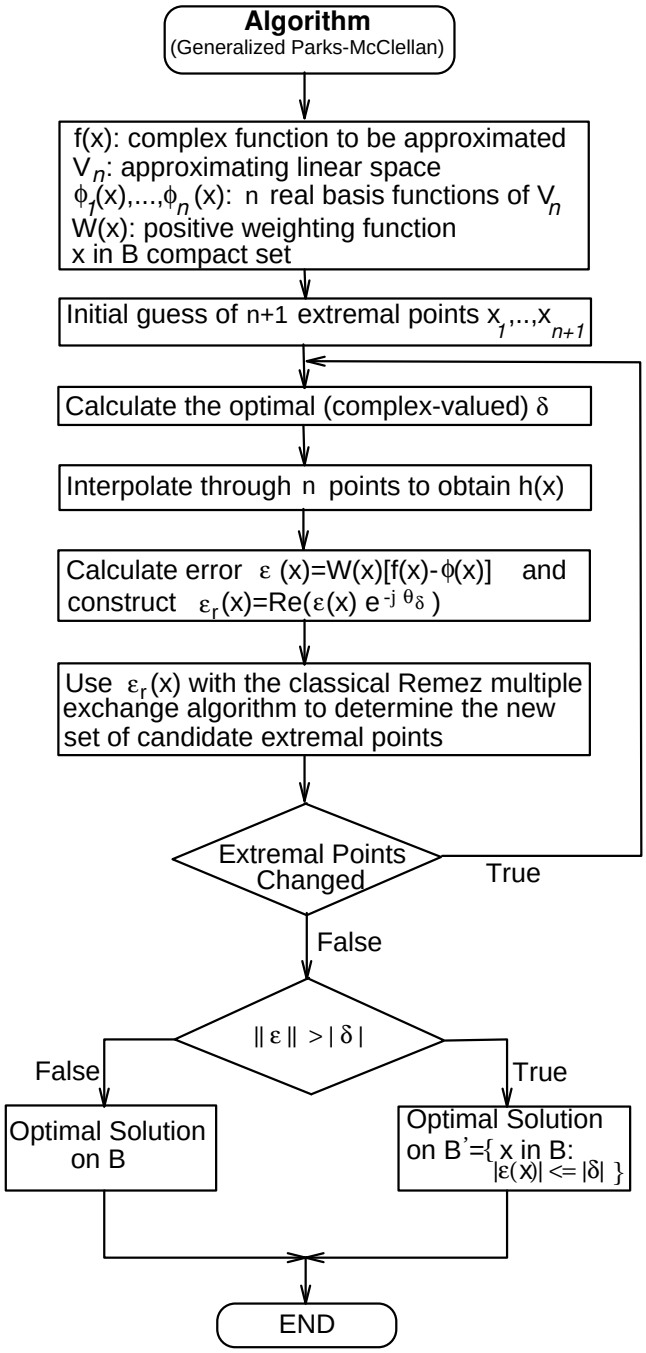


Figure 3.6. Block diagram of the complex Remez algorithm.

that the presented algorithm has the *same* structure as the second Remez (Parks-McClellan) algorithm. In particular, when $f(x)$ is a real-valued function, δ will also be real from (2.20) and, therefore, $\varepsilon_r(x) = \pm\varepsilon(x)$. In this case, our algorithm reduces to the Parks-McClellan (second Remez) algorithm as modified by McCallig [37] for approximating real-valued frequency responses in the Chebyshev sense. Compared to existing techniques (see section 3.6), the proposed algorithm is significantly more efficient in terms of memory and speed of convergence.

Given a desired approximation domain B, the new algorithm converges to the best-weighted Chebyshev approximation on B when the optimal error satisfies a given alternation property (see light-shaded set in Fig. 2.2). This convergence behavior has been observed in all examples, but a mathematical convergence proof is yet to be developed. In the general case (corresponding to the dark-shaded region in Fig. 2.2), the algorithm converges to the optimal Chebyshev solution on some compact set $B' \subseteq B$. When $B' \approx B$, the resulting approximating filter (optimal on B') is highly satisfactory. The optimality of the resulting filter can always be directly determined using a simple rule (Fig. 3.6).

It should be noted that, for a given B, the optimal deviation cannot be less than the value of $|\delta|$ which is computed by the multiple-exchange complex Remez algorithm. In contrast to other methods (e.g. linear programming [22]) which stop when the computed error is within a specified tolerance from the optimal deviation, the optimal solutions resulting from the complex Remez algorithm correspond to the *true* optimal deviation.

3.2.3 The Ascent-Descent Multiple-Exchange Algorithm

This section presents a new multiple-exchange ascent-descent algorithm that computes the best Chebyshev approximation of a complex-valued function $f(x)$ with respect to an n-dimensional subspace V_n on a desired compact set B of the real line. The proposed algorithm can be directly generalized to solve the unconstrained

multi-dimensional Chebyshev problem formulated in section 2.1.

Like the complex Remez method (section 3.2.2), the presented ascent-descent algorithm exploits the fact that the best Chebyshev approximation on B is also the best Chebyshev approximation on some finite subset $B_s \subset B$. It is based on an ascent strategy which iteratively constructs a sequence of best approximations on subsets B_s with monotonically increasing error norms. The best approximations on the sparse finite subsets B_s are computed using a generalized descent method based on the work of Demjanov and Malozemov [38]. Demjanov and Malozemov applied a generalized descent method to solve general minimax problems in one step on the desired domain B. However, when applied to the Chebyshev problem, this one-step strategy was found to be very computationally intensive due to the use of a dense grid (or subgrid) at each iteration and it exhibited slow and unstable convergence. A robust and more efficient technique results from using the ascent strategy described earlier, where the descent method is applied to solve a sequence of subproblems on sparse finite subsets $B_s \subset B$. This ascent strategy, which replaces the dense set B with sparse subsets B_s, is at the core of efficient exchange Chebyshev approximation algorithms [39, 29, 40].

The steps of the ascent-descent algorithm can be summarized as follows:

1. **Select initial subset** B_s of r points, where $n+1 \leq r \leq 2n+1$ and $B_s \subset B$. Take an initial approximation $\phi_{s,0}$ on B_s.

2. **Compute the optimal solution** ϕ_s **on** B_s with optimal deviation $\|\varepsilon_s\|$. The descent method is used for computing the best approximation on the subset B_s consisting of more than $n+1$ points. This is the main computational step and is described later in this section.

3. **Calculate the error** $\varepsilon(x) = f(x) - \varphi_s(x)$, where $x \in B$.

4. **Check stopping criteria.** The algorithm terminates if the extremal points of $\varepsilon(x)$ are in B_s.

5. **Update the subset B_s.** The located new local maxima of $\varepsilon(x)$ are added to the current subset B_s while points from B_s whose error magnitude is smaller than $\|\varepsilon_s\|$ are removed. The size of B_s is not necessarily kept constant but varies in general. Return to Step 2 with the current ϕ_s serving as the initial solution $\phi_{s,0}$ on the updated set B_s. □

Steps 2–5 constitute one iteration of the ascent-descent algorithm. It should be noted that each iteration (Steps 2–5) includes descent steps executed in Step 2. While Steps 3–5 are simple and clear, Steps 1 and 2 require further description.

Selecting the initial subset B_s (Step 1)

An initial approximation and an initial subset B_s can be generated by using the complex Remez algorithm of section 3.2.2. B_s is formed by taking the set of all local maxima of the error corresponding to the initial approximation. The advantage of this approach is that the complex Remez algorithm is extremely efficient and running it does not produce significant overhead. In fact, note that one iteration of the ascent-descent algorithm includes several descent steps, and that one iteration of the complex Remez method is significantly *less* complex than a single descent step. Moreover, for many problems, the resulting initial approximation computed by the complex Remez method is the optimal Chebyshev solution and, thus, the ascent-descent algorithm does not need to execute. This approach is further reinforced by the following two observations:

1. When a subset-optimal solution (i,e., optimal on a reduced set $B' \subset B$) is computed by the complex Remez method, it has been observed that the resulting optimal deviation $|\delta|$ is very close to the optimal error norm $\|\varepsilon_o\|$ (its upper bound).

2. It has been observed that, in many cases, the *local* maxima of the resulting error correspond or are very close in location to the extremal points (*absolute*

maxima) of the optimal Chebyshev error $\|\varepsilon_o\|$.

The first observation implies that $|\delta|$, computed by the complex Remez algorithm, is a tight lower bound of the optimal error. Therefore, $|\delta|$ can be used in Step 5 to avoid a large increase in the size of B_s. This is done by removing from B_s the points at which the error magnitude is less than $|\delta|$. The second observation suggests that a faster convergence of the ascent-descent algorithm may result by using the complex Remez algorithm as an initial design procedure in Step 1.

Descent Method (Step 2)

The descent method presented in this section is based upon the work of Demjanov–Malozemov [41, 38] and Wolfe [42] and is suitably adapted for minimizing the non-differentiable Chebyshev error norm.

Let $f(x)$ be the function that is to be approximated on B_s with respect to V_n, and let $\phi_{s,0}(\mathbf{x})$ be an initial approximation given by the basis coefficient vector

$$\mathbf{c}_0 = [c_{01}, c_{02}, \ldots, c_{0n}]^T \tag{3.24}$$

whose elements are the n (complex or real) coefficients associated with the (complex or real) basis functions $\{\phi_i\}_{i=0}^n$ of V_n. The superscript T in (3.24) refers to the transpose operation. The descent method iteratively generates a sequence $\{\mathbf{c}_k\}$ of basis coefficient vectors, $\{\mathbf{d}_k\}$ of perturbation vectors, and $\{t_k\}$ of positive scalars such that

$$\mathbf{c}_{k+1} = \mathbf{c}_k + t_k \mathbf{d}_k \tag{3.25}$$

and

$$\|\varepsilon_{s,k+1}(x)\| \leq \|\varepsilon_{s,k}(x)\| \tag{3.26}$$

where $\varepsilon_{s,k}(x)$ is the approximation error

$$\varepsilon_{s,k}(x) = f(x) - \phi_{s,k}(x) = f(x) - \sum_{i=1}^{n} c_{ki}\phi_i(x) \tag{3.27}$$

and k is the iteration number. The perturbation vectors $\{\mathbf{d}_k\}$ correspond to descent directions and $\{t_k\}$ must be chosen so that $\|\varepsilon_k(x)\|$ would significantly decrease at the next iteration. A line search method could be used to find the t_k that would result in a maximum decrease of $\|\varepsilon_k(x)\|$ along the direction $\mathbf{d}_k$. Alternatively, a more efficient procedure for finding the best t_k was presented in [38, pp. 109–112]. Standard gradient techniques cannot be used in this case for generating the directions $\{\mathbf{d}_k\}$ since the Chebyshev error norm is a non-differentiable function of the coefficient vector $\mathbf{c}$.

With r denoting the size of B_s, the Chebyshev approximation problem can be reformulated as the minimization of the function

$$\varphi(\mathbf{c}) = \max_{i\in(1,\ldots,r)} e_i(\mathbf{c}) \tag{3.28}$$

where

$$e_i(\mathbf{c}) = |f(x_i) - \Phi_i^T\mathbf{c}|^2 \tag{3.29}$$

and

$$\Phi_i = [\phi_1(x_i), \phi_2(x_i), \ldots, \phi_n(x_i)]^T. \tag{3.30}$$

Each $e_i(\mathbf{c})$ is a convex differentiable function with a complex gradient vector g_i given by

$$g_i = \frac{\partial e_i(\mathbf{c})}{\partial \mathbf{c}} = -2\overline{\Phi}_i\varepsilon_i \tag{3.31}$$

where $\overline{\Phi}_i$ is the complex conjugate of Φ_i, and $\varepsilon_i = f(x_i) - \Phi_i^T\mathbf{c}$. Note that g_i is a vector in the n-dimensional complex space Z_n which is isomorphic to the $2n$-dimensional real Euclidean space R_{2n}. A point $z = (z_1, \ldots, z_n) \in Z_n$, with complex coordinates $z_j = \alpha_j + j\beta_j$, corresponds to the point $z = (\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n) \in R_{2n}$. In what follows, g_i refers to the real vector in R_{2n}.

For a given coefficient vector $\mathbf{c}$, consider the set of extremal indices $I_e(\mathbf{c})$ defined as

$$I_e(\mathbf{c}) = \{i \in (1, \ldots, r) : e_i(\mathbf{c}) = \varphi(\mathbf{c})\} \tag{3.32}$$

In other words, $I_e(\mathbf{c})$ contains every index i (corresponding to the ith point x_i in B_s) for which $\varepsilon(x)$ attains its maximum on B_s. Letting

$$G(\mathbf{c}) = \{g_i : i \in I_e(\mathbf{c})\}\,, \tag{3.33}$$

consider the convex hull $G_c(\mathbf{c})$ of $G(\mathbf{c})$. $G_c(\mathbf{c})$ is a polyhedron in R_{2n} and there is a unique point $g_{min} \in G_c(\mathbf{c})$ having minimum Euclidean norm [43]. The following gradient characterization results for $\varphi(\mathbf{c})$ [41, 43]

$$\nabla\varphi(\mathbf{c}) = g_{min} \tag{3.34}$$

and $-g_{min}$ is the direction of steepest descent at $\mathbf{c}$. Note that $\nabla\varphi(\mathbf{c})$ depends only on the set of extremal points represented by $I_e(\mathbf{c})$. So, the problem of finding the steepest descent direction reduces to the problem of finding the point of smallest norm in the convex hull of a given finite point set. An algorithm especially designed for that calculation has been presented by Wolfe [42]. The filter coefficient vector $\mathbf{c}_o$ minimizes $\varphi(\mathbf{c})$, and therefore the approximation error $\|\varepsilon\|$, if and only if

$$\nabla\varphi(\mathbf{c}_o) = 0 \tag{3.35}$$

or, equivalently (see (3.34)),

$$0 \in G_c(\mathbf{c}_o)\,. \tag{3.36}$$

Using (3.31), it can be shown [32, p. 21] that the optimality condition (3.36) reduces to the Kolmogoroff optimality criterion for Chebyshev approximation (see Theorems 2.1 and 2.2).

While a direct generalization of the steepest descent method does not in general lead to convergence [41, 43], successive approximation and conjugate subgradient methods based on (3.34) have been developed for minimizing non-differentiable functions [38, 43, 44]. A new descent method was presented in [45] where the descent directions $\{\mathbf{d}_k\}$ were efficiently computed by exploiting the Kolmogoroff optimality criterion (Theorem 2.1). However, some of the generated descent directions could be

very feeble causing convergence problems. This Kolmogoroff-based descent method is described in Appendix C. The descent method presented in this section is based upon the techniques presented in [38] and [42]. It is suitably adapted for solving the Chebyshev approximation problem, which was reformulated as (3.28–3.30), and, consequently, for solving the filter design problem. Before describing the steps of the proposed descent method, some new definitions are needed. Define

$$I_{e,\epsilon}(\mathbf{c}) = \{i \in (1, \ldots, r) : \varphi(\mathbf{c}) - e_i(\mathbf{c}) \leq \epsilon\}, \qquad \epsilon \geq 0 \tag{3.37}$$

and

$$G_\epsilon(\mathbf{c}) = \{g_i : i \in I_{e,\epsilon}(\mathbf{c})\}. \tag{3.38}$$

Also, let $G_{c,\epsilon}(\mathbf{c})$ denote the convex hull of $G_\epsilon(\mathbf{c})$ and $g_{min,\epsilon}$ the point in $G_{c,\epsilon}(\mathbf{c})$ nearest to the origin. Clearly, $I_{e,0}(\mathbf{c}) = I_e(\mathbf{c})$, $G_0(\mathbf{c}) = G(\mathbf{c})$, $G_{c,0}(\mathbf{c}) = G_c(\mathbf{c})$, and $g_{min,0} = g_{min}$.

The basic steps of the descent algorithm can now be summarized as follows:

1. **Set initial parameters.** Fix two parameters $\epsilon_0 > 0$ (e.g.,$\epsilon_0 = 0.012$) and $\rho_0 > 0$ (e.g., $\rho_o = 1.0$), and take an initial approximation $\mathbf{c}_0$ on the desired set B_s, i.e. $\phi_{s,0}(x) = \sum_{i=1}^{n} c_{0i}\phi_i(x)$. Since the passage from $\mathbf{c}_k$ to $\mathbf{c}_{k+1}$ $(k = 0, 1, \ldots)$ is effected the same way, suppose that the k-th approximation $\mathbf{c}_k$ is already computed.

2. **Set current approximation and accuracy.** Set $\mathbf{c} = \mathbf{c}_k$, $\epsilon = \epsilon_0/2^k$ and $\rho = \rho_0/2^k$.

3. **Compute the ϵ-gradient, $g_{min,\epsilon}$.** Find the point $g_{min,\epsilon}$ of $G_{c,\epsilon}(\mathbf{c})$ nearest to the origin using the technique by Wolfe [42].

4. **Check accuracy of current approximation.** If $\|g_{min,\epsilon}\| \leq \rho$, goto Step 8.

5. **Compute the ϵ-steepest descent direction $\mathbf{p}_k$**

$$\mathbf{p}_k = -\frac{g_{min,\epsilon}}{\|g_{min,\epsilon}\|} \tag{3.39}$$

6. **Determine the best step size t_k.** Consider the ray

$$\mathbf{c}(t) = \mathbf{c} + t\mathbf{p}_k \tag{3.40}$$

and determine $t_k \geq 0$ such that

$$\varphi(\mathbf{c}(t_k)) = \min_{t \geq 0} \varphi(\mathbf{c}(t)) \tag{3.41}$$

7. **Refine approximation until the set accuracy is reached.** Set $\mathbf{c} = \mathbf{c}(t_k)$ and repeat from Step 3.

8. **Compute generalized gradient, g_{min}.** The technique by Wolfe [42] is used to find the point g_{min} of $G_c(\mathbf{c}_k)$ nearest to the origin. (see also [38, Appendix IV]).

9. **Check stopping criteria.** If $g_{min} \equiv \mathbf{0}$, then $\mathbf{c}_k$ is the vector of the coefficients of the best approximation $\phi_s(x)$ of the function $f(x)$ on $B_s = \{x_i : i = 1, \ldots, r\}$ and the algorithm terminates.

10. **Update approximation and repeat with higher accuracy.** The approximation $\mathbf{c}_{k+1}$ is now given by

$$\mathbf{c}_{k+1} = \mathbf{c}. \tag{3.42}$$

Return to Step 2. □

The main computational work lies in the calculation of the generalized gradient (Steps 3 and 8). Wolfe showed [42, pp. 137–138] that such calculation requires roughly $O(s_I^2 \log(s_I))$ operations (one operation = floating multiply + floating add), where s_I is the size of the set of extremal indices $I_{e,\epsilon}(\mathbf{c})$. Starting with a sparse set B_s of size r $(n+1 \leq r \leq 2n+1)$, note that s_I is bounded by r and that, in general, $s_I \ll r$ for most iterations, except possibly at the last iteration. Other suggestions for reducing further the computational work can be found in [42]. A convergence proof for this successive approximation descent method can be found in [38].

3.2.4 Relation to the Real (Linear-Phase) Filter Design Problem

The real-valued filter design problem corresponds to the case where the function $A(\omega)$, given by (3.1), reduces to a real-valued function. In this latter case, the complex Remez algorithm always converges to the unique optimal solution of the desired function $D(\omega)$ on the specified frequency bands B.

Given a desired filter frequency response $D(\omega)$ that is to be approximated by an FIR filter having an impulse response $h(k)$, nonzero for $k = N_1, \ldots, N_2$, $A(\omega)$ is obtained from $D(\omega)$ as follows:

$$A(\omega) = e^{j\frac{N_1+N_2}{2}\omega} D(\omega). \tag{3.43}$$

From (3.43), it can be seen that $A(\omega)$ is real-valued if $D(\omega)$ has a linear phase and is of the form

$$D(\omega) = e^{-j\frac{N_1+N_2}{2}\omega} D_{\Re}(\omega) \tag{3.44}$$

where $D_{\Re}(\omega)$ is a real-valued function continuous on B. More generally, it can be shown that the FIR filter design problem reduces to a real-valued approximation problem when $D(\omega)$ has a *generalized* linear phase [1] of the form

$$D(\omega) = e^{-j(\frac{N_1+N_2}{2}\omega+\beta)} D_{\Re}(\omega) \tag{3.45}$$

where β is a constant and $D_{\Re}(\omega)$ is a real-valued function continuous on B. This result can be deduced from the following theorem:

Theorem 3.1 Let V_n be the subspace defined as in sections 2.4.1 and 3.1, and let γ be a (complex or real) nonzero constant. $\phi_o(\omega) \in V_n$ is the best Chebyshev approximation of $f(\omega)$ on B (i.e., $\omega \in B$) if and only if $\gamma\phi_o(\omega) \in V_n$ is the best Chebyshev approximation of $\gamma f(\omega)$ on B. □

Proof: Since V_n satisfies the Haar condition, $\phi_o(\omega)$ is the unique best approximation of $f(\omega)$ with respect to V_n on B. Let $\hat{\phi}_o(\omega)$ be the unique best Chebyshev approxi-

mation of $\gamma f(\omega)$ with respect to V_n on B. Then,

$$\|\hat{\varepsilon}_o\| = \|\gamma f(\omega) - \hat{\phi}_o(\omega)\| = \min_{\phi(\omega)\in V_n} \|\gamma f(\omega) - \phi(\omega)\|. \tag{3.46}$$

Factoring out the constant γ, (3.46) becomes

$$\|\hat{\varepsilon}_o\| = |\gamma| \|f(\omega) - \frac{\hat{\phi}_o(\omega)}{\gamma}\| \tag{3.47}$$

From (3.47), it is clear that $\|\hat{\varepsilon}_o\|$ is minimized if and only if $\|f(\omega) - \dfrac{\hat{\phi}_o(\omega)}{\gamma}\|$ is minimized. Also, note that $\hat{\phi}_o(\omega)/\gamma \in V_n$, which is a *linear* subspace. So, it follows that

$$\phi_o(\omega) = \frac{\hat{\phi}_o(\omega)}{\gamma}. \tag{3.48}$$

□

When $D(\omega)$ is of the form (3.45), $A(\omega)$ is given by

$$e^{-j\beta} D_{\Re}(\omega). \tag{3.49}$$

So, if $H_{\Re o}$ is the best Chebyshev approximation of $D_{\Re}(\omega)$, it follows from Theorem 3.1 that $e^{-j\beta} H_{\Re o}$ is the best Chebyshev approximation of $A(\omega)$. Consequently,

$$H_o(\omega) = e^{-j(\frac{N_1+N_2}{2}\omega+\beta)} H_{\Re o} \tag{3.50}$$

is the best Chebyshev approximation of $D(\omega)$. So, the problem of approximating (3.45) in the Chebyshev sense reduces to the problem of finding the best Chebyshev approximation of the real-valued function $D_{\Re}(\omega)$.

It should be noted that, when $D(\omega)$ is a real-valued function, $A(\omega)$ is generally a complex-valued function which depends on the indices N_1 and N_2 (see (3.43)). So, in that case, a complex problem results unless

$$N_1 = -N_2 \tag{3.51}$$

which implies that the approximating filter should be noncausal and should have odd length.

A remark in relation to even-length filters is now in order. Even-length FIR filters are commonly used to approximate real-valued ideal frequency responses by solving a *real* Chebyshev approximation problem [11, 1]. However, in this latter case, even-length filters add a half-sample phase term to the resulting real-valued best Chebyshev solution. It follows that the resulting even-length filter is *not* the best Chebyshev approximating even-length filter of the real-valued ideal function. In order to find such filter, the phase term needs to be explicitly incorporated in the approximation problem. Thus, a complex Chebyshev approximation problem needs to be solved.

3.3 Multidimensional Chebyshev Approximation

The multiple-exchange ascent-descent algorithm of section 3.2.3 can be used with minor modifications to solve the unconstrained multidimensional problem stated in section 2.1. In fact, it can be easily seen that the formulation (3.28–3.30) applies to the general Chebyshev problem with B_s taken to be a compact set of a multidimensional space.

The generalized algorithm for the multidimensional problem consists of the same steps as the ascent-descent algorithm of section 3.2.3 with the following minor modifications:

1. The domain B and the subsets B_s are compact sets of an m-dimensional space E_m and, thus, the scalar point x should be replaced by the vector point $\mathbf{x} \in E_m$.

2. The complex Remez algorithm, which applies only to the constrained Chebyshev problem of section 2.4.1, can no longer be used as an initial stage. So, other means should be used to compute the initial solution on the initial set B_s (Step 1 of the ascent-descent algorithm).

Issues related to the implementation and performance of the above generalized ascent-descent algorithm are not considered in this work.

3.4 Special Types of Filters

3.4.1 Low Delay Filters with Nearly-linear Phase

In many signal processing applications, such as speech and image processing, it is desirable to pass some portion of the signal frequency band with minimal distortion. For that purpose, linear-phase systems are particularly desirable since the effect of the linear-phase is a simple delay.

There is often a need to design digital filters with other than exactly linear phase. While exactly linear-phase filters exhibit a constant group delay, the delay they introduce is proportional to the filter length N and is equal to $(N-1)/2$. Consequently, the introduced delay may be unacceptably large, especially when using filters having a high degree of selectivity (sharp cutoff edges). Furthermore, in real-time applications (e.g., real-time speech and video processing), selective filters are required to have a constant group delay that is as small as possible. Minimum-phase FIR filters cause less delay but introduce phase distortion which may have a severe effect on the shape of the processed signal.

Complex approximation can be used to design filters which have less delay than the exactly linear-phase filter of the same length, and which have approximately a constant delay in the passband. The resulting complex filters are called "nearly linear-phase" and are obtained by approximating a desired linear-phase frequency response given by

$$D(\omega) = \begin{cases} e^{-j\,\tau_d\,\omega}\,, & \omega \in \text{passbands} \\ 0\,, & \omega \in \text{stopbands} \end{cases} \tag{3.52}$$

where τ_d is the desired group delay in the filter passbands. Different group delays can be specified for each of the passbands. Note that the phase term is explicitly included in the approximation problem. In this way, the desired delay is fixed and is not determined by the FIR filter length N. Moreover, increasing N does not increase the group delay but rather leads to a better approximation of the desired constant

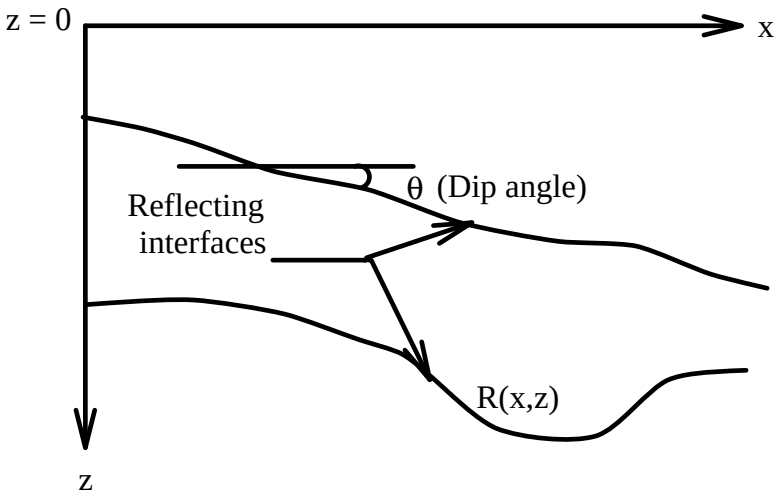


Figure 3.7. Simplified model of the earth from [46].

delay. This point is illustrated by the design examples in section 3.5. In addition, if $D(\omega)$ is conjugate symmetric, the frequency response of the optimal Chebyshev filter approximating (3.52) will also be conjugate symmetric [31, p. 27] and, therefore, the approximating filter coefficients will be real-valued.

An interesting problem mentioned in [5] is to determine the value of τ_d such that the designed causal FIR filter has the smallest error deviation. Chen and Parks observed [5] that the desired group delay which gives the minimum error deviation is smaller than that of an exactly linear-phase filter of the same length.

3.4.2 Seismic Migration Filters with Quasi-Quadratic Phase

The term *migration* refers to a processing technique that is applied to seismic data sections to compensate for some undesirable effects of wave propagation [46, 4, 47]. The geometry of the seismic 2-D migration problem is illustrated in Fig. 3.7, which is a simplified view of a two-dimensional earth. The variable x represents position on earth, which is assumed to be flat, and the variable z represents depth. The objective is to define the boundaries of the earth layers. For this purpose, downward

propagating waves are initiated by acoustic sources at the earth surface ($z = 0$) for all time $t > 0$. It is desirable to obtain an estimate of the true reflectivity $R(x, z)$ at a point (x, z) from the measurements done at the surface. The propagation of a wave $\psi(x, z, t)$ one way, either up or down, with a uniform velocity v is governed by the following wave equation

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial z^2} = \frac{1}{v^2} \frac{\partial^2 \psi}{\partial t^2} \tag{3.53}$$

with the measured wave $\psi(x, 0, t)$ serving as a boundary condition for (3.53). The "exploding reflector" concept [47] is applied by selecting upgoing waves with a velocity of one-half the velocity of the real medium. In the "exploding reflector" concept, the acoustic sources are thought as being spread on the reflecting interfaces and initiated simultaneously at $t = 0$. The receivers at $z = 0$ record the upward wave $\psi(x, 0, t)$ propagating at a constant velocity for all $t > 0$.

The migration procedure starts with $\psi(x, 0, t)$ measured at the surface and computes $\psi(x, z, t)$ for all desired depths. To allow for vertical velocity (v) variations, the wavefield is extrapolated recursively in small depth increments Δz over which the velocity is assumed to be constant. $\psi(x, z, t = 0)$ is then a good approximation of $R(x, z)$.

The operation of back-propagating the wave field across a layer of thickness Δz, e.g. from z_o to $z_o + \Delta z$, is actually a linear filtering operation. Letting $\Psi(K_x, z, \Omega)$ denote the 2-D Fourier transform of $\psi(x, z, t)$ at depth z and taking the Fourier transform of both sides of (3.53), one obtains a second-order differential equation for $\Psi(K_x, z, \Omega)$ in the variable z. Given the initial condition at $z = z_o$, the solution of the obtained differential equation for an upward propagating wave is given by [4]

$$\begin{aligned} \Psi(K_x, z_o + \Delta z, \Omega) &= e^{j\Delta z \sqrt{(\frac{\Omega^2}{v^2} - K_x^2)}} \Psi(K_x, z_o, \Omega) \\ &= D_A(K_x, \Omega) \Psi(K_x, z_o, \Omega) \end{aligned} \tag{3.54}$$

where the extrapolation operator $D_A(K_x, \Omega)$ is a linear shift-invariant analog filter. In practice, the wave field is presented in sampled form as $\psi(n_1 \Delta x, n_2 \Delta z, n_3 \Delta t)$,

where n_1, n_2, n_3 are integers, Δt is the temporal sampling interval, and $\Delta x, \Delta z$ are respectively the horizontal and vertical spatial sampling intervals. The extrapolation operation is then performed using a digital space-time filter whose frequency response approximates

$$D(k_x, \omega) = D_A(\frac{k_x}{\Delta x}, \frac{\omega}{\Delta t}) = e^{j\frac{\Delta z}{\Delta x}\sqrt{\frac{\Delta x^2}{\Delta t^2}\frac{\omega^2}{v^2} - k_x^2}}. \tag{3.55}$$

$D(k_x, \omega)$ in (3.55) is the ideal frequency response of an all-pass filter having a non-linear phase.

The extrapolation in depth is usually done for a fixed frequency ω_o and a fixed velocity v_o. The process is repeated for other frequency and velocity values using frequency- and velocity-dependent migration filters. Therefore, for a fixed ratio ω_o/v_o, $D(k_x, \omega_o) = D(k_x)$ is a 1-D migration filter with a cutoff frequency equal to $(\Delta x/\Delta t)(\omega_o/v_o)$. The objective of the migration filter design problem is to approximate the ideal frequency response $D(k_x)$. Since $D(k_x)$ is a complex-valued positive-symmetric function, it can be approximated by an N-length FIR (Finite Impulse Response) digital filter whose frequency response is given by [48]

$$H(k_x) = h_o + 2\sum_{n=1}^{(N+1)/2} h_n \cos nk_x \tag{3.56}$$

where the filter coefficients h_n are complex-valued. Note that, even if an arbitrary FIR filter is used, the resulting optimal filter will be positive-symmetric when an odd-length filter is used to approximate $D(k_x)$. This property follows directly from Chebyshev approximation theory [31, p. 27]. Setting $\alpha = (\Delta x/\Delta t)(\omega/v)$, the approximation needs mostly to be accurate in the region $|k_x| < |\alpha|$ which corresponds to the wavenumbers (k_x) for which waves are propagating [4, 46]. The evanescent region $|k_x| > |\alpha|$ is presumed to contain little or no energy.

Migration filters are rated in terms of the maximum angle (from the vertical direction) at which propagating waves can be accurately extrapolated. This angle is related to the interfaces from which the upward traveling waves have been reflected. The maximum angle rating determines the maximum dip in an interface (Fig. 3.7)

which can be estimated by the migration algorithm. The wavenumber k_x and the propagation angle from vertical θ are related as follows:

$$k_x = \alpha \sin\theta. \tag{3.57}$$

The dip angle $\theta = 90^o$ corresponds to the boundary between the propagating and evanescent regions.

Migration filters are commonly used to extrapolate 3-D seismic wave fields. The 3-D depth extrapolation is performed, one-frequency at a time, using 2-D migration filters $H(k_x, k_y)$ whose frequency response approximates (3.55) with k_x replaced by $\sqrt{k_x^2 + k_y^2}$. Very good 2-D approximations $H(k_x, k_y)$ can be efficiently obtained from the 1-D approximation $H(k_x)$ using McClellan transformations [48, 49, 50]. So, only the coefficients $\{h_n\}$ of the 1-D extrapolation filters (3.56) need to be actually computed.

3.5 Design Examples

In order to display the performance of the multiple-exchange approximation algorithm, this section presents several illustrative numerical examples and compares the results with other existing algorithms [22, 19]. In all the design examples, the real phase-rotated error (3.21) is computed by substituting for θ_δ in (3.21) the angle of the optimal error at the first extremal point.

Before presenting the design examples, a note about the frequency grid is in order. For implementation purposes, the continuous frequency bands are sampled and represented by a dense uniform grid. As a result, the original edge frequencies may not all fall on the used frequency grids. In practice, these grids are acceptable as a representation of the original frequency bands since the displacement (or rounding) of the edge frequencies is relatively small. However, the Chebyshev error norm is sensitive even to a small displacement of band edges. So, one cannot compare the error norms resulting from different algorithms unless the same frequency grids are

Table 3.1. Summary of the numerical results. $|\delta|$ is the lower bound on the optimal error and is computed by the complex Remez algorithm. $\|E_{opt}\|$ is the optimal Chebyshev error norm.

		$\|\delta\|$	$\|E_{opt}\|$	# extremals	# iterations	
					Stage 1	Stage 2
Nearly-linear Phase	Exp. 1 ($N = 32$)	0.0439	0.0439	34	11	0
	Exp. 1 ($N = 38$)	0.0268	0.0268	39	11	0
	Exp. 2 ($N = 25$)	0.0372	0.0372	26	11	0
	Exp. 2 ($N = 100$)	2.17e-5	2.17e-5	101	55	0
	Exp. 3 ($N = 31$)	0.0745	0.0754	35	9	4
	Exp. 4 ($N = 40$)	0.248	0.2505	43	8	3
Exactly-linear Phase	Exp 1 ($N = 32$)	0.04956	0.04956	17	8	0
	Exp. 2 ($N = 25$)	0.01665	0.01665	28	9	0
	Exp. 3 ($N = 31$)	0.1106	0.1106	17	7	0
	Exp. 4 ($N = 40$)	0.2407	0.2407	43	8	0
Seismic Migration	Exp. 1 ($N = 19$)	0.0498	0.0498	11	7	0
	Exp. 2 ($N = 35$)	0.7828	0.7828	19	7	0
	Exp. 2 ($N = 36$)	0.7526	0.7681	39	9	7
	Exp. 3 ($N = 31$)	0.9704	0.9755	18	9	5

used in these algorithms. The exact edge frequencies were used in all the designs by inserting the desired edge frequencies in the uniformly spaced grid. However, in order for the Haar condition to hold, the exact points $\omega = 0$ or $\omega = \pi$ cannot belong to the grid in some cases (see section 3.1). In these cases, an edge point at $\omega = 0$ or $\omega = \pi$ is represented by a neighboring point in the grid.

The multiple-exchange approximation algorithm used for the design examples of this section was entirely coded in MATLAB[1], and run on a multi-user SUN SPARC 10 workstation. The design specifications are given in terms of the normalized frequency $f = \omega/2\pi$. The numerical results for the presented design examples are summarized in Table 3.1.

[1]MATLAB is a trademark of *The MathWorks Inc.*

3.5.1 Reduced Delay, Nearly Linear-Phase FIR Filters

In this section, nearly linear-phase filters are designed. The term "reduced delay" refers to the fact that the desired group delay is set to be smaller than $(N-1)/2$, where N is the specified filter length. Note that $(N-1)/2$ is the group delay of the exactly linear-phase filter.

Example 1: nearly linear-phase, causal, lowpass filter

The design specifications are as follows:

$$\text{Desired: } D(f) = \begin{cases} e^{-j(2\pi f)12.5} & \text{if } 0 \le |f| \le 0.06 \\ 0 & \text{if } 0.12 \le |f| \le 0.5 \end{cases}$$

$$\text{Weight: } W(f) = \begin{cases} 1 & \text{if } |f| \le 0.06 \\ 10 & \text{if } 0.12 \le |f| \le 0.5 \end{cases}$$

$$\text{Approximating filter length: } N = 32.$$

This example corresponds to a reduced group delay, nearly linear-phase, and causal lowpass filter. Similar designs were considered in [5, 6, 19, 17]. The complex Remez stage of the multiple-exchange algorithm converged to the optimal solution in 11 iterations. The resulting optimal filter has an optimal error norm $\|E_{opt}\| = |\delta| = 0.0439$. The filter characteristics are illustrated in Fig. 3.8.

The FIR filter group delay (Fig. 3.8(b)) corresponds to the nearly linear-phase characteristic in the passband. Note that the optimal error (Fig. 3.8(c)) assumes its maximum value at $N+2$ extremal points with an alternating phase shift of π. This alternation can be clearly seen from the plot of the real phase-rotated error $E_r(f)$ (Fig. 3.8(c)). The progression of the algorithm is shown in Fig. 3.9. Note that, although the Chebyshev error norm $\|E\|$ is not monotonically decreasing, the magnitude of the subset error norm δ is monotonically increasing. They both converge to the optimal value at the last iteration. It is important to note that $\|E_{opt}\|$ *cannot*

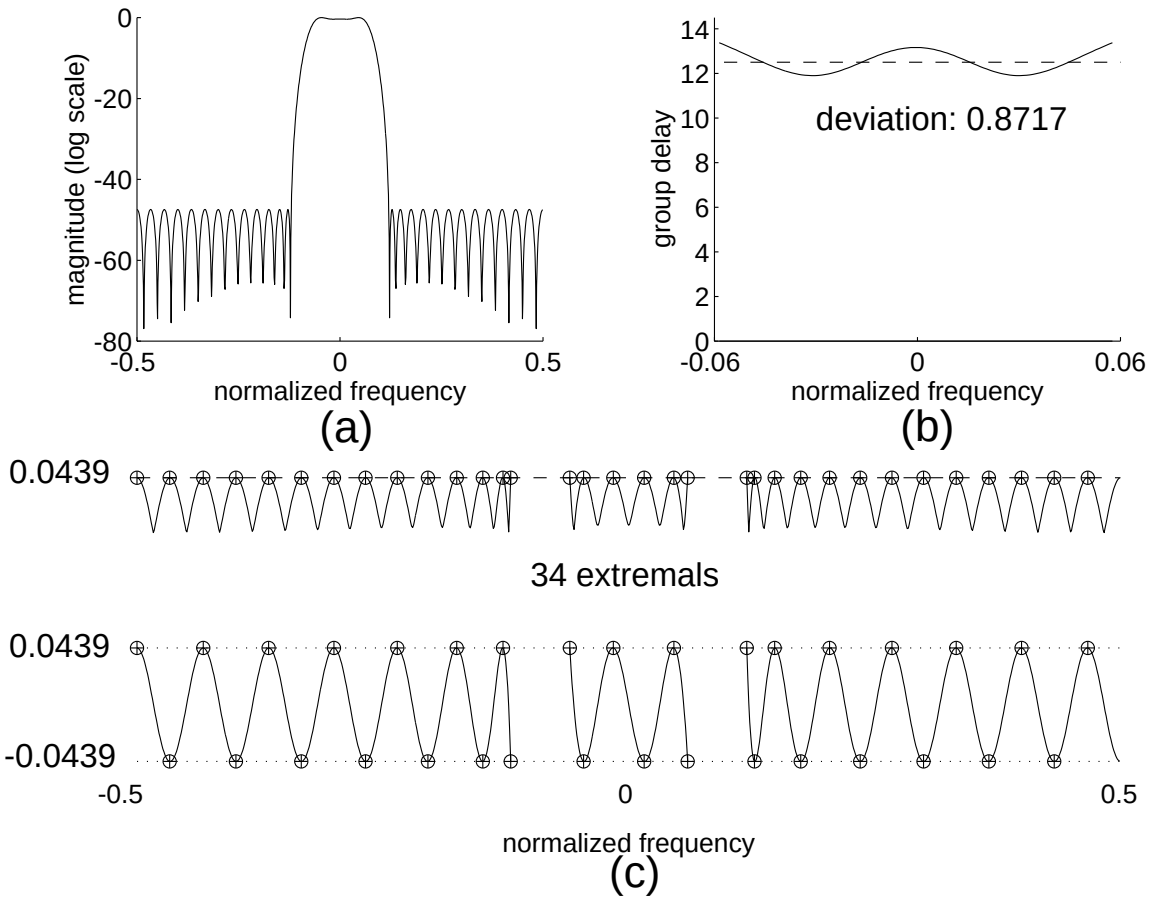


Figure 3.8. Example 1–A: nearly linear-phase filter, $N = 32$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

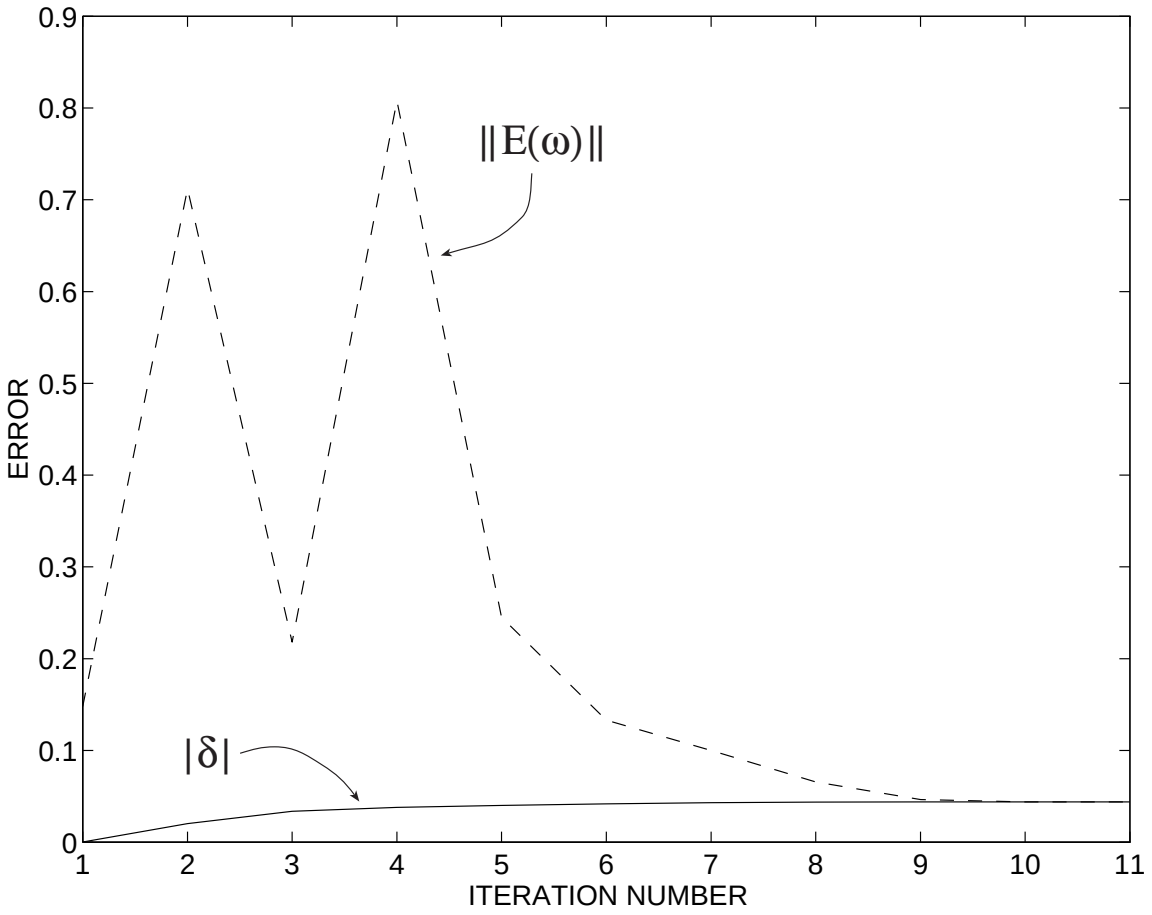


Figure 3.9. Algorithm progression - Example 1 (nearly linear phase filter $N = 32$). The solid and dashed curves represent the values of $|\delta|$ and $\|E(\omega)\|$ respectively.

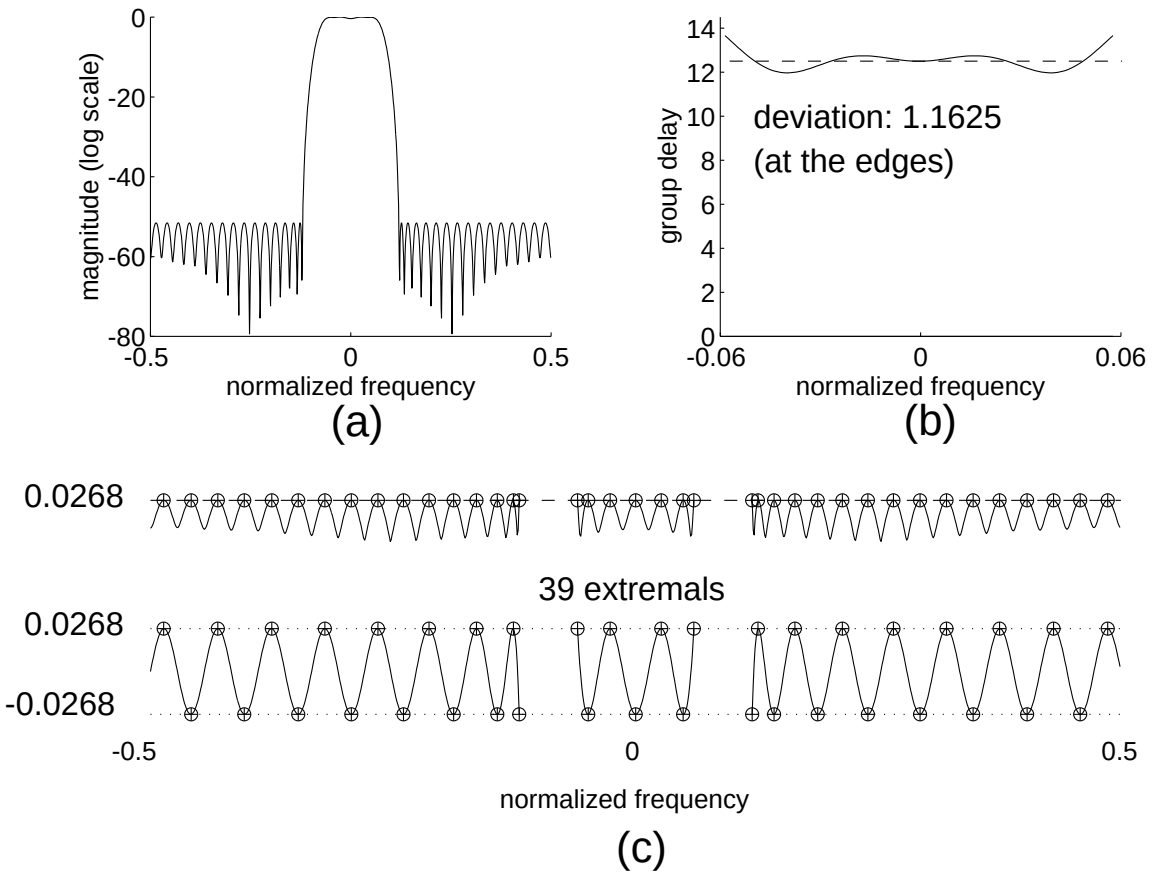


Figure 3.10. Example 1–B: nearly linear-phase filter, $N = 38$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

be less than $|\delta|$ since $|\delta|$ is a lower bound of $\|E_{opt}\|$ as given in (2.32). The design took 1 second to complete on a multi-user SUN SPARC 10 workstation.

Figure 3.10 shows the optimal filter characteristics when the filter length is increased to $N = 38$. The optimal solution was obtained in 11 iterations of the complex Remez initial stage (less than 2 seconds on a multi-user SUN SPARC 10). The optimal deviation is $\|E_{opt}\| = |\delta| = 0.0268$. Note that the increase in the filter length results in a better approximation of the constant group delay in the passband. However, for this case, a larger phase deviation is obtained near the frequency edges.

Example 2: nearly linear-phase, causal, bandpass filter

This example corresponds to a nearly linear-phase and causal bandpass filter with two passbands and different passband group delays. It was introduced in [22] with the following design specifications:

$$\text{Desired: } D(f) = \begin{cases} 0 & \text{if } -0.5 \le f \le -0.4 \\ e^{-j(2\pi f)10} & \text{if } -0.3 \le f \le -0.2 \\ 0 & \text{if } -0.1 \le f \le 0 \\ 2e^{-j(2\pi f)8} & \text{if } 0.1 \le f \le 0.4 \end{cases}$$

$$\text{Weight: } W(f) = \begin{cases} 1 & \text{in passbands} \\ 10 & \text{in stopbands} \end{cases}$$

Approximating filter length: $N = 25$.

It should be noted that an exactly linear-phase filter would have a group delay of 12 in both passbands (see section 3.5.2). The complex Remez stage of the multiple-exchange algorithm converged to the optimal solution in 11 iterations with $\|E_{opt}\| = |\delta| = 0.037$. The filter characteristics are illustrated in Fig. 3.11. The different desired group delays are shown in Fig. 3.11(b). The design took 1 second to complete on a multi-user SUN SPARC 10 workstation.

With an approximating filter length of $N = 100$, the resulting optimal filter has $\|E_{opt}\| = 2.17e{-}5$ and the design took 55 iterations of the complex Remez stage to

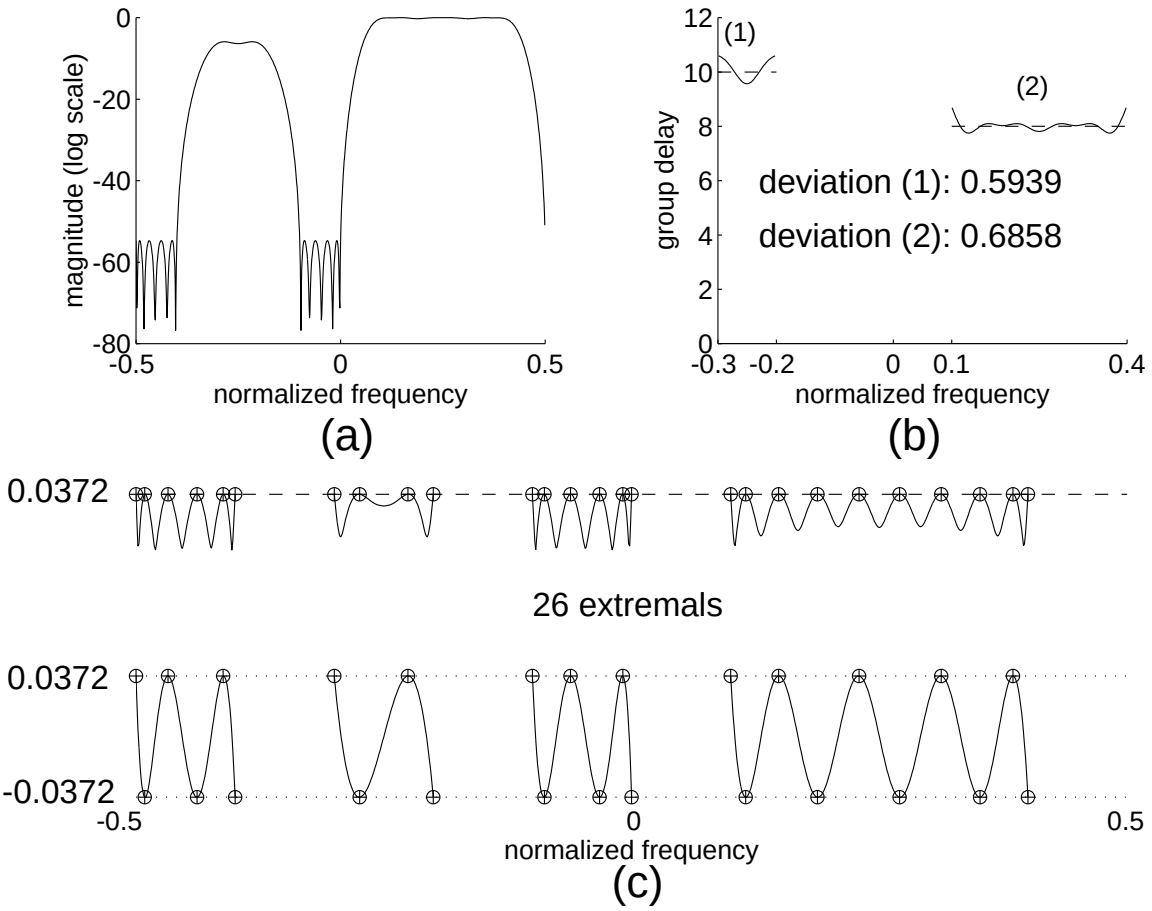


Figure 3.11. Example 2: nearly linear-phase filter, $N = 25$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

complete (33 seconds on a multi-user SUN SPARC 10 workstation). The resulting phase deviations are 0.0025 and 0.0013 in the first and second passbands, respectively.

Example 3: nearly linear-phase, causal, bandpass filter

This example corresponds to a reduced group delay, nearly linear-phase, and causal bandpass filter with a single passband and two stopbands. It was considered in [5]. The design specifications are as follows:

$$\text{Desired: } D(f) = \begin{cases} e^{-j(2\pi f)12} & \text{if } 0.15 \le |f| \le 0.28 \\ 0 & \text{if } 0 \le |f| \le 0.1 \text{ and } 0.33 \le |f| \le 0.5 \end{cases}$$

$$\text{Weight: } W(f) = \begin{cases} 1 & \text{if } 0.15 \le |f| \le 0.28 \\ 10 & \text{if } 0 \le |f| \le 0.1 \text{ and } 0.33 \le |f| \le 0.5 \end{cases}$$

$$\text{Approximating filter length: } N = 31.$$

The multiple-exchange algorithm converged to the optimal solution in 9 iterations of the first stage and 4 iterations of the second stage. The resulting optimal filter has an optimal error norm $\|E_{opt}\| = 0.0754$. The lower bound $|\delta|$ computed by the complex Remez initial stage is 0.0745. The filter characteristics are illustrated in Fig. 3.12. The optimal error has 35 ($= N + 4$) extremal points. From the real rotated error in Fig. 3.12(c), it can be observed that the error alternates on the extremal frequencies in [-0.5,0].

Example 4: nearly linear-phase, causal, lowpass filter

The design specifications are as follows:

$$\text{Desired: } D(f) = \begin{cases} e^{-j(2\pi f)15} & \text{if } 0 \le |f| \le 0.23 \\ 0 & \text{if } 0.25 \le |f| \le 0.5 \end{cases}$$

$$\text{Weight: } W(f) = \begin{cases} 1 & \text{if } |f| \le 0.23 \\ 10 & \text{if } 0.25 \le |f| \le 0.5 \end{cases}$$

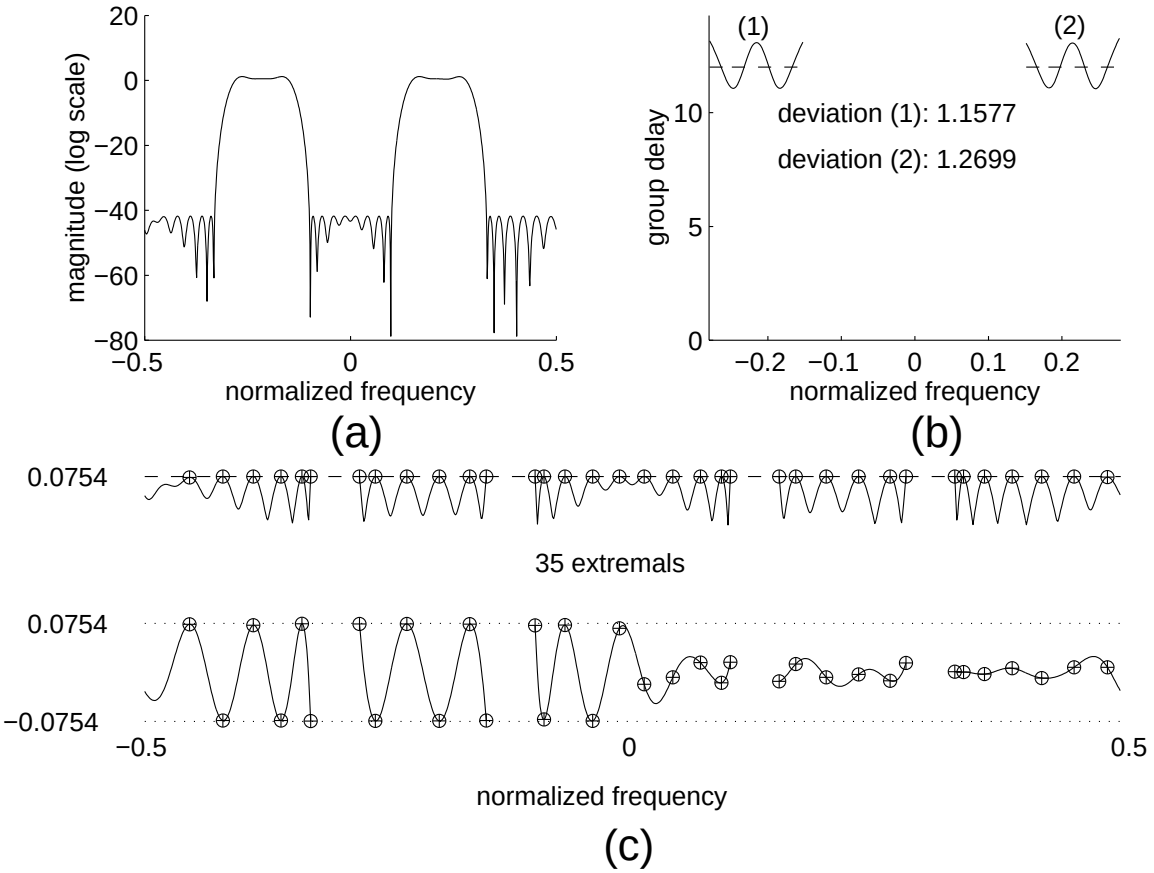


Figure 3.12. Example 3: nearly linear-phase filter, $N = 31$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

Approximating filter length: $N = 40$.

This example corresponds to a reduced group delay, nearly linear-phase, causal low-pass filter. The group delay in the passband is set to 15 whereas an exactly linear filter would have a delay of 19.5. The multiple-exchange algorithm converged to the optimal solution in 8 iterations of the first stage and 3 iterations of the second stage. The resulting optimal filter has an optimal error norm $\|E_{opt}\| = 0.2505$. The filter characteristics are illustrated in Fig. 3.13. The phase deviation, shown in Fig. 3.13(b), is relatively large for this example because the even-length filter is approximating an integer desired delay 15 (no half-sample phase shift). From the real rotated error plot in Fig. 3.13(c), the error extremals appear to alternate indicating the complex Remez algorithm should have resulted in the optimal solution on the desired frequency domain. However, Figure 3.14 plots the angle of the optimal error at the extremal points and shows that the error is *not* alternating at its extremals but its angle is a perturbed version of a truly alternating angle (quasi-alternating behavior).

For this example, the complex Remez algorithm resulted in a solution optimal on 99% of the desired frequency bands. The characteristics of the sub-optimal filter resulting from the complex Remez stage are shown in Fig. 3.15. It should be noted that, although the error deviation is large on a very small subset near the edge frequencies (Fig. 3.15(c)), the phase deviation in the passband (Fig. 3.15(b)) is smaller than the resulting optimal filter. The perfect alternating property of the error at the resulting $N + 1$ maxima (represented by circles in Fig. 3.15(c)) is illustrated by Fig. 3.16. Observe the fact that the local maxima of the error resulting from the complex Remez stage (Fig. 3.15(c)) correspond to the extremals (absolute maxima) of the optimal error (Fig. 3.13(c)).

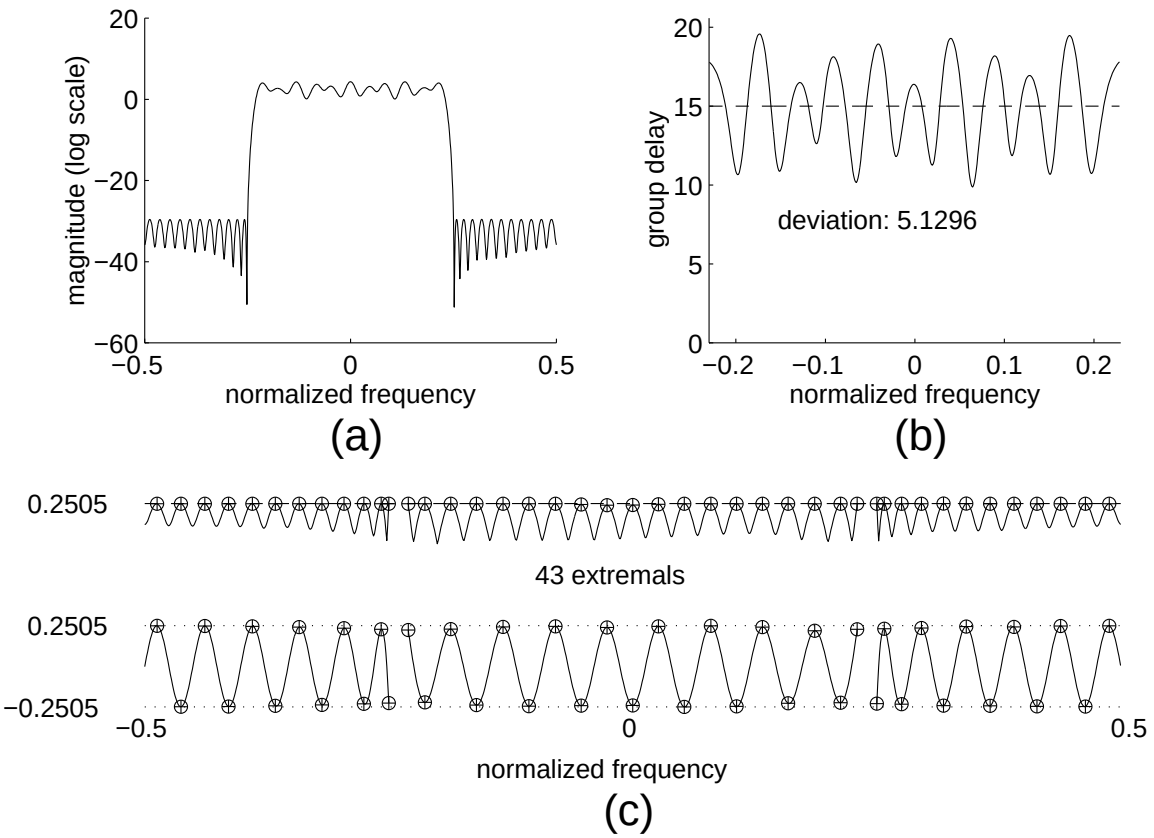


Figure 3.13. Example 4: nearly linear-phase filter, $N = 40$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

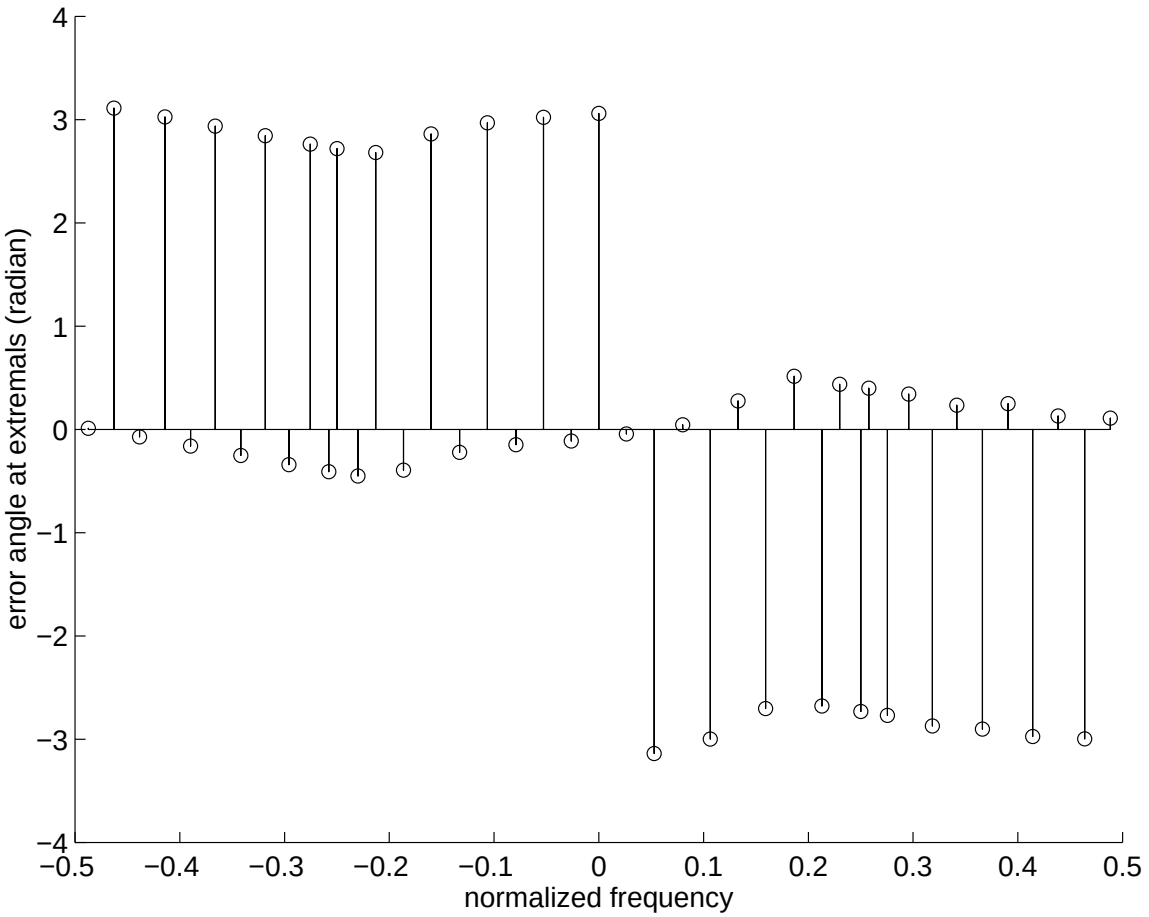


Figure 3.14. Example 4: nearly linear-phase filter, $N = 40$. Angle of the optimal error at the extremal points indicating a quasi-alternating behavior.

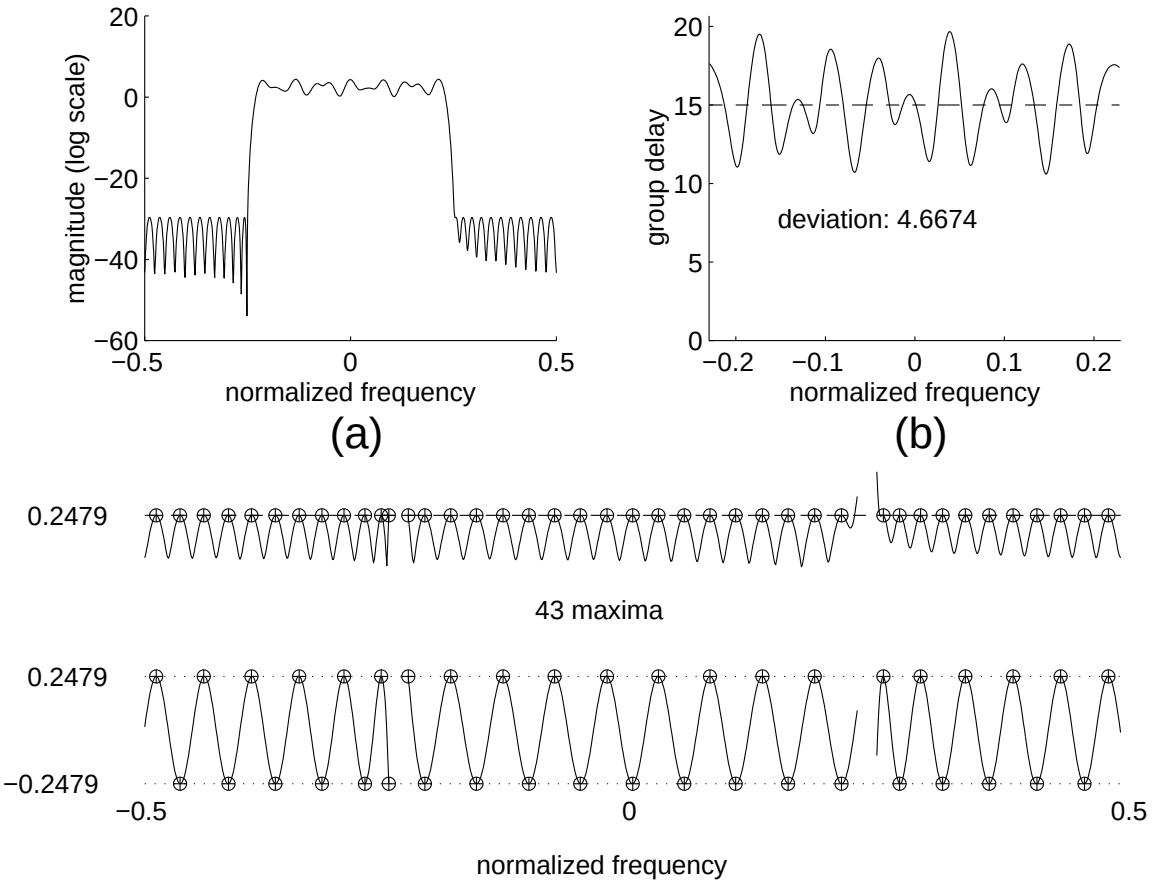


Figure 3.15. Example 4: nearly linear-phase filter, $N = 40$. Characteristics of the sub-optimal filter computed by the efficient complex Remez algorithm. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

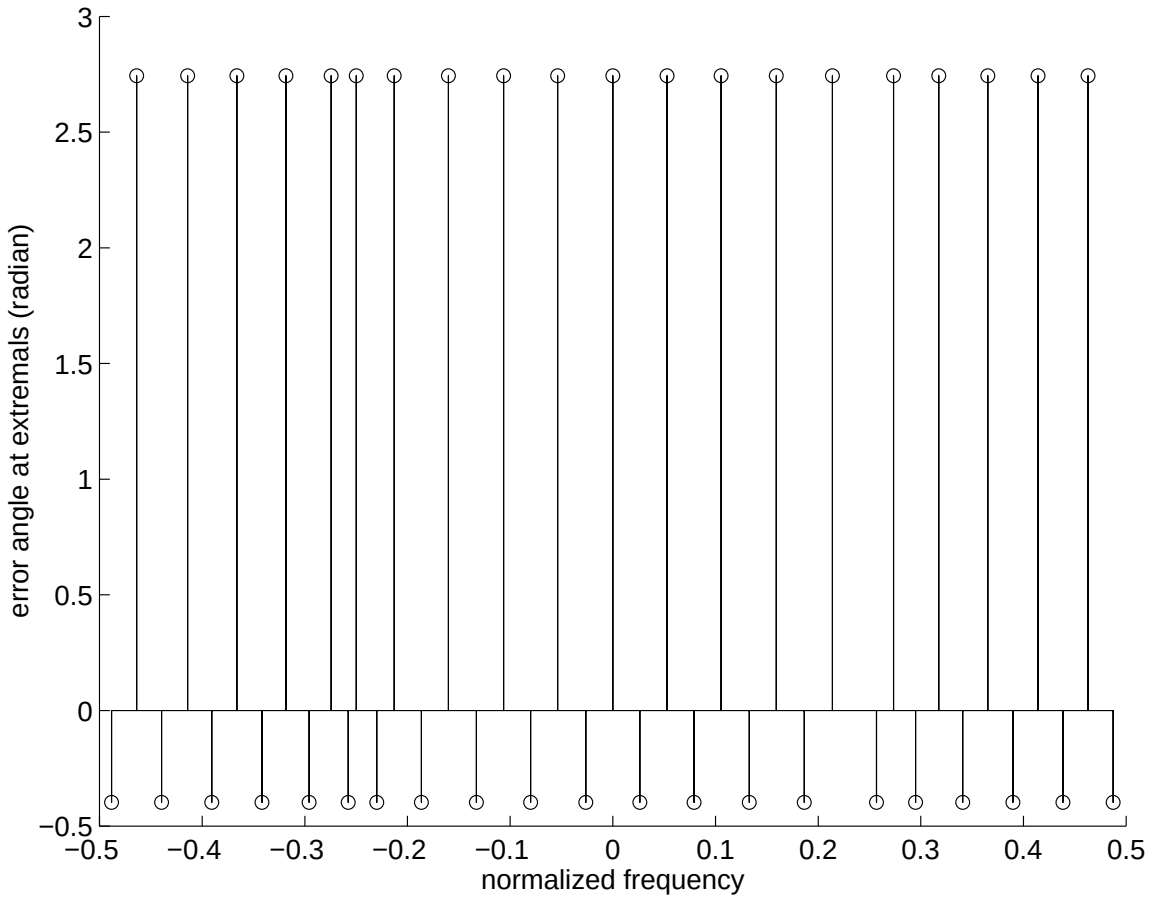


Figure 3.16. Example 4: nearly linear-phase filter, $N = 40$. Angle of the error, which is sub-optimal on B and optimal on a reduced set $B' \subset B$ (see Fig. 3.15(c)), at the extremals on B'. A perfect alternation behavior is exhibited.

3.5.2 Exactly Linear-Phase FIR Filters

This section designs the exactly linear-phase filters corresponding to the nearly linear-phase filters of section 3.5.1. The same design specifications are used except that the delay is set to be equal to $(N-1)/2$, where N is the specified FIR filter length.

Example 1: exactly linear-phase, causal, lowpass filter

This example has the same specifications as Example 1 of section 3.5.1 ($N = 32$) but the desired group delay is set to 15.5 in the passband. The complex Remez algorithm (first stage) converged to the optimal solution in 8 iterations (less than 1 second on a SUN SPARC 10). The resulting optimal filter has an optimal error norm $\|E_{opt}\| = |\delta| = 0.04956$ which is *larger* than the corresponding reduced delay, nearly linear-phase filter in section 3.5.1, which indicates that a nearly linear-phase filter not only can have a reduced group delay but also a reduced Chebyshev error norm. The filter characteristics are illustrated in Fig. 3.17. Note that, although the phase specifications were explicitly included as part of the approximation problem, the exactly linear-phase optimal solution was obtained as expected.

The same unique optimal solution would result if the linear phase term was removed and the approximating filter was specified to be noncausal. It is easy to verify that the resulting Chebyshev approximation problem is equivalent to the original design problem.

Example 2: exactly linear-phase, causal, bandpass filter

This example has the same specifications as Example 2 ($N = 25$) of section 3.5.1 except that the two passband group delays are equal to that of an exactly linear-phase filter, i.e. $(N-1)/2 = 12$. The complex Remez algorithm (first stage) converged to the optimal solution in 9 iterations (less than 1 second on a SUN SPARC 10) with an optimal error $\|E_{opt}\| = |\delta| = 0.01665$. Note that, in this case, the exactly linear-phase design resulted in a smaller Chebyshev error than the corresponding nearly

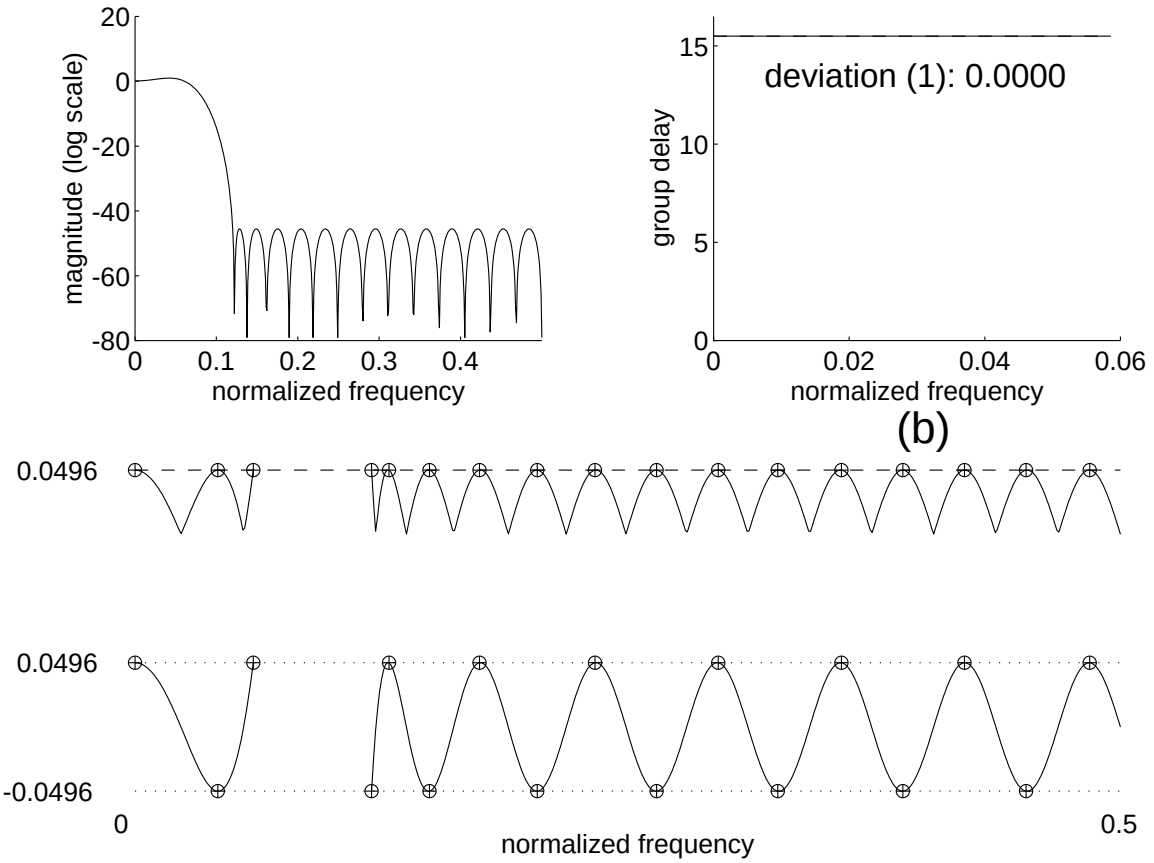


Figure 3.17. Example 1: exactly linear-phase lowpass filter, $N = 32$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

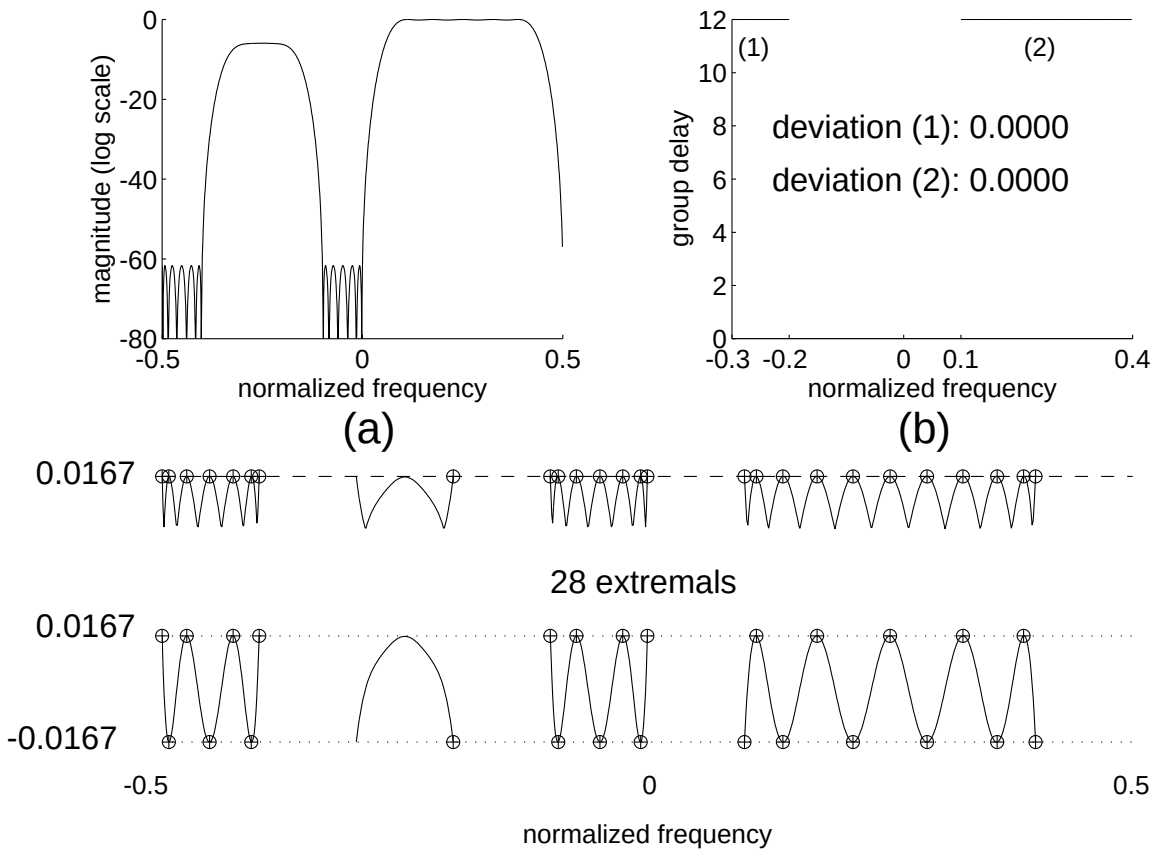


Figure 3.18. Example 2: exactly linear-phase passband filter, $N = 25$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

linear-phase filter (Example 2, section 3.5.1). The optimal filter characteristics are illustrated in Fig. 3.19. Figure 3.19(c) shows that the optimal error alternates on $N+3$ extremal points. The passband group delays are shown in Fig. 3.18(b). Although the desired group delays were explicitly included in the design specifications, the exactly linear-phase optimal filter was obtained. The same solution would result if the linear phase term was removed and the approximating filter was specified to be noncausal.

Example 3: exactly linear-phase, causal, bandpass filter

This example has the same specifications as Example 3 of section 3.5.1 except that the passband group delay is set to $(N-1)/2 = 15$. The complex Remez algorithm (first stage) computed the optimal solution in 7 iterations (0.5 second). The resulting optimal deviation is $\|E_{opt}\| = |\delta| = 0.111$ which is larger than the corresponding nearly linear-phase filter. The optimal filter characteristics are shown in Fig. 3.19.

Example 4: exactly linear-phase, causal, lowpass filter

This example has the same specifications as Example 4 of section 3.5.1 except that the desired group delay is set to the exact linear-phase filter delay, i.e., $(N-1)/2 = 19.5$. The properties of the optimal filter computed by the complex Remez algorithm are displayed in Fig. 3.20. This optimal filter was obtained in 8 iterations of the first stage (0.5 second). The resulting Chebyshev error norm is $\|E_{opt}\| = |\delta| = 0.241$.

3.5.3 Seismic Migration Filters

Example 1

This example corresponds to a non-causal seismic migration filter. The design specifications are as follows:

$$\text{Desired: } D(f) = \begin{cases} e^{j(0.53)\sqrt{(1.9)^2-(2\pi f)^2}} & , |f| \le 1.9/(2\pi) \\ 0 & , 0.4 \le |f| \le 0.5 \end{cases}$$

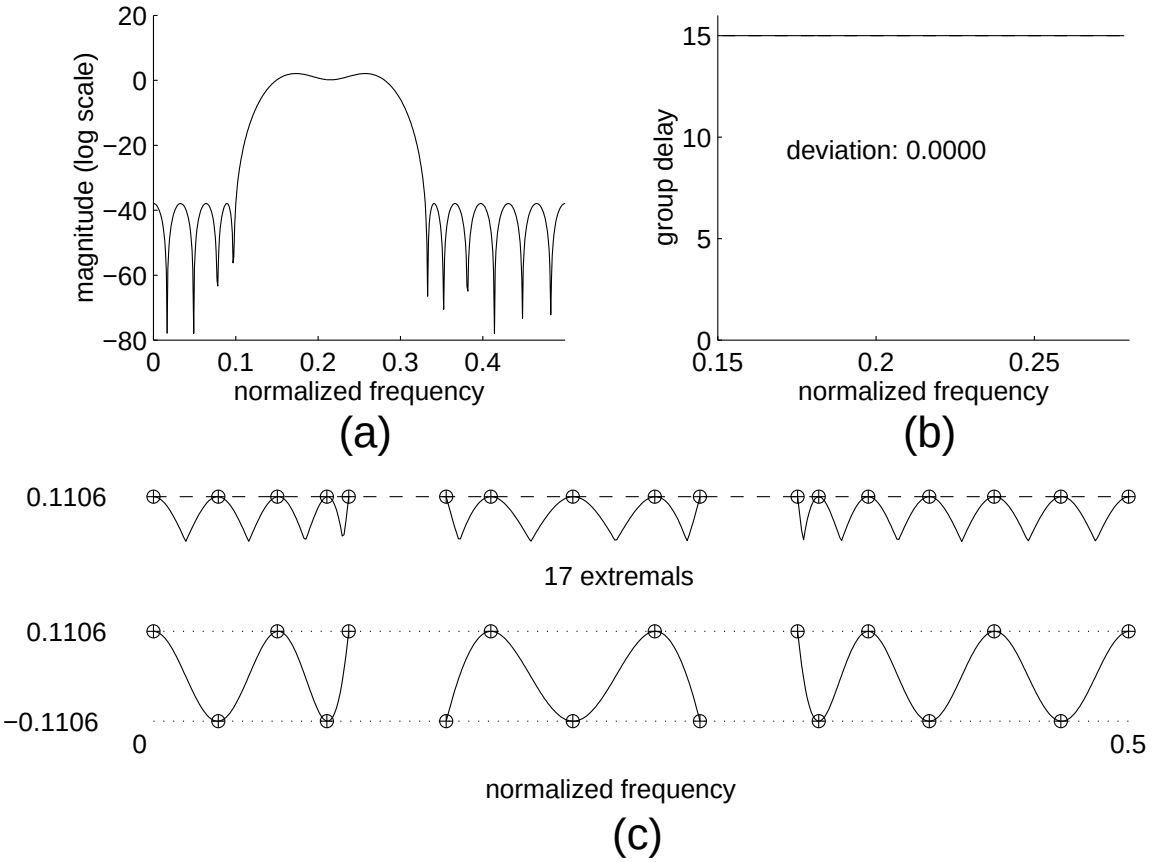


Figure 3.19. Example 3: exactly linear-phase passband filter, $N = 31$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

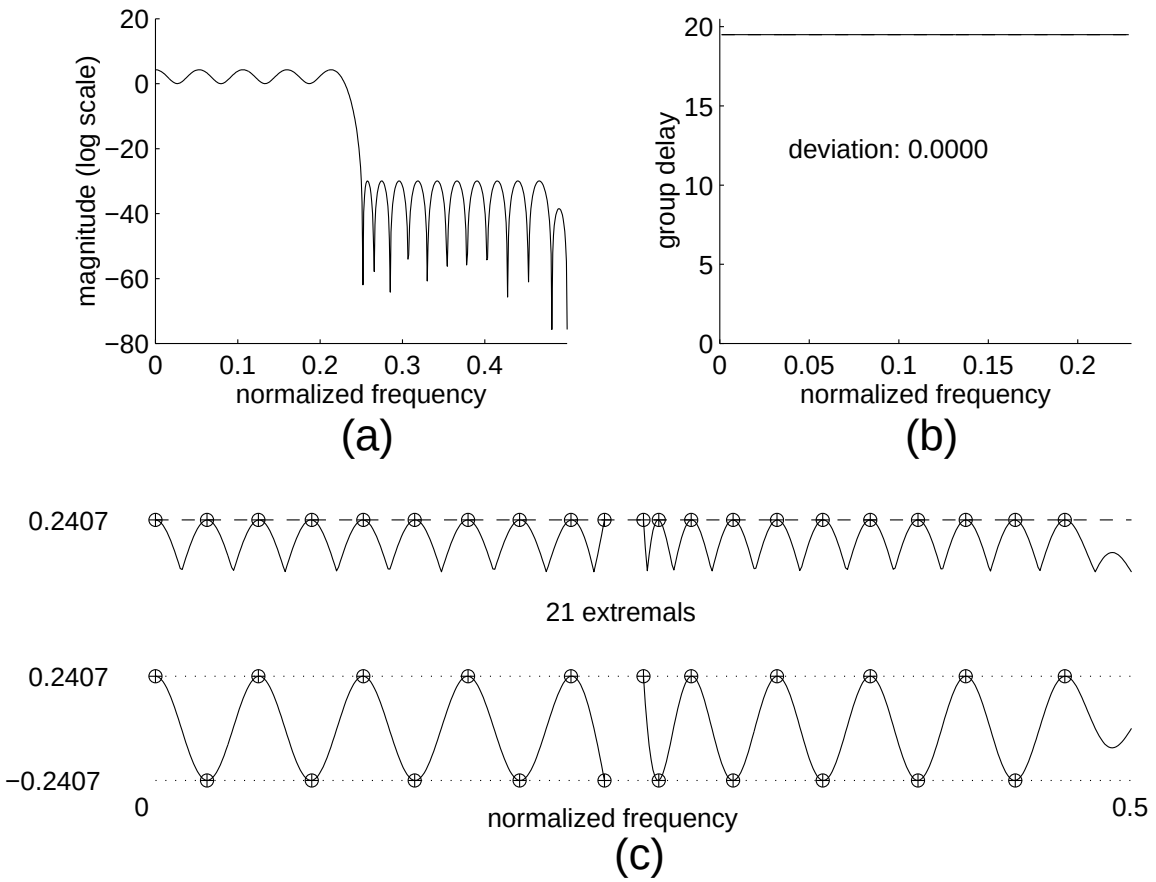


Figure 3.20. Example 4: exactly linear-phase passband filter, $N = 40$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband group delays. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

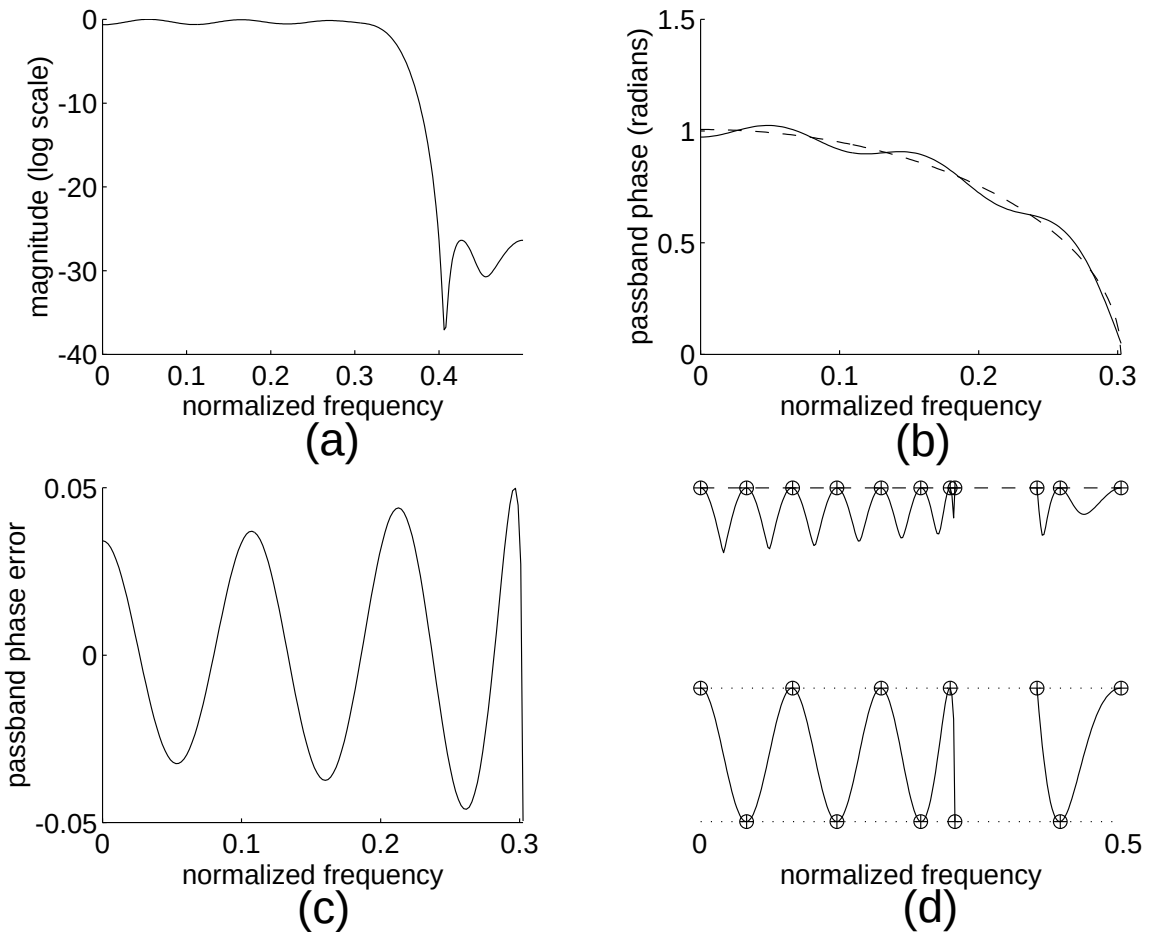


Figure 3.21. Example 1: seismic migration filter, $N = 19$. (a) Magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Phase error in passband. (d) Magnitude of the error (top), and alternating real rotated error (bottom).

Approximating filter length: $N = 19$.

The starting impulse response index is $N_1 = -9$. No weighting function is used for this example.

The complex Remez algorithm (first stage) converged to the optimal solution in 7 iterations. The optimal filter properties are shown in Fig. 3.21. Since $D(f)$ is symmetric with respect to $f = 0$, the resulting optimal filter is also symmetric with an optimal error $\|E_{opt}\| = |\delta| = 0.0498$, and the design specifications need only be given

on $B \subseteq [0, 0.5]$. It follows that only $n = (N+1)/2$ cosine basis functions are needed for the approximation. Figure 3.21 shows that the optimal complex error alternates with $(N+3)/2$ $(= n+1)$ extremal points as dictated by the Haar condition. The design took half a second to complete on a multi-user SUN SPARC 10 workstation.

Example 2

In this example, a non-causal seismic migration filter is designed with the following specifications:

$$\text{Desired: } D(f) = \begin{cases} e^{j\sqrt{(2)^2-(2\pi f)^2}} & \text{if } |f| \le 2\sin(75^o)/2\pi \\ 0 & \text{if } 2/2\pi \le |f| \le 0.5 \end{cases}$$

$$\text{Weight: } W(f) = \begin{cases} 50 & \text{if } |f| \le 2\sin(75^o)/2\pi \\ 1 & \text{if } 2/2\pi \le |f| \le 0.5 \end{cases}$$

$$\text{Filter starting index: } \begin{cases} N_1 = -17 & \text{for } N = 35 \\ N_1 = -18 & \text{for } N = 36 \end{cases}$$

where $2\sin(75^o)/2\pi \approx 0.3075$ and $2/2\pi \approx 0.3183$. It should be noted that odd-length FIR filters (i.e., N odd) are normally used to approximate the positive-symmetric migration filters.

The complex Remez algorithm (first stage) converged to the optimal solution in 7 iterations for $N = 35$ with an optimal deviation $\|E_{opt}\| = 0.7828$. For $N = 36$, the multiple-exchange algorithm obtained the optimal solution in 9 iterations of Stage 1 and 7 iterations of Stage 2 with $\|E_{opt}\| = 0.7681$. Figures 3.22 and 3.23 show the resulting optimal filters when $N = 35$ and $N = 36$, respectively.

For $N = 35$, $A(\omega)$ given by (3.1) is symmetric with respect to the point $\omega = 0$. Therefore, the resulting non-causal optimal filter is also symmetric with respect to $\omega = 0$. In this case, the optimal solution was obtained by the efficient complex Remez method (initial stage) with $\|E_{opt}\| = |\delta| = 0.7828$. In this case, only $n = (N+1)/2$ *cosine* basis functions are needed for the approximation, and the design specifications

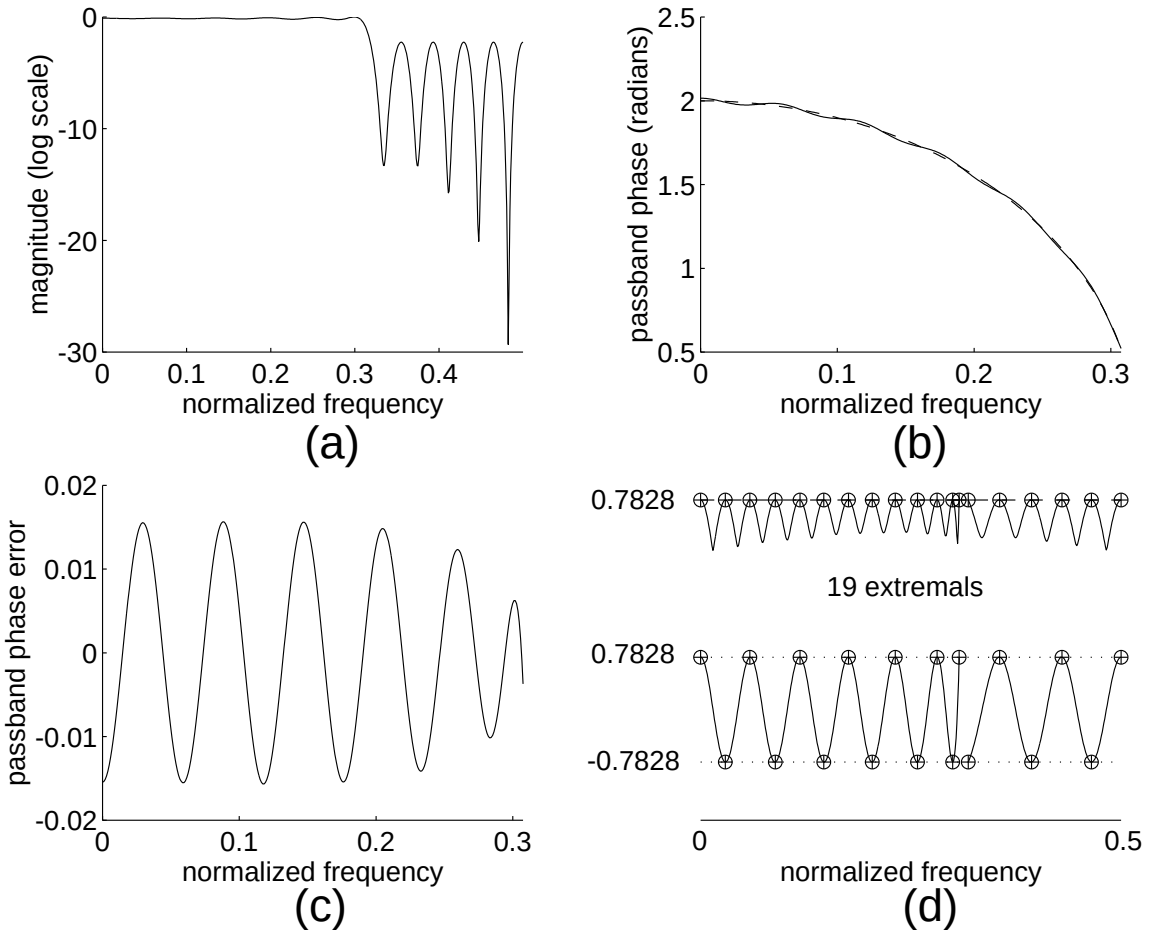


Figure 3.22. Example 2: seismic migration filter, $N = 35$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Phase error in passband. (d) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

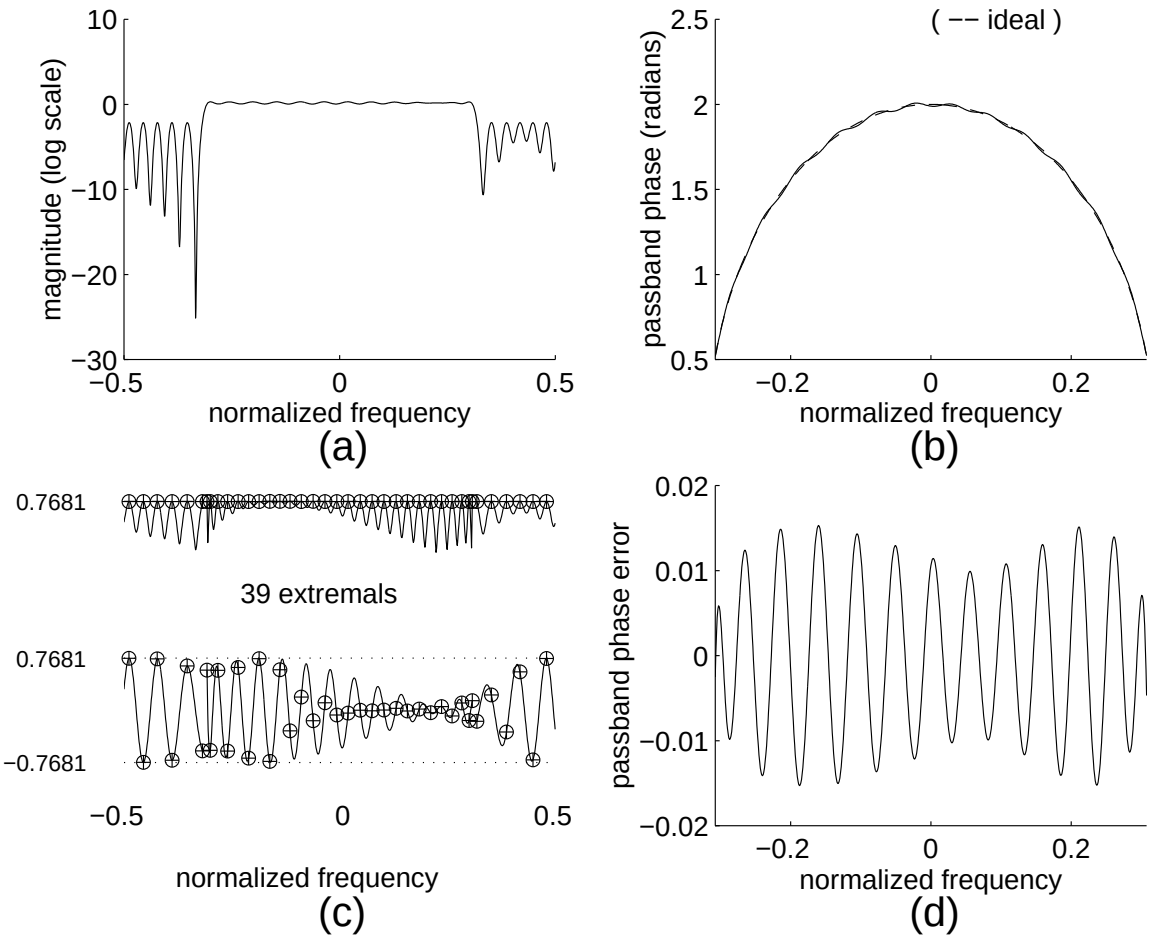


Figure 3.23. Example 2: seismic migration filter, $N = 36$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Phase error in passband. (d) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

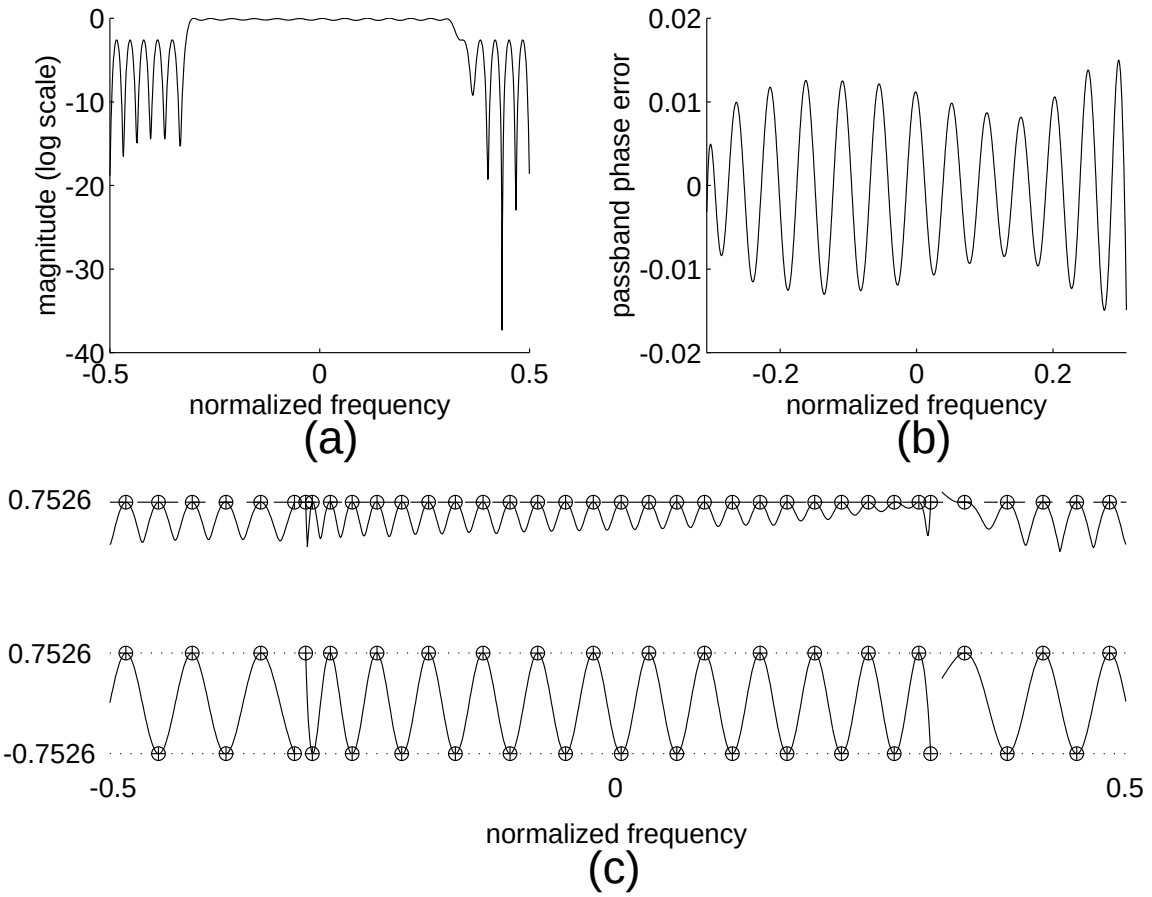


Figure 3.24. Example 2: seismic migration filter, $N = 36$. Initial sub-optimal solution computed by the complex Remez method. (a) FIR filter magnitude response in dB. (b) Phase error in passband. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom).

need only be given on $B \subset [0, 0.5]$. This symmetry property does not hold when N is even since $A(\omega) = e^{-j0.5\omega}D(\omega)$ is not symmetric (Section 3.1).

When $N = 36$, the initial complex Remez stage converged to a sub-optimal solution in 9 iterations. The computed sub-optimal solution is optimal on the reduced set

$$\begin{aligned} B' =& [-0.5, -0.3183] \cup [-0.3075, 0.3075] \cup [0.3350, 0.4995] \\ =& \{\frac{\omega}{2\pi} \in B : |E| \leq |\delta|\} \end{aligned}$$

which includes the passband and constitutes about 99% of the set of interest B. The optimal deviation on B' is $|\delta| = 0.7526$. In the case of migration filter design, the approximation need only be accurate in the passband (propagating region) [4] as long as the stopband (evanescent region) deviation is not much larger than unity. Moreover, the optimal error $|\delta|$ on $B' \subset B$ is smaller than the optimal Chebyshev error on the whole set of interest B, resulting in a smaller passband error. The properties of the resulting sub-optimal filter (optimal on B' and sub-optimal on B) are illustrated in Fig. 3.24.

Example 3

This example corresponds to a non-causal seismic migration filter with the following specifications:

$$\text{Desired: } D(f) = \begin{cases} e^{j\sqrt{(2)^2-(2\pi f)^2}} & \text{if } |f| \leq 2\sin(75^\circ)/2\pi \\ 0 & \text{if } 2/2\pi \leq |f| \leq 0.5 \end{cases}$$

$$\text{Weight: } W(\omega) = \begin{cases} 500 & \text{if } |f| \leq 2\sin(75^o)/2\pi \\ 1 & \text{if } 2/2\pi \leq |f| \leq 0.5 \end{cases}$$

$$\text{Approximating filter length: } N = 31$$

The starting impulse response index is $N_1 = -15$. The large passband weighting is used to force almost perfect match in the passband. In fact, for migration filters,

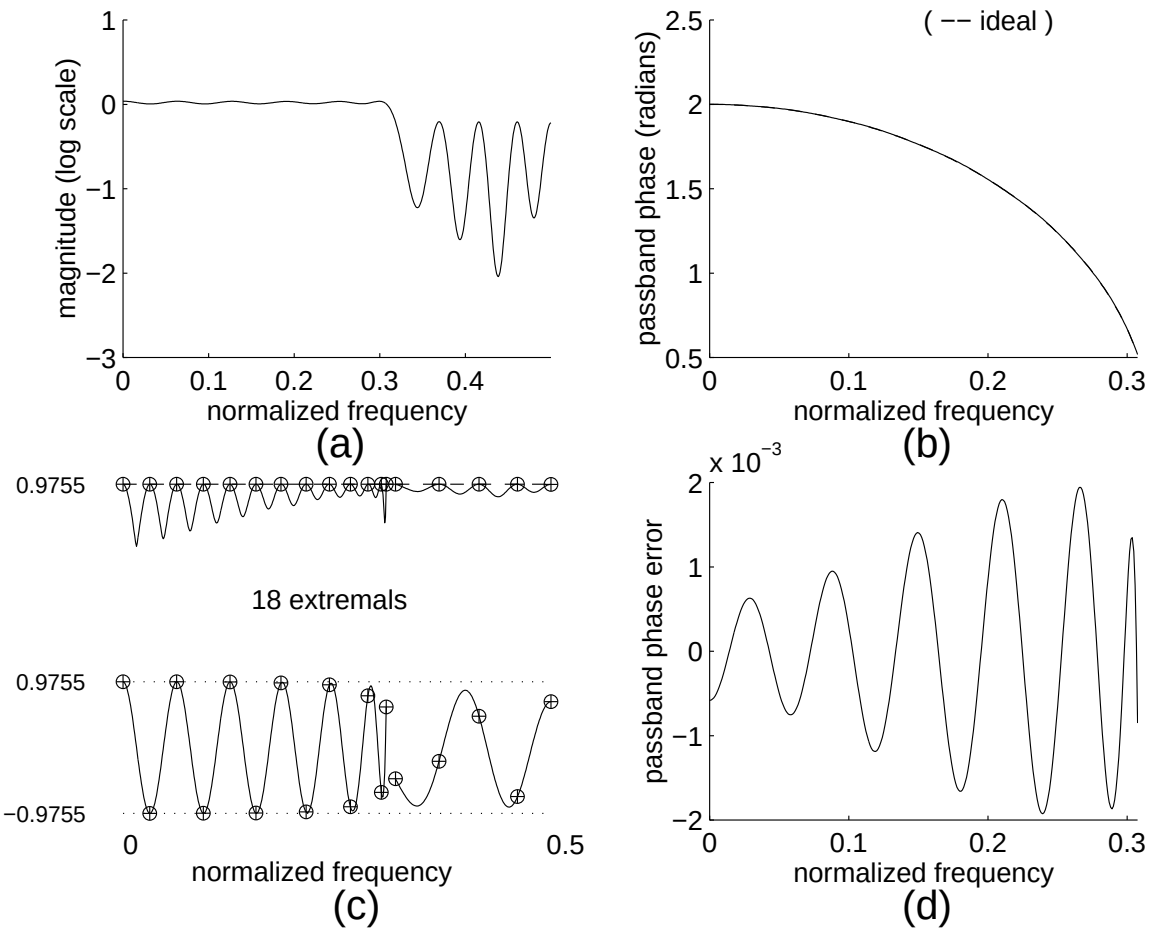


Figure 3.25. Example 3: seismic migration filter, $N = 31$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Magnitude of the weighted error (top), and real phase-rotated weighted error (bottom). (d) Phase error in passband.

the approximation need only be accurate in the passband as long as the stopband magnitude deviation is not larger than unity [4, p. 361].

The optimal solution was obtained in 9 iterations of Stage 1 and 5 iterations of Stage 2. The filter properties are shown in Fig. 3.25. Since $D(f)$ is symmetric with respect to $f = 0$, the resulting optimal filter is also symmetric with an optimal error $\|E_o\|$=0.9755. The lower bound $|\delta|$, which is computed by the complex Remez method, is 0.9704.

3.6 Comparison with other existing algorithms

A comparison of the running time of the proposed algorithm with the Burnside-Parks [22] and Preuss-Schulist [19] algorithms is given in this section. The latest available versions of these algorithms were obtained from their respective authors. Note that, while the Preuss-Schulist algorithm and the proposed algorithm are fully coded in MATLAB, the Burnside-Parks algorithm is partially coded in MATLAB with the main design procedure written in the C language. Compared to a MATLAB-generated code, the C language results in a significantly faster code, especially when control and loop structures are used (which is the case here). In what follows, the terms *simple iterations* and *major iterations* are used to refer to the iterations of the complex Remez method (first stage) and the ascent-descent method (second stage), respectively.

The timing comparison is done with both [22, 19] in terms of number of seconds when only the complex Remez method (first stage) is executed, and with [22] in terms of the number of major iterations when the ascent-descent algorithm is executed. The reason is that, like the ascent-descent method, the algorithm in [22] progresses in major iterations, each of which consists of several simplex iterations [22]. A major iteration of the ascent-descent method has roughly a computational complexity comparable to one major iteration of the algorithm in [22](see section 3.2.3).

Table 3.2. Running time comparison with the Burnside-Parks [22] and Preuss-Schulist [19] algorithms run on a SUN SPARC 10 workstation.

		New Algorithm (MATLAB)		Burnside-Parks [22] (C & MATLAB)		Preuss-Schulist [19] (MATLAB)	
		major iter.	$\|E_{opt}\|$	major iter.	$\|E_{opt}\|$	major iter.	$\|E_{opt}\|$
Nearly-linear Phase	Exp. 1 ($N = 32$)	0 (1 sec.)	0.0439	8 (3 sec.)	0.0442	0 (7 sec.)	0.0439
	Exp. 1 ($N = 38$)	0 (2 sec.)	0.0268	11 (4 sec.)	0.0270	0 (10 sec.)	0.0268
	Exp. 2 ($N = 25$)	0 (1 sec.)	0.0372	8 (5 sec.)	0.0369	—	—
	Exp. 2 ($N = 100$)	0 (33 sec.)	2.17e-5	13 (720 sec.)	2.13e-5	—	—
	Exp. 3 ($N = 31$)	4	0.0754	11	0.0754	—	—
	Exp. 4 ($N = 40$)	3	0.2505	9	0.2515	—	—
Exactly-linear Phase	Exp 1 ($N = 32$)	0 (1 sec.)	0.04956	7 (2 sec.)	0.0497	0 (10 sec.)	0.04956
	Exp. 2 ($N = 25$)	0 (1 sec.)	0.01665	11 (7 sec.)	0.0168	—	—
	Exp. 3 ($N = 31$)	0 (0.5 sec.)	0.1106	7 (2 sec.)	0.1109	0 (8 sec.)	0.1106
	Exp. 4 ($N = 40$)	0 (0.5 sec.)	0.2407	9 (4 sec.)	0.2417	—	—
Seismic Migration	Exp. 1 ($N = 19$)	0 (0.5 sec.)	0.0498	13 (4 sec.)	0.416	0 (11 sec.)	0.0498
	Exp. 2 ($N = 35$)	0 (1 sec.)	0.7828	10 (31 sec.)	0.7877	—	—
	Exp. 2 ($N = 36$)	7	0.7681	10	0.7709	—	—
	Exp. 3 ($N = 31$)	5	0.9755	10	0.9797	—	—

On the other hand, the simple iterations of the complex Remez algorithm and of the method presented in [19] are not comparable to the aforementioned major iterations. In fact, a complete execution of the complex Remez algorithm produces significantly less overhead than a single major iteration. It is also important to note that the algorithm in [22] uses an initial procedure (artificial variable method of linear programming) to compute a starting solution. Similarly, the complex Remez initial stage

is used to compute a starting solution for the second stage. However, the advantage of the complex Remez algorithm is that, in many cases, it produces an initial solution which is optimal or very close to optimal. Another advantage is that the complex Remez method is more efficient in terms of computational complexity and speed than the *initial* procedure used in [22].

The timing results are shown in Table 3.2. It can be seen that the new Chebyshev approximation algorithm converges more rapidly than the other techniques while maintaining a high accuracy. It should be noted that the optimal error norm cannot be smaller than the error norm resulting from the complex Remez algorithm (initial stage). In Example 2 of the nearly-linear phase designs, the Burnside-Parks algorithm resulted in an error norm smaller than the one computed by the complex Remez algorithm only because of small differences in the used approximation grids. Table 3.2 shows that the proposed two-stage algorithm requires a smaller number of major iterations than the technique presented in [22]. Moreover, in contrast with [22], the number of executed major iterations is zero in many cases. It should be noted that the number of major iterations (if any) is independent of the method used to compute the optimal solutions on the constructed subsets B_s. Instead of the descent method, the simplex method could have been used as in [20, 21, 22]. If the simplex method is used for computing the best approximations on the subset B_s, the resulting algorithm would have exactly the same computational complexity per major iteration as [22] but would still converge in a smaller total number of major iterations. So, compared to the considered previous techniques [22, 19], it can be concluded that the new multiple-exchange two-stage approach results in a more efficient algorithm for optimal Chebyshev approximation.

CHAPTER 4

Multidimensional FIR Filter Design by Transformation

The multi-dimensional Chebyshev filter design problem is much more involved than the one-dimensional design problem. While an efficient characterization of Chebyshev solutions is still possible for a set of complex one-dimensional problems, none is available as one passes to higher dimensions [3, 40]. The exchange ascent-descent method of Chapter 3 can be used to compute the optimal, complex or real, multi-dimensional Chebyshev solutions. Linear programming based techniques [51, 21], exchange algorithms [40, 52], and reweighted least-squares methods [53, 54, 55] have been presented in the literature for solving the real multi-dimensional Chebyshev approximation problem. However, the above techniques are generally computationally intensive and put a limit on the size of the designed filter.

Since the direct solution of multi-dimensional Chebyshev approximation problems is computationally expensive, transformations have been used to efficiently design near-optimal multi-dimensional filters. Although suboptimal, the transformation methods are very fast since they convert the multi-dimensional filter design problem into a one-dimensional filter design problem. A very efficient and popular transformation method is due to McClellan [50, 3]. This technique transforms a 1-D prototype filter $H(\omega)$ into a 2-D filter $H(\omega_1, \omega_2)$ by a change of variables. The transformation is performed by using a transformation function $F(\omega_1, \omega_2)$ which would map the prototype filter to the desired 2-D filter. In contrast with other transformation techniques [56], the McClellan transformation method allows the designer to control the Chebyshev approximation error and to preserve the symmetry properties of the

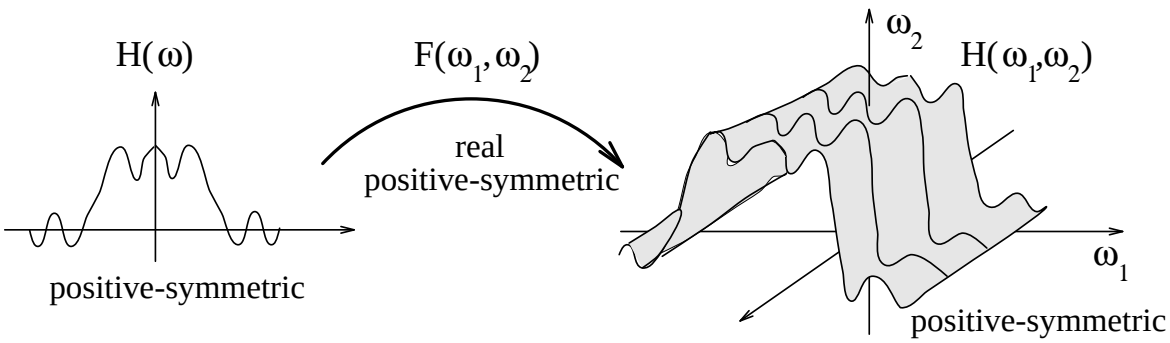


Figure 4.1. The original McClellan transformation method maps 1-D positive-symmetric filters $H(\omega)$ into 2-D positive-symmetric filters $H(\omega_1, \omega_2)$ using real and symmetric transformation functions $F(\omega_1, \omega_2)$.

desired 2-D filters. In practice, it results in very good filters that are close to optimal. In the original McClellan transformation method, the prototype filter $H(\omega)$, the transformation function $F(\omega_1, \omega_2)$, and the 2-D designed filter $H(\omega_1, \omega_2)$ are restricted to be real and positive-symmetric [50, 57]. Hale [48] relaxed these restrictions by introducing complex, but still symmetric, prototype filters.

This chapter studies the applicability and limitations of the McClellan transformation method and, as a result, extends this method so that new types of prototype filters can be transformed and new types of multi-dimensional filters can be designed. The main purpose of this study is to identify which prototype filters can be transformed (section 4.2), determine what types of positive and negative symmetries can be designed (section 4.3), and present procedures for transforming the new identified prototypes as well as rules for achieving the possible symmetries.

4.1 Brief Overview of the McClellan Transformation Method

As illustrated by Fig. 4.1, the McClellan transformation maps 1-D positive-

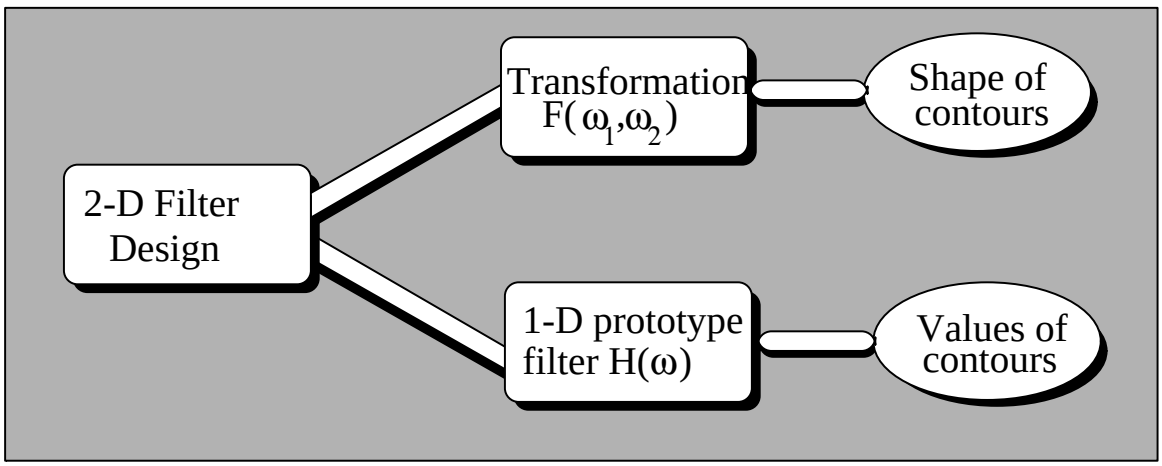


Figure 4.2. The transformation method breaks down the 2-D design problem into two smaller design problems.

symmetric filters $H(\omega)$ of the form

$$H(\omega) = h_o + 2\sum_{n=1}^{L} h_n \cos(n\omega) \tag{4.1}$$

into 2-D positive-symmetric filters $H(\omega_1, \omega_2)$ by means of the substitution

$$\cos\omega = F(\omega_1, \omega_2). \tag{4.2}$$

$F(\omega_1, \omega_2)$ is a real and symmetric transformation function of the form [58, 57]

$$F(\omega_1, \omega_2) = \sum_{q=0}^{Q}\sum_{r=0}^{R} t_{qr} \cos(q\omega_1)\cos(r\omega_2) + \sum_{i=0}^{I}\sum_{j=0}^{J} s_{ij} \sin(i\omega_1)\sin(j\omega_2). \tag{4.3}$$

The substitution (4.2) is possible due to the fact that the function $\cos(n\omega)$ is the nth Chebyshev polynomial $T_n[\cos\omega]$, which is a polynomial of degree n in the variable $\cos\omega$. As a result of this substitution, the following 2-D filter is obtained

$$H(\omega_1, \omega_2) = h_o + 2\sum_{n=1}^{L} h_n T_n[F(\omega_1, \omega_2)]. \tag{4.4}$$

Note that $F(\omega_1, \omega_2)$ is itself the frequency response of a 2-D FIR filter, which ensures that the resulting filter $H(\omega_1, \omega_2)$ is a 2-D FIR filter. Furthermore, in order to have

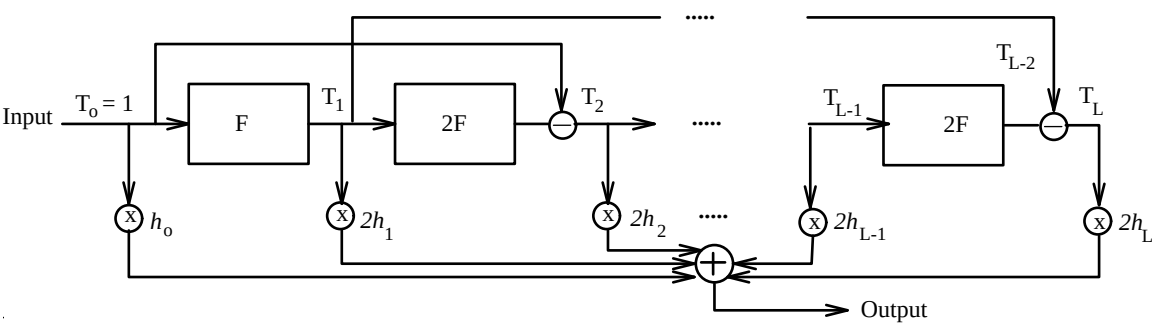


Figure 4.3. Realization of the 2-D filter by means of a Chebyshev structure.

a well-defined mapping, $F(\omega_1, \omega_2)$ is required to take the same values as the *cosine* function it replaces, i.e. $|F(\omega_1, \omega_2)| \leq 1$.

Figure 4.2 illustrates the fact that the transformation method breaks down the 2-D design problem into two decoupled smaller problems: the design of the transformation function $F(\omega_1, \omega_2)$ and the design of the 1-D prototype filter $H(\omega)$. $F(\omega_1, \omega_2)$ defines the shape of the contours of the resulting transformed 2-D filter, while $H(\omega)$ determines the values of the frequency response $H(\omega_1, \omega_2)$ along these contours. By virtue of the recurrence formula for the Chebyshev polynomials [32], the produced filters can be realized efficiently using the Chebyshev structure shown in Fig. 4.3 [59].

The primary advantage of this implementation is its modularity and its flexibility. To change the prototype filter without changing the shape of the 2-D contours, one simply changes the weighting coefficients $\{h_n\}$. To change the shape of the contours, one needs only to change the coefficients inside each of the subfilters F. As a result of this implementation, a variety of filters can be efficiently realized. More details about the McClellan transformation can be found in [50, 4, 3, 60].

4.2 Identification of Transformable Prototype Filters

This section identifies the class of prototype filters that can be mapped into multi-dimensional filters by means of a transformation function. The discussion emphasizes the use of "well-behaved" transformation functions, i.e., functions that do not distort but exactly map the values of the prototype frequency response to corresponding contours in the multi-dimensional frequency domain. Well-behaved mappings are important because they allow the designer to control the values of the designed multi-dimensional filter and, therefore, control the Chebyshev error norm.

4.2.1 Transforming an Arbitrary Prototype Filter

A general complex-valued one-dimensional FIR filter of length N, with the coefficients $\{h_k;\ k = K_1, \ldots, K_1 + N - 1\}$, has a frequency response $H(\omega)$ given by

$$H(\omega) = \sum_{k=K_1}^{K_1+N-1} h_k e^{-jk\omega} = \sum_{k=K_1}^{K_1+N-1} h_k (e^{-j\omega})^k \tag{4.5}$$

One way to transform the prototype (4.5), which has no particular symmetry, into a 2-D filter is to replace $e^{-j\omega}$ by a suitable transformation function $F(\omega_1, \omega_2)$

$$e^{-j\omega} \longleftrightarrow F(\omega_1, \omega_2). \tag{4.6}$$

A well-behaved mapping $F(\omega_1, \omega_2)$ must be the frequency response of a 2-D FIR digital filter and must take the same values of the function $e^{-j\omega}$, i.e.,

$$F(\omega_1, \omega_2) = \sum_{q=Q_1}^{Q_1+Q-1} \sum_{r=R_1}^{R_1+R-1} f(q, r) e^{-j(\omega_1 q + \omega_2 r)} \tag{4.7}$$

and

$$|F(\omega_1, \omega_2)| = 1. \tag{4.8}$$

Appendix B shows that the following theorem holds.

Theorem 4.1 Any function $F(\omega_1, \omega_2)$ satisfying (4.7) and (4.8) is a complex-valued phase function of the form

$$F(\omega_1, \omega_2) = e^{-j(n_1\omega_1 + n_2\omega_2 + C)} \tag{4.9}$$

where n_1 and n_2 are integer constants and C is any real-valued constant. □

Theorem 4.1 states that the impulse response of an allpass FIR filter has exactly one non-zero sample.

Alternatively, it is shown below that an arbitrary 1-D filter can be expressed in terms of Chebyshev and other related polynomials. Rewriting $e^{-j\omega k}$ as $\cos(k\omega) - j\sin(k\omega)$, (4.5) becomes

$$H(\omega) = \sum_{k=K_1}^{K_1+N-1} (c_k \cos(k\omega) + s_k \sin(k\omega)) \tag{4.10}$$

where $c_k = h_k$ and $s_k = -jh_k$. The functions $\cos(k\omega)$ and $\sin(k\omega)$ can be expressed as

$$\cos(k\omega) = T_k[\cos\omega] \tag{4.11}$$

$$\sin(k\omega) = \begin{cases} S_k[\sin\omega], & k \text{ odd} \\ S_k[\sin\omega]\cos\omega, & k \text{ even} \end{cases} \tag{4.12}$$

where $T_k[x]$ and $S_k[x]$ satisfy the following recurrence formulae:

$$\begin{cases} T_k[x] = 2xT_{k-1}[x] - T_{k-2}[x] \\ T_0[x] = 1 \\ T_1[x] = x \end{cases} \tag{4.13}$$

and (Appendix B)

$$\begin{cases} S_k[x] = \begin{cases} 2S_{k-1}[x](1-x^2) - S_{k-2}[x], & k \text{ odd} \\ 2S_{k-1}[x] - S_{k-2}[x], & k \text{ even} \end{cases} \\ S_0[x] = 0 \\ S_1[x] = x \end{cases} \tag{4.14}$$

Note that $T_k[x]$ is the kth Chebyshev polynomial of degree k in x, while $S_k[x]$ is a related polynomial whose degree is k for k odd, and $k-1$ for k even. $S_k[x]$ is shown (Appendix B) to be related to $T_k[x]$ as follows:

$$S_k[x] = \begin{cases} (-1)^m T_{2m+1}[x], & k = 2m+1 \ (k \text{ odd}) \\ 2(-1)^{m-1} \sum_{p=1}^{m} T_{2p-1}[x], & k = 2m \ (k \text{ even}) \text{ and } k \neq 0 \end{cases} \tag{4.15}$$

The following relations also hold (Appendix B)

$$T_{2m}[\cos\omega] = (-1)^m T_{2m}[\sin\omega] \tag{4.16}$$

$$S_{2m}[\sin\omega]\cos\omega = (-1)^{m+1} S_{2m}[\cos\omega]\sin\omega. \tag{4.17}$$

Now, using (4.11) and (4.12), $H(\omega)$ can be rewritten as

$$\begin{aligned} H(\omega) = & \sum_{k=K_1}^{K_1+N-1} c_k T_{|k|}[\cos\omega] + \sum_{\substack{k=K_1 \\ k \text{ odd}}}^{K_1+N-1} (\frac{k}{|k|}) s_k S_{|k|}[\sin\omega] + \\ & \sum_{\substack{k=K_1 \\ k \text{ even}}}^{K_1+N-1} (\frac{k}{|k|}) s_k S_{|k|}[\sin\omega]\cos\omega. \end{aligned} \tag{4.18}$$

where $(\frac{k}{|k|})$ is the sign of the index k.

Equation (4.18) indicates that two real transformation functions are needed to transform an arbitrary filter by means of the two substitutions

$$\begin{cases} \cos\omega & \longleftrightarrow & F_1(\omega_1, \omega_2) \\ \sin\omega & \longleftrightarrow & F_2(\omega_1, \omega_2) \end{cases} \tag{4.19}$$

As before, the mapping functions $F_1(\omega_1, \omega_2)$ and $F_2(\omega_1, \omega_2)$ must be the frequency responses of 2-D FIR filters with $-1 \leq F_1(\omega_1, \omega_2) \leq 1$ and $-1 \leq F_2(\omega_1, \omega_2) \leq 1$. Furthermore, in order to preserve the characteristics of the one-dimensional prototype $H(\omega)$, $F_1(\omega_1, \omega_2)$ and $F_2(\omega_1, \omega_2)$ must satisfy the following additional constraint

$$F_1^2(\omega_1, \omega_2) + F_2^2(\omega_1, \omega_2) = 1. \tag{4.20}$$

Theorem 4.2 If $F_1(\omega_1, \omega_2)$ and $F_2(\omega_1, \omega_2)$ are the frequency responses of 2-D FIR filters and satisfy (4.20), then

$$\begin{cases} F_1(\omega_1, \omega_2) &= \cos(n_1\omega_1 + n_2\omega_2 + C) \\ F_2(\omega_1, \omega_2) &= \sin(n_1\omega_1 + n_2\omega_2 + C) \end{cases} \tag{4.21}$$

where n_1 and n_2 are integer constants and C is a real-valued constant. □

Proof: Let $F(\omega_1, \omega_2) = F_1(\omega_1, \omega_2) + j\, F_2(\omega_1, \omega_2)$. $F(\omega_1, \omega_2)$ corresponds to the frequency response of a 2-D FIR filter and $|F(\omega_1, \omega_2)| = 1$. Then, by Theorem 4.1,

$$F(\omega_1, \omega_2) = e^{j(n_1\omega_1 + n_2\omega_2 + C)} \tag{4.22}$$

which leads to (4.21). □

Although arbitrary prototype filters can be transformed, the constraints (4.9) or (4.21) limit the choice of admissible mappings and, therefore, the filters that can be designed to those with colinear isopotentials. A broader selection of transformation mappings with arbitrary contours could be obtained if the constraints (4.9) and (4.21) are not satisfied exactly but are satisfied in an approximative sense. One possible method is sketched below but will not be treated further in this work.

Note that the substitutions (4.6) and (4.19) correspond to a transformation of variables which maps the 1-D frequency ω to a contour in the 2-D frequency space (ω_1, ω_2). Let the desired contour mapping be

$$\omega = \mathcal{C}_d(\omega_1, \omega_2) \tag{4.23}$$

where $\mathcal{C}_d(\omega_1, \omega_2)$ is a function representing the desired contour in the 2-D space. Then, the mapping functions $F_1(\omega_1, \omega_2)$ and $F_2(\omega_1, \omega_2)$ of (4.19) could be constructed such that they best approximate $\cos(\mathcal{C}_d(\omega_1, \omega_2))$ and $\sin(\mathcal{C}_d(\omega_1, \omega_2))$, respectively, relative to some error norm. A similar technique was proposed in [61] where two real-valued transformation functions were used to map a zero-phase 1-D filter into zero-phase

multi-dimensional filters. It should be noted that $F_1(\omega_1, \omega_2)$ and $F_2(\omega_1, \omega_2)$ are real-valued FIR frequency responses and, thus, they may be expressed in the general form

$$\begin{aligned} F_{\{1,2\}}(\omega_1, \omega_2) &= \sum_{q=0}^{Q}\sum_{r=0}^{R} c_{qr} \cos(q\omega_1) cos(r\omega_2) + \\ &\quad \sum_{i=0}^{I}\sum_{j=0}^{J} s_{ij} \sin(i\omega_1) \sin(j\omega_2) + \\ &\quad \sum_{m=0}^{M}\sum_{k=0}^{K} cs_{mk} \cos(m\omega_1) \sin(k\omega_2) + \\ &\quad \sum_{u=0}^{U}\sum_{v=0}^{V} sc_{uv} \sin(u\omega_1) \cos(v\omega_2). \end{aligned} \tag{4.24}$$

The mapping design problem can now be formulated as follows. Among all the possible coefficients $\{\{c_{qr}\}, \{s_{ij}\}, \{cs_{mk}\}, \{sc_{uv}\}\}$, find the ones that minimize the weighted errors

$$E_1(\omega_1, \omega_2) = W_1(\omega_1, \omega_2)[\cos(\mathcal{C}_d(\omega_1, \omega_2)) - F_1(\omega_1, \omega_2)] \tag{4.25}$$

$$E_2(\omega_1, \omega_2) = W_2(\omega_1, \omega_2)[\sin(\mathcal{C}_d(\omega_1, \omega_2)) - F_2(\omega_1, \omega_2)] \tag{4.26}$$

in the least-square or minimax (Chebyshev) sense over the frequency region of interest. $W_1(\omega_1, \omega_2)$ in (4.25) and $W_2(\omega_1, \omega_2)$ in (4.26) are real and positive weighting functions. In addition, a well-behaved mapping requires that

$$-1 \leq F_{\{1,2\}}(\omega_1, \omega_2) \leq 1. \tag{4.27}$$

The condition (4.27) is necessary but not sufficient in this case since the values of the prototype frequency response may be altered by the mapping process. This is due to the fact that the constraint (4.20) is not exactly satisfied. The constraint (4.27) could be either incorporated into the approximation problem or satisfied by performing a simple scaling operation [57]. The scaling operation does not affect the shape of the resulting contours; it only changes the 1-D angular frequency which is mapped to each contour.

The scaling operation can be performed as follows [57]. Let F_{mx} and F_{mn} denote, respectively, the maximum and minimum of a transformation function $F(\omega_1, \omega_2)$, and define

$$s_1 = \frac{2}{F_{\mathrm{mx}} - F_{\mathrm{mn}}} \tag{4.28}$$

$$s_2 = 1 - s_1 F_{\mathrm{mx}}. \tag{4.29}$$

Then, the scaling operation can be simply performed by constructing a new mapping $F_{\mathrm{new}}(\omega_1, \omega_2)$ as follows [57]:

$$F_{\mathrm{new}}(\omega_1, \omega_2) = s_1 F(\omega_1, \omega_2) + s_2. \tag{4.30}$$

It is easy to verify that $|F_{\mathrm{new}}(\omega_1, \omega_2)| \leq 1$.

4.2.2 Transforming Symmetric Complex Prototype Filters

This section shows that transformation functions, which can approximate arbitrary contours with an exact map of the frequency response values of the prototype filter, can be obtained when the prototype filter satisfies some symmetry conditions. These well-behaved transformation functions are real-valued resulting in multi-dimensional filters whose rectangular regions of support have odd length sides. The required symmetry conditions on the prototype filter are derived in this section for both odd- and even-length prototype filters using expression (4.18) for the prototype frequency response. The general expression (4.18) is useful since it leads to a clear statement of the conditions required on the transformation function to obtain a well-behaved mapping for any type of prototype filter $H(\omega)$.

Odd-Length Prototype Filters

From the discussion in section 4.2.1, it can be concluded that a well-behaved mapping, approximating arbitrary contours, can only be used in the cases where (4.18) reduces to an expression in terms of only one of the trigonometric functions $\cos\omega$ and $\sin\omega$.

The resulting prototype filter can then be mapped into a multi-dimensional filter by replacing the trigonometric function ($\cos\omega$ or $\sin\omega$) by any real transformation function F satisfying $|F| \leq 1$. The reduction of (4.18) to the desired expression can be done by considering all possible combinations of the three summation terms in (4.18). However, this latter requirement places a constraint on the type of the prototype filter. With $\{h_n\}$ denoting the coefficients of an odd-length prototype filter $H(\omega)$, three main cases can be distinguished from (4.18):

1. *Elimination of the summation terms depending on* $\sin\omega$ (i.e., second and third term in (4.18))

   This case corresponds to a positive symmetric prototype ($h_n = h_{-n}$). The second and third summation terms of (4.18) cancel out and (4.18) reduces to

   $$H(\omega) = h_o + 2\sum_{k=1}^{(N-1)/2} h_k T_k[\cos\omega] \tag{4.31}$$

2. *Elimination of the summation terms depending on* $\cos\omega$ (i.e., first and third term in (4.18))

   This case corresponds to a negative symmetric prototype ($h_n = -h_{-n}$) with the additional constraint $h_n = 0$ for n even. In this case, the frequency response $H(\omega)$ is negative symmetric with respect to $\omega = 0$ and positive symmetric with respect to the point $\omega = \pi/2$ ($H(\omega) = H(\pi-\omega)$). The first and third summation terms cancel out and (4.18) becomes

   $$H(\omega) = -2j\sum_{\substack{k=1 \\ k \text{ odd}}}^{(N-1)/2} h_k S_k[\sin\omega]. \tag{4.32}$$

   From (4.15), (4.32) could be rewritten as

   $$H(\omega) = -2j\sum_{m=0}^{\lfloor (N-3)/4 \rfloor} h_{2m+1}(-1)^m T_{2m+1}[\sin\omega] \tag{4.33}$$

   where the symbol $\lfloor . \rfloor$ denotes the operation of rounding to the smallest integer.

3. *Reducible combinations of distinct terms*

The three distinct summation terms in (4.18) can be combined as follows.

(a) Term 1 + Term 2

(b) Term 1 + Term 3

(c) Term 2 + Term 3

(d) Term 1 + Term 2 + Term 3

Out of the four possible combinations listed above, only the first one (Term 1 + Term 2) is reducible to an expression in terms of only one of the trigonometric functions $\{\cos\omega, \sin\omega\}$. The reason is that no mathematical relation is yet known for reducing the third term in (4.18) to a form with only one of the considered trigonometric functions. Using the relations (4.15) and (4.16) derived in section 4.2.1, it is shown below that the first combination can be reduced to the desired form if the impulse response of the prototype filter satisfies

$$h_n = (-1)^n h_{-n}. \tag{4.34}$$

The relation (4.34) implies that the prototype frequency response $H(\omega)$ is symmetric with respect to $\omega = \pi/2$ and that case 2, considered above, corresponds to a special case of (4.34). From (4.34), (4.18) reduces to

$$H(\omega) = h_o + 2\sum_{m=1}^{\lfloor (N-1)/4 \rfloor} h_{2m} T_{2m}[\cos\omega] - 2j \sum_{m=0}^{\lfloor (N-3)/4 \rfloor} h_{2m+1} S_{2m+1}[\sin\omega]. \tag{4.35}$$

where the symbol $\lfloor . \rfloor$ denotes the operation of rounding to the smallest integer. Now, using (4.15) and (4.16), (4.35) can be rewritten as

$$\begin{aligned} H(w) &= h_o + 2\sum_{m=1}^{\lfloor (N-1)/4 \rfloor} h_{2m}(-1)^m T_{2m}[\sin\omega] - \\ &\quad 2j \sum_{m=0}^{\lfloor (N-3)/4 \rfloor} h_{2m+1}(-1)^m T_{2m+1}[\sin\omega] \\ &= h_o + 2\sum_{n=0}^{(N-1)/2} (-1)^n j^n h_n T_n[\sin\omega]. \end{aligned} \tag{4.36}$$

Even-Length Prototypes

Transformation functions approximating arbitrary contours can also be used to map even-length prototype filters to 2-D filters whose region of support consist of $(2I+1)\times(2J+1)$ points, where I and J are positive integers. A method was presented in [62] where even-length, negative-symmetric, linear-phase FIR prototype filters (type 4 [1]) were transformed into 2-D filters with real or purely imaginary frequency responses. In this section, a new design procedure is presented for designing $(2I+1)\times(2J+1)$ 2-D filters by transforming complex-valued, positive- or negative-symmetric, even-length prototype filters.

First, it is easy to show that the unified formulation presented in [3] for 1-D linear-phase filters applies also to complex-valued, positive- or negative-symmetric, 1-D filters. It follows that a complex symmetric prototype filter of even length $2L$ can be expressed as

$$H(\omega) = e^{-j\omega/2}Q(\omega)P(\omega) \tag{4.37}$$

where

$$P(\omega) = \sum_{n=0}^{L-1} c_n T_n[\cos\omega] \tag{4.38}$$

$$Q(\omega) = \begin{cases} \cos(\omega/2), & \text{if positive symmetry} \\ \sin(\omega/2), & \text{if negative symmetry} \end{cases} \tag{4.39}$$

Note that $P(\omega)$ is an odd-length FIR filter and that the coefficients $\{c_n\}$ are in general complex-valued. It is easy to see that (4.37) can be rewritten as

$$H(\omega) = e^{-j\omega/2}Q(\omega)\sum_{n=0}^{L-1} c_n T_n[s(2(Q(\omega))^2-1)] \tag{4.40}$$

where $Q(\omega)$ is given by (4.39) and

$$s = \begin{cases} 1, & \text{if positive symmetry} \\ -1, & \text{if negative symmetry} \end{cases} \tag{4.41}$$

Now, define

$$\hat{H}(\omega) = e^{j\omega/2}H(\omega) = Q(\omega)\sum_{n=0}^{L-1} c_n T_n[s(2(Q(\omega))^2-1)] \tag{4.42}$$

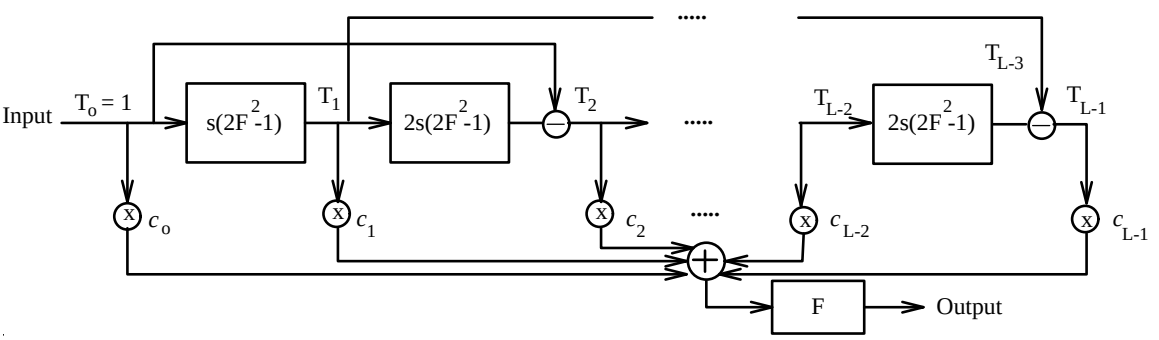


Figure 4.4. Realization of the odd-length 2-D filters which are designed by transforming even-length 1-D filters. The coefficients $\{c_n\}$ are obtained from the coefficients $\{h_n\}$ of the prototype 1-D filter as described in Step 2 of the design procedure.

The function $\hat{H}(\omega)$ can be transformed into a 2-D filter $H(\omega_1, \omega_2)$ by substituting for $Q(\omega)$ in (4.42) a valid transformation function $F(\omega_1, \omega_2)$. In order to preserve the values of the function $\hat{H}(\omega)$, $F(\omega_1, \omega_2)$ must take the same values as $Q(\omega)$. So, $F(\omega_1, \omega_2)$ must be the frequency response of a 2-D FIR filter with $-1 \leq F(\omega_1, \omega_2) \leq 1$. The new design procedure for generating a 2-D filter of (odd) size $(2I+1) \times (2J+1)$ from an even-length prototype (4.37) is outlined below.

1. Determine and design an appropriate prototype filter (4.37) with impulse response $h(n)$ of length $2L$.

2. Using $h(n)$, determine the $2L-1$ point filter $p(n)$, where $p(n)$ is the inverse Discrete-Time Fourier Transform (DTFT) of $P(\omega)$ given by (4.38).

3. Using the coefficients $f(n_1, n_2)$ of the selected transformation function $F(\omega_1, \omega_2)$, compute $f_2(n_1, n_2)$, where $f_2(n_1, n_2)$ is the inverse DTFT of

$$F_2(\omega_1, \omega_2) = s(2(F(\omega_1, \omega_2))^2 - 1). \tag{4.43}$$

4. Using $F_2(\omega_1, \omega_2)$, transform $P(\omega)$ into the 2-D filter $P(\omega_1, \omega_2)$ with impulse response $p(n_1, n_2)$.

5. Convolve $p(n_1, n_2)$ with $f(n_1, n_2)$ to get $h(n_1, n_2)$, where $h(n_1, n_2)$ is the desired impulse response of the transformed two-dimensional filter.

The designed filters can be efficiently realized using the Chebyshev structure shown in Fig. 4.4. Two-dimensional odd-length complex filters having different symmetries can be designed with this method. These symmetry properties are discussed in section 4.3.

4.3 Design of Complex Positive- and Negative-symmetric 2-D Filters: Feasibility and Symmetry Conditions

This section considers all possible centro and quadrant symmetries that can be achieved using the transformation method. For each of the possible symmetries, conditions are derived on the prototype filter and the transformation function for designing a complex 2-D filter having the desired symmetry properties. Only the complex prototype filters of section 4.2.2, which can be mapped using a single real-valued transformation function, are considered in this analysis. The frequency response $H(\omega)$ of these filters, with an odd length N, can be expressed as

$$H(\omega) = \sum_{n=0}^{(N-1)/2} c_n T_n[x(\omega)] \tag{4.44}$$

where $x(\omega) = \cos\omega$ or $x(\omega) = \sin\omega$, and N is the filter length. Furthermore, note that

$$T_n[x] = \begin{cases} T_n[-x], & n \text{ even} \\ -T_n[-x], & n \text{ odd} \end{cases} \tag{4.45}$$

For the even-length filters, $\hat{H}(\omega)$ given by (4.42) is used as the prototype to be mapped. The transformed 2-D filter is obtained by replacing $x(\omega)$ in (4.44) or $Q(\omega)$ in (4.42) by a transformation function $F(\omega_1, \omega_2)$. Therefore, the well-behaved mappings $F(\omega_1, \omega_2)$ are real-valued with $-1 \leq F(\omega_1, \omega_2) \leq 1$. It follows that $f(n_1, n_2)$, the

impulse response of $F(\omega_1, \omega_2)$, has a rectangular region of support with odd-length sides $(2K+1) \times (2R+1)$. It also follows that the transformed 2-D filter would have a region of support with odd-length sides $(2I+1) \times (2J+1)$. Complex 2-D filters whose region of support has even-length sides $(2I \times 2J)$ can be designed in some special cases and are discussed in section 4.3.2.

For convenience, the expression "odd-length" for 2-D filters will be used to denote 2-D filters having a rectangular region of support with odd-length sides; similarly, an "even-length" 2-D filter is used to denote a 2-D filter whose rectangular region of support has even-length sides.

4.3.1 Design of Odd-Length 2-D Filters

For 2-D complex FIR filters whose impulse response $h(n_1, n_2)$ is nonzero only over a rectangle with sides consisting of $(2I+1) \times (2J+1)$ points, only the following four types of centro and quadrant symmetries are possible in general:

1. *Positive centrosymmetry:* $h(n_1, n_2) = h(-n_1, -n_2)$

   - For any prototype filter of the form (4.44) or (4.42), a sufficient condition for obtaining a positive centro-symmetric 2-D filter $H(\omega_1, \omega_2)$ is to have a positive centro-symmetric mapping

   $$F(\omega_1, \omega_2) = F(-\omega_1, -\omega_2). \tag{4.46}$$

   - Positive centro-symmetric 2-D filters can also be obtained by using mappings with negative symmetries

   $$F(\omega_1, \omega_2) = -F(-\omega_1, -\omega_2) \tag{4.47}$$

   or

   $$\left\{ \begin{array}{rcl} F(\omega_1, \omega_2) & = & \pm F(-\omega_1, -\omega_2) \\ F(-\omega_1, \omega_2) & = & \mp F(\omega_1, -\omega_2) \end{array} \right\}, \quad 0 \leq \omega_1, \omega_2 \leq \pi \tag{4.48}$$

However, in these cases, the prototype filter has to have an odd length and be positive symmetric with respect to both $\omega = 0$ and $\omega = \pi/2$. This latter requirement eliminates the odd-degree Chebyshev polynomials in (4.44).

2. *Negative centrosymmetry:* $h(n_1, n_2) = -h(-n_1, -n_2)$

   From case 1, it can be deduced that $F(\omega_1, \omega_2)$ cannot be positive centro-symmetric. For any prototype filter of the form (4.44) or (4.42), a negative centro-symmetric complex 2-D filter is obtained by using a negative centro-symmetric mapping

   $$F(\omega_1, \omega_2) = -F(-\omega_1, -\omega_2). \tag{4.49}$$

   In addition, if an odd-length prototype filter is used, it needs to be positive symmetric with respect to $\omega = \pi/2$ ($\omega = 0$) and negative symmetric with respect to $\omega = 0$ ($\omega = \pi/2$). This latter requirement eliminates the even-degree Chebyshev polynomials in (4.44). No restrictions are placed on the even-length prototype filters.

3. *Positive quadrant symmetry:* $h(n_1, n_2) = h(-n_1, -n_2) = h(-n_1, n_2) = h(n_1, -n_2)$

   This case corresponds to a positive centro-symmetric filter with an additional quadrant symmetry constraint.

   - For any prototype filter of the form (4.44) or (4.42), A sufficient condition for designing a 2-D filter which is positive quadrant symmetric is to have a positive quadrant symmetric transformation function

     $$F(\omega_1, \omega_2) = F(-\omega_1, -\omega_2) = F(-\omega_1, \omega_2) = F(\omega_1, -\omega_2). \tag{4.50}$$

   - With an odd-length filter (4.44) having $c_{2m+1} = 0$, other types of symmetries are possible for $F(\omega_1, \omega_2)$, i.e.,

     $$F(\omega_1, \omega_2) = s_1 F(-\omega_1, \omega_2) = s_2 F(-\omega_1, -\omega_2) = s_3 F(-\omega_1, \omega_2) \tag{4.51}$$

     where s_1, s_2, and s_3 are either -1 or 1 and are not necessarily equal.

4. *Negative quadrant symmetries*

   In this case, the following symmetry properties are considered for the 2-D filter $H(\omega_1, \omega_2)$:

$$H(\omega_1, \omega_2) = s_1 H(-\omega_1, \omega_2) = s_2 H(-\omega_1, -\omega_2) = s_3 H(-\omega_1, \omega_2) \tag{4.52}$$

   where s_1, s_2, and s_3 are either -1 or 1 with exactly two of them equal to -1. Such 2-D symmetries are obtained by using a transformation function $F(\omega_1, \omega_2)$ with the same type of symmetry as the desired 2-D filter, i.e.,

$$F(\omega_1, \omega_2) = s_1 F(-\omega_1, \omega_2) = s_2 F(-\omega_1, -\omega_2) = s_3 F(-\omega_1, \omega_2) \tag{4.53}$$

   where s_1, s_2, and s_3 are as in (4.52). In addition, if an odd-length prototype (4.44) is used, it needs to have zero even-indexed coefficients, i.e., $\{c_{2m}\} = 0$. No restrictions are placed on the even-length prototype filters. Other types of negative quadrant symmetries, where only one of $\{s_i\}_{i=1}^3$ is -1, do not seem possible in general for FIR filters.

4.3.2 Design of Even-Length 2-D Filters

Even-length complex prototype filters can be transformed, in some special cases, to produce complex 2-D filters with a rectangular region of support having even-length $(2I \times 2J)$ sides. A design procedure was presented in [57] for designing real and quadrant symmetric $(2I \times 2J)$ filters. This design procedure is extended here to allow the design of complex-valued filters with positive and negative symmetries.

Although even-length prototype filters cannot directly satisfy positive or negative symmetry conditions due to a half-sample phase shift term, it was shown earlier that they can be rewritten as

$$H(\omega) = e^{-j\omega/2} \hat{H}(\omega) \tag{4.54}$$

where $\hat{H}(\omega)$ can be a positive- or negative-symmetric function of ω and is given by (4.42). Using even-length prototype filters of the form (4.54), a design procedure

is presented for designing complex $(2I \times 2J)$ FIR digital filters with a rectangular region of support $\{-I+1 \leq n_1 \leq I, -J+1 \leq n_2 \leq J\}$ and having the following symmetry properties:

$$\hat{H}(\omega_1, \omega_2) = (-1)^{i_1}\hat{H}(-\omega_1, \omega_2) = (-1)^{i_2}\hat{H}(-\omega_1, -\omega_2) = (-1)^{i_3}\hat{H}(-\omega_1, \omega_2) \quad (4.55)$$

where

$$\hat{H}(\omega_1, \omega_2) = e^{j(\omega_1+\omega_2)/2}H(\omega_1, \omega_2) \quad (4.56)$$

and

$$i_1 \otimes i_2 \otimes i_3 = 0, \qquad i_1, i_2, i_3 \in \{0, 1\}. \quad (4.57)$$

The symbol $\otimes$ in (4.57) denotes the logical XOR operation.

The frequency response $H(\omega_1, \omega_2)$ satisfying (4.55) can be expressed as

$$H(\omega_1, \omega_2) = e^{-j(\omega_1+\omega_2)/2}Q(\omega_1, \omega_2)P(\omega_1, \omega_2) \quad (4.58)$$

where

$$P(\omega_1, \omega_2) = \sum_{n_1=0}^{I-1}\sum_{n_2=0}^{J-1} p(n_1, n_2)\cos n_1\omega_1 \cos n_2\omega_2 \quad (4.59)$$

$$Q(\omega_1, \omega_2) = \begin{cases} \cos(\omega_1/2)\cos(\omega_2/2), & \text{if } i_1 = i_2 = i_3 = 0 \\ \cos(\omega_1/2)\sin(\omega_2/2), & \text{if } i_1 = 0;\ i_2 = i_3 = 1 \\ \sin(\omega_1/2)\sin(\omega_2/2), & \text{if } i_1 = i_3 = 1;\ i_2 = 0 \\ \sin(\omega_1/2)\cos(\omega_2/2), & \text{if } i_1 = i_2 = 1;\ i_3 = 0 \end{cases} \quad (4.60)$$

Note that $P(\omega_1, \omega_2)$ is a 2-D quadrant symmetric filter of size $(2I-1) \times (2J-1)$. It is therefore possible to construct filters $P(\omega_1, \omega_2)$ from $P(\omega)$ in (4.38) using a McClellan transformation for odd-length filters as discussed earlier in this section (see case 3 above).

The design procedure for generating a 2-D filter (4.58) from a 1-D prototype filter (4.54) can now be stated.

1. Determine and design an appropriate prototype filter (4.54) with impulse response $h(n)$ of length $2L$.

2. Using $h(n)$, determine the $2L-1$ point filter $p(n)$, where $p(n)$ is the inverse Discrete-Time Fourier Transform of $P(\omega)$ given by (4.38).

3. Using a suitable transformation function $F(\omega_1,\omega_2)$, transform $P(\omega)$ to the 2-D quadrant symmetric filter $P(\omega_1,\omega_2)$, given by (4.59), with impulse response $p(n_1,n_2)$.

4. Convolve $p(n_1,n_2)$ with $\hat{q}(n_1,n_2)$ to get $h(n_1,n_2)$, where $\hat{q}(n_1,n_2)$ is the impulse response of

$$\hat{Q}(\omega_1,\omega_2) = e^{-j(\omega_1+\omega_2)/2}Q(\omega_1,\omega_2). \tag{4.61}$$

Any transformation function $F(\omega_1,\omega_2)$, which maps $P(\omega)$ to a 2-D quadrant positive-symmetric filter $P(\omega_1,\omega_2)$ could be used. However, the contours of the mapping $F(\omega_1,\omega_2)$ in Step 3 would be altered by the convolution of Step 4. The contour shape is preserved only if $F(\omega_1,\omega_2)$ is chosen such that its contours correspond to the contours

$$|Q(\omega_1,\omega_2)|^2 = \text{constant}. \tag{4.62}$$

The proper prototype filter and mapping function $F(\omega_1,\omega_2)$ need thus be determined for each possible symmetry of $H(\omega_1,\omega_2)$ and choice of $Q(\omega_1,\omega_2)$. From (4.60), there are four possible symmetries corresponding to the following four choices of $Q(\omega_1,\omega_2)$:

1. $Q(\omega_1,\omega_2) = \cos(\omega_1/2)\cos(\omega_2/2)$

   In this case,

$$\hat{H}(\omega_1,\omega_2) = \hat{H}(-\omega_1,\omega_2) = \hat{H}(-\omega_1,-\omega_2) = \hat{H}(\omega_1,-\omega_2). \tag{4.63}$$

   In addition, note that the 2-D designed filter cannot be highpass since

$$H(\pi,\omega_2) = H(\omega_1,\pi) = 0. \tag{4.64}$$

   The requirement (4.62) leads to

$$F(\omega_1,\omega_2) = 0.5(\cos\omega_1 + \cos\omega_2 + \cos\omega_1\cos\omega_2 - 1) \tag{4.65}$$

which is McClellan's original transformation for nearly circular contours. Scaled and shifted versions of (4.65) can also be used as long as they correspond to a well-behaved mapping with $-1 \leq F(\omega_1, \omega_2) \leq 1$. The mapping $F(\omega_1, \omega_2)$ selected as in (4.65) satisfies

$$F(0, \omega) = F(\omega, 0) = \cos \omega. \tag{4.66}$$

It follows that

$$H(\omega, 0) = H(0, \omega) = e^{-j\omega/2} \cos(\omega/2) P(\omega) = H_s(\omega) \tag{4.67}$$

where $H_s(\omega)$ is an even-length, positive symmetric filter. In order to preserve the characteristics of the prototype filter, the 1-D prototype filter (4.54) should be an even-length positive symmetric filter, i.e.,

$$H(\omega) = H_s(\omega) = e^{-j\omega/2} \cos(\omega/2) P(\omega). \tag{4.68}$$

Using a similar reasoning, it is easy to see that, if the mapping $F_2(\omega_1, \omega_2) = -F(\omega_1, \omega_2)$ is selected, a negative symmetric even-length prototype filter would be the appropriate filter to use. So, the most appropriate prototype filter depends on which scaled and shifted version of (4.65) is used. Figure 4.5(a) shows the contours of (4.65).

2. $Q(\omega_1, \omega_2) = \cos(\omega_1/2) \sin(\omega_2/2)$

   This case corresponds to

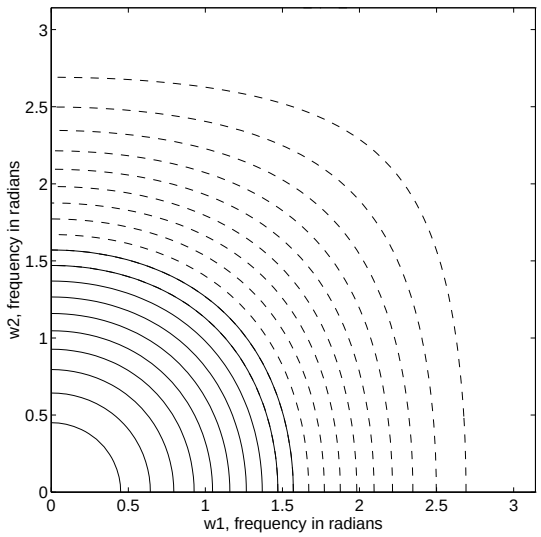
$$\hat{H}(\omega_1, \omega_2) = \hat{H}(-\omega_1, \omega_2) = -\hat{H}(-\omega_1, -\omega_2) = -\hat{H}(\omega_1, -\omega_2) \tag{4.69}$$

and

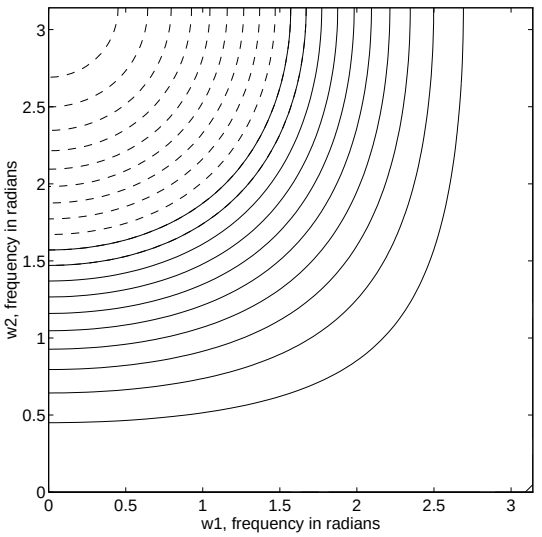
$$H(\pi, \omega_2) = H(\omega_1, 0) = 0. \tag{4.70}$$

Shifted and scaled versions of the following transformation $F(\omega_1, \omega_2)$ should be used:

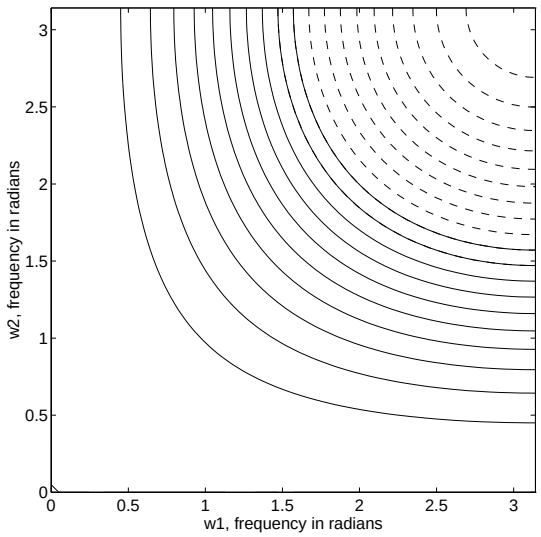
$$F(\omega_1, \omega_2) = 0.5(1 - \cos \omega_1 + \cos \omega_2 + \cos \omega_1 \cos \omega_2). \tag{4.71}$$



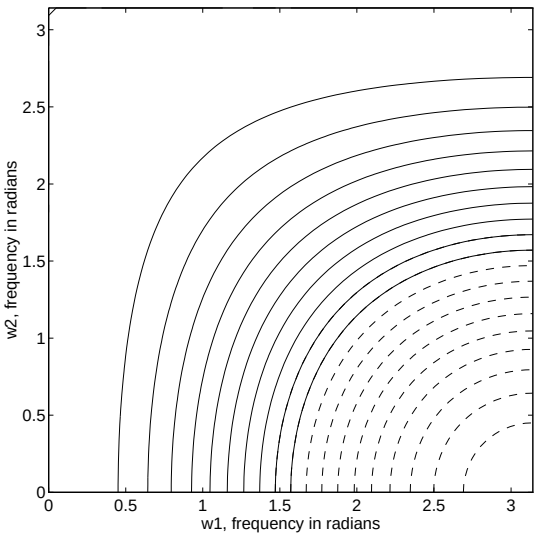
(a) Contours of (4.65).



(b) Contours of (4.71).



(c) Contours of (4.80).



(d) Contours of (4.86).

Figure 4.5. Contours of the subfilters (4.65), (4.71), (4.80), and (4.86). The solid curves correspond to $0 \leq F(\omega_1, \omega_2) \leq 1$ while the dashed curves correspond to $-1 \leq F(\omega_1, \omega_2) < 0$.

As in the previous case, the prototype filter depends on which version of the mapping (4.71) is selected. If (4.71) is used, then

$$\begin{cases} F(0,\omega) &= \cos\omega \\ F(\omega,\pi) &= -\cos\omega = \cos(\omega+\pi). \end{cases} \tag{4.72}$$

Also, note that

$$\begin{cases} Q(0,\omega) &= \sin(\omega/2) \\ Q(\omega,\pi) &= \cos(\omega/2) = \sin((\omega+\pi)/2). \end{cases} \tag{4.73}$$

It follows that

$$\begin{cases} H(0,\omega) &= e^{-j\omega/2}\sin(\omega/2)P(\omega) \\ H(\omega,\pi) &= e^{-j(\omega+\pi)/2}\sin((\omega+\pi)/2)P(\omega+\pi). \end{cases} \tag{4.74}$$

So, with (4.71), it seems appropriate to use an even-length, negative symmetric, prototype filter

$$H(\omega) = e^{-j\omega/2}\sin(\omega/2)P(\omega). \tag{4.75}$$

On the other hand, if the mapping $F_2(\omega_1,\omega_2) = -F(\omega_1,\omega_2)$ is used, then

$$\begin{cases} F(0,\omega) &= \cos(\omega+\pi) \\ F(\omega,\pi) &= \cos\omega \end{cases} \tag{4.76}$$

and

$$\begin{cases} H(0,\omega) &= -je^{-j(\omega+\pi)/2}\cos((\omega+\pi)/2)P(\omega+\pi) = -jH_s(\omega+\pi) \\ H(\omega,\pi) &= -je^{-j\omega/2}\cos(\omega/2)P(\omega) = -jH_s(\omega) \end{cases} \tag{4.77}$$

where $H_s(\omega)$ is the even-length, positive symmetric, prototype filter (4.68).

The contours of (4.71) are plotted in Fig. 4.5(b).

3. $Q(\omega_1,\omega_2) = \sin(\omega_1/2)\sin(\omega_2/2)$

The 2-D designed filter satisfies the following properties

$$\hat{H}(\omega_1,\omega_2) = -\hat{H}(-\omega_1,\omega_2) = \hat{H}(-\omega_1,-\omega_2) = -\hat{H}(\omega_1,-\omega_2) \tag{4.78}$$

and

$$H(0,\omega_2) = H(\omega_1,0) = 0. \tag{4.79}$$

The 2-D design requires the use of shifted and scaled versions of the following transformation mapping:

$$F(\omega_1,\omega_2) = 0.5(1+\cos\omega_1+\cos\omega_2-\cos\omega_1\cos\omega_2). \tag{4.80}$$

Note that

$$\begin{cases} F(\pi,\omega) &= \cos\omega \\ F(\omega,\pi) &= \cos\omega \end{cases} \tag{4.81}$$

leading to

$$H(\pi,\omega) = H(\omega,\pi) = -je^{-j\omega/2}\sin(\omega/2)P(\omega) = -jH_{as}(\omega) \tag{4.82}$$

where $H_{as}(\omega)$ is the even-length, negative symmetric, prototype filter (4.75). Now, if the mapping $F_2(\omega_1,\omega_2) = -F(\omega_1,\omega_2)$ is used, the obtained 2-D filter $H(\omega_1,\omega_2)$ satisfies

$$H(\pi,\omega) = H(\omega,\pi) = jH_s(\omega+\pi) \tag{4.83}$$

where $H_s(\omega)$ is the even-length, positive symmetric, prototype filter (4.68). As in the other cases, the choice of the mapping dictates which prototype is most appropriate for the design.

The contours of (4.80) are shown in Fig. 4.5(c).

4. $Q(\omega_1,\omega_2) = \sin(\omega_1/2)\cos(\omega_2/2)$

   The 2-D designed filter $H(\omega_1,\omega_2)$ is such that

$$\hat{H}(\omega_1,\omega_2) = -\hat{H}(-\omega_1,\omega_2) = -\hat{H}(-\omega_1,-\omega_2) = \hat{H}(\omega_1,-\omega_2) \tag{4.84}$$

   and

$$H(0,\omega_2) = H(\omega_1,\pi) = 0. \tag{4.85}$$

This case is similar to case 2, where $Q(\omega_1, \omega_2) = \cos(\omega_1/2)\sin(\omega_2/2)$, with ω_1 and ω_2 interchanged. The resulting transformation function $F(\omega_1, \omega_2)$ is then given by

$$F(\omega_1, \omega_2) = 0.5(1 + \cos\omega_1 - \cos\omega_2 + \cos\omega_1 \cos\omega_2) \tag{4.86}$$

and its contours are shown in Fig. 4.5(d).

4.4 Design Examples

4.4.1 Example 1: negative centro-symmetric 2-D filter, odd-length prototype

This example corresponds to the design of a 2-D negative centro-symmetric filter whose region of support has odd-length sides. The transformation subfilter $F(\omega_1, \omega_2)$, which is used for this example, is a first-order negative centro-symmetric function of the form

$$A\sin\omega_1 + B\sin\omega_2 + C\sin\omega_1\cos\omega_2 + D\cos\omega_1\sin\omega_2. \tag{4.87}$$

In addition, the following mapping constraints are applied

$$(\omega, \omega) \longleftrightarrow 0 \tag{4.88}$$

$$(\omega, -\omega) \longleftrightarrow \omega \tag{4.89}$$

resulting in the subfilter

$$F(\omega_1, \omega_2) = 0.5(\sin\omega_1 - \sin\omega_2). \tag{4.90}$$

The contours of this subfilter are shown in Fig. 4.6. Constraint (4.88) maps $\omega = 0$ to the line $\omega_1 = \omega_2$ and constraint (4.89) maps the prototype filter to the line $\omega_2 = -\omega_1$.

An odd-length negative-symmetric prototype filter is designed with the following

ideal specifications:

$$D(\omega) = \begin{cases} 1, & (\frac{\pi}{2} - \omega_p) < \omega < (\frac{\pi}{2} + \omega_p) \\ -1, & (-\frac{\pi}{2} - \omega_p) < \omega < (-\frac{\pi}{2} + \omega_p) \\ 0, & \text{otherwise} \end{cases} \tag{4.91}$$

where ω_p controls the passband cutoff frequency and is such that $0 < \omega_p < (\pi/2)$. Figure 4.7 shows the properties of the optimal prototype filter with a length $N = 31$ and $\omega_p = 0.25\pi$ (the actual band edges are placed $\pm 0.01\pi$ around the ideal cutoff frequency). This optimal filter was obtained using the multiple-exchange ascent algorithm of Chapter 3. The frequency response of the designed two-dimensional passband filter is shown in Fig. 4.8. The cutoff of the passband can be controlled through the parameter ω_p of the prototype filter. With $\omega_p = 0.1\pi$, the properties of the resulting prototype and 2-D filters are shown in Figs. 4.9 and 4.10, respectively.

4.4.2 Example 2: complex quadrant-symmetric 2-D filter, even-length prototype

In this example, a complex, quadrant-symmetric, 2-D filter, with a fan-shaped phase response, is designed by transforming an even-length prototype filter. The resulting filter has different "phase" specifications in the passbands and can be used, for instance, to process the phase information in images. The new design procedure presented in section 4.2.2 is used to transform the even-length prototype into the desired 2-D filter by means of an appropriate subfilter which is derived below.

Note that the even-length FIR prototype filter $H(\omega)$ is of the form (see section 4.2.2)

$$H(\omega) = e^{-j\omega/2}\hat{H}(\omega) \tag{4.92}$$

where $\hat{H}(\omega)$ is the function that is actually transformed into the 2-D filter and which needs to be specified. For this example, $\hat{H}(\omega)$ is chosen such that it approximates the

following ideal function:

$$\hat{D}(\omega) = \begin{cases} e^{j\frac{\pi}{4}}, & 0 \leq |\omega| < \omega_p \qquad \text{(band 1)} \\ 1, & \omega_p < |\omega| < \pi \qquad \text{(band 2)} \end{cases} \tag{4.93}$$

where ω_p is the cutoff frequency between the different "phase" bands. The exchange algorithm of Chapter 3 is then used to compute the coefficients $\{h_n\}$ of the even-length prototype filter whose frequency response $H(\omega)$ approximates

$$D(\omega) = e^{-j\omega/2}\hat{D}(\omega). \tag{4.94}$$

As mentioned in section 4.2.2, the subfilter $F(\omega_1, \omega_2)$ replaces the function $\cos(\omega/2)$. Since it is desired to obtain a positive quadrant-symmetric filter, a quadrant-symmetric subfilter is used (see section 4.3). Now, in order to map each 1-D band into a separate 2-D region, the subfilter must satisfy the following specifications:

$$\begin{cases} \cos(\omega_p/2) < F(\omega_1, \omega_2) \leq 1, & \text{in band 1} \\ -1 \leq F(\omega_1, \omega_2) < \cos(\omega_p/2), & \text{in band 2} \end{cases} \tag{4.95}$$

In this example, it is desirable to map the values of the band $0 \leq |\omega| < \omega_p$ to the wedge-shaped region $|\omega_2| > |\omega_1|$, and the values of the band $\omega_p < |\omega| < \pi$ to the region $|\omega_2| < |\omega_1|$. For that purpose, a first-order positive quadrant-symmetric subfilter is used with the mapping constraint

$$(w, w) \longleftrightarrow \cos\omega_p \tag{4.96}$$

which maps the cutoff frequency ω_p to the line $\omega_2 = \omega_1$ separating the two fan-shaped regions. As a result, the following first-order subfilter is obtained

$$F(\omega_1, \omega_2) = \cos\omega_p + t_{10}(\cos\omega_1 - \cos\omega_2) \tag{4.97}$$

where the parameter t_{10} needs to be positive and selected such that

$$-1 \leq F(\omega_1, \omega_2) \leq 1. \tag{4.98}$$

It can be easily verified that the biggest range of admissible values for $F(\omega_1, \omega_2)$ is obtained when

$$t_{10} = 0.5(1 - \cos \omega_p). \tag{4.99}$$

The selection (4.99) for t_{10} produces a well-behaved mapping for all values of ω_p such that $0 \leq \omega_p \leq \pi$. With $\omega_p = \pi/4$, the contours of the quadrant-symmetric subfilter are shown in Fig. 4.11. The properties of the optimal prototype filter and of the designed 2-D fan-shaped "phase" filter are shown in Figs. 4.12 and 4.13, respectively. In the actual design of the prototype filter, a transition band of width 0.1π was used around the ideal cutoff frequency $\omega_p = \pi/4$, and the starting index N_1 of the impulse response was set to the value -14.

It is easy to verify that the resulting 2-D "phase" filters are optimal on the obtained passband and stopband regions [3]. Note that, in this case, the designed filters can be regarded as having a "phase" passband (band 1) and a "phase" stopband (band 2).

4.4.3 Example 3

This example corresponds to the design of a 2-D filter whose region of support has even-length sides. For this purpose, an even-length prototype filter needs to be transformed using one of the subfilters (4.65), (4.71), (4.80), and (4.86). The selection of the subfilter is determined by the symmetry property of the desired 2-D filter. In this example, it is desirable to have the designed 2-D filter $H(\omega_1, \omega_2)$ satisfy the following symmetry condition:

$$\hat{H}(\omega_1, \omega_2) = -\hat{H}(-\omega_1, \omega_2) = \hat{H}(-\omega_1, -\omega_2) = -\hat{H}(\omega_1, -\omega_2). \tag{4.100}$$

Therefore, subfilter (4.80) is used to transform an even-length negative-symmetric prototype filter with the following ideal specifications:

$$D(\omega) = \begin{cases} e^{-j\omega/2}, & \omega_{p1} < \omega < \omega_{p2} \\ -e^{-j\omega/2}, & -\omega_{p2} < \omega < -\omega_{p1} \\ 0, & \text{otherwise} \end{cases} \tag{4.101}$$

where ω_{p1} and ω_{p2} control the passband width and $0 < \omega_{p1}, \omega_{p2} \leq \pi$. The prototype filter length N is set to 30.

Recall, from section 4.3, that the subfilter (4.80) maps the prototype filter characteristics to the lines $\omega_1 = \pi$ and $\omega_2 = \pi$. With $\omega_{p1} = \pi/2$ and $\omega_{p2} = \pi$, the properties of the optimal prototype are shown in Fig. 4.14. In the actual design of the prototype filter, the ideal cutoff ω_{p1} was represented by a stopband cutoff frequency equal to 0.45π and a passband cutoff frequency equal to 0.55π. Figure 4.15 plots the transformed 2-D function $\hat{H}(\omega_1, \omega_2) = H(\omega_1, \omega_2)e^{j(\omega_1+\omega_2)/2}$. The resulting 2-D filter $H(\omega_1, \omega_2)$ is a highpass filter of size 30×30 and satisfies the desired symmetry condition (4.100), as illustrated in Fig. 4.15.

With $\omega_{p1} = \pi/4$ and $\omega_{p2} = \pi/2$, the optimal prototype filter characteristics are shown in Fig. 4.16, and a bandpass 2-D filter $H(\omega_1, \omega_2)$ is obtained as shown in Fig. 4.17. A transition band of width 0.02π was used around the cutoff edges of the prototype filter.

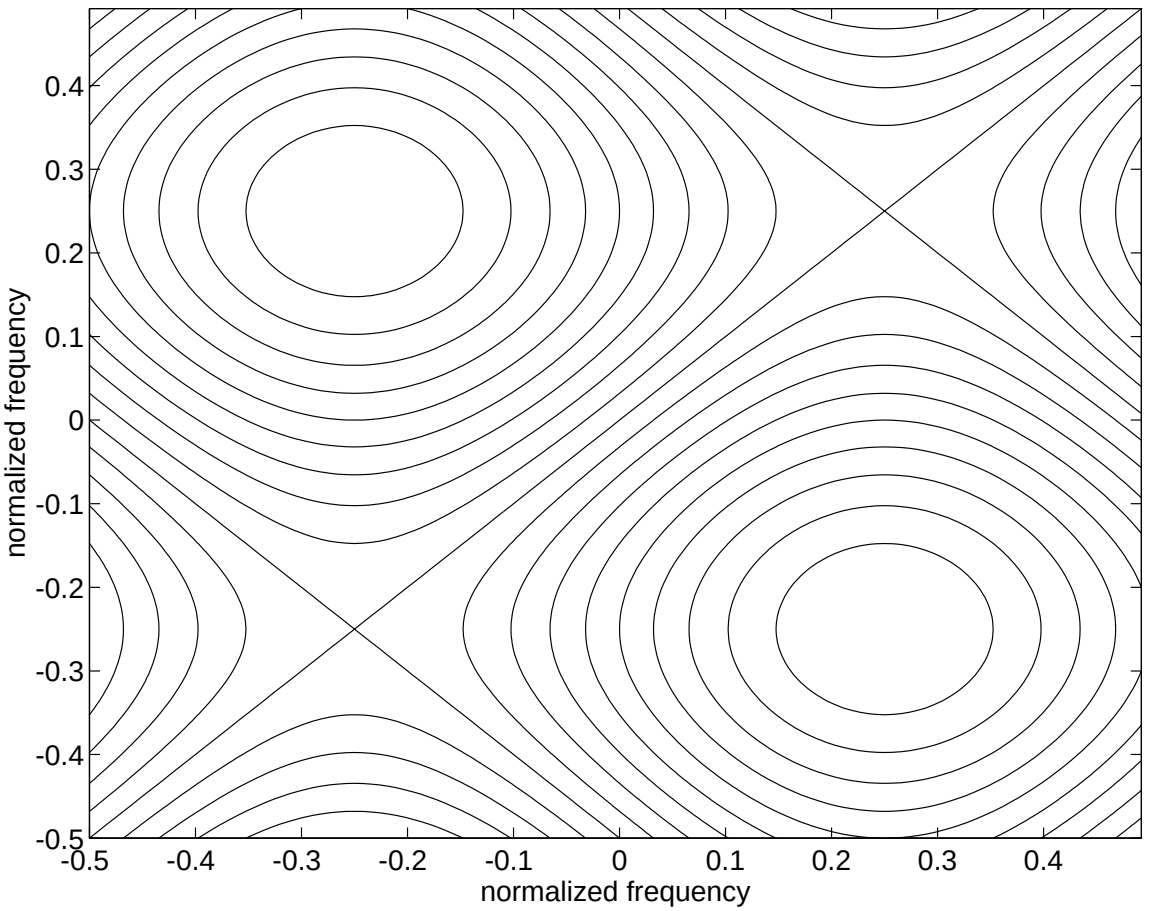


Figure 4.6. Example 1: contours of the negative centro-symmetric subfilter.

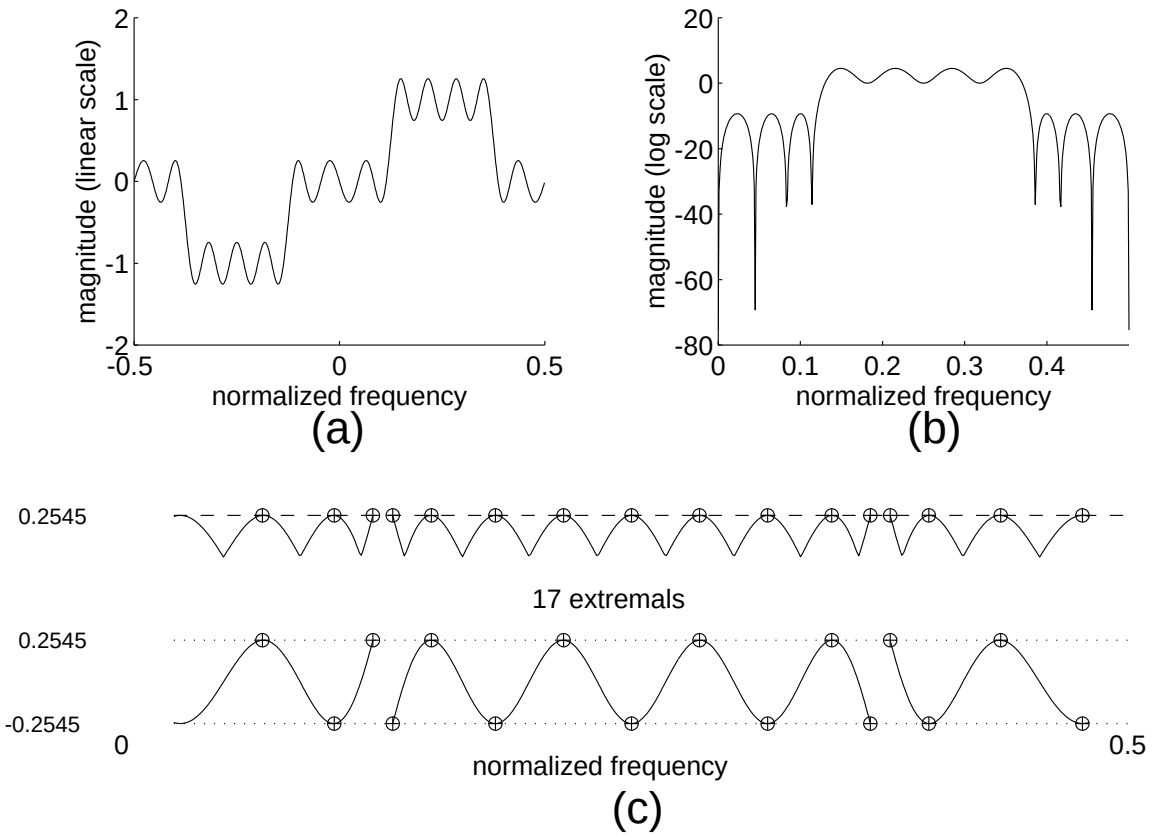


Figure 4.7. Example 1: prototype filter, $N = 31$ and $\omega_p = \pi/4$. (a) FIR filter magnitude response. (b) FIR filter magnitude response in dB. (c) Magnitude of the error (top), and real alternating error (bottom).

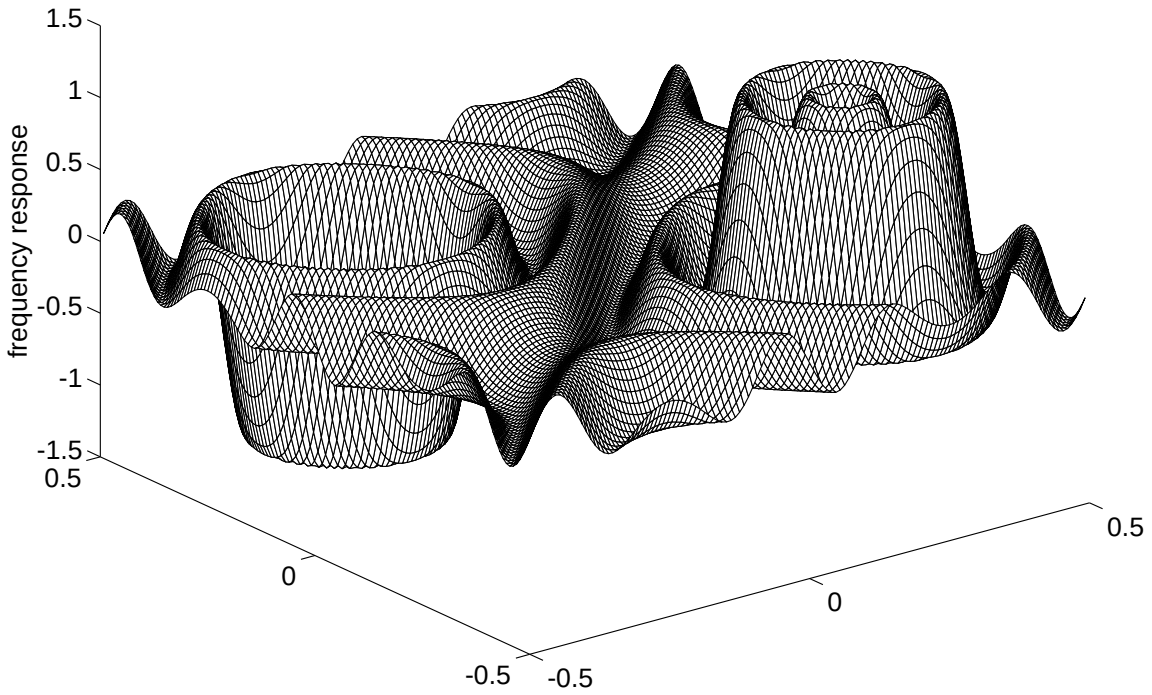


Figure 4.8. Example 1: frequency response of the transformed 2-D filter, $\omega_p = \pi/4$.

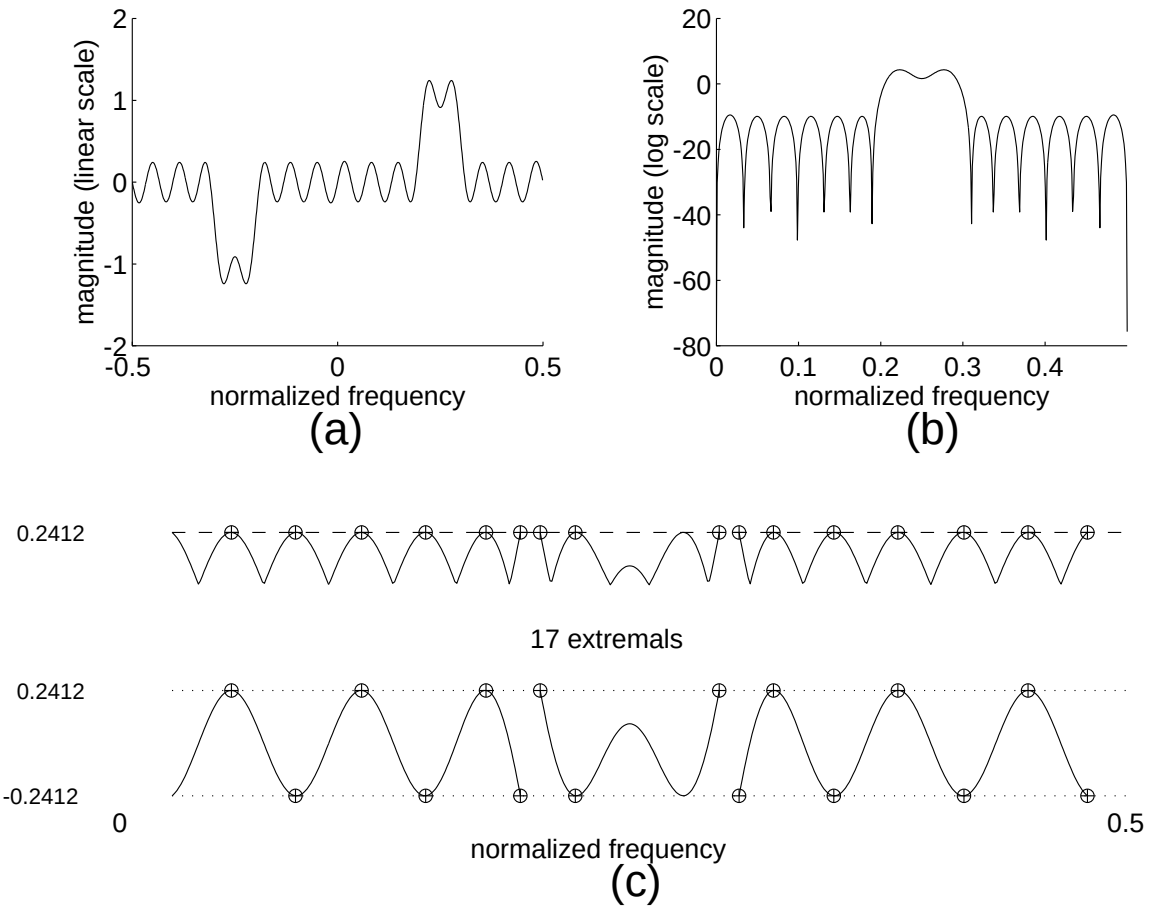


Figure 4.9. Example 1: prototype filter, $N = 31$ and $\omega_p = \pi/10$. (a) FIR filter magnitude response. (b) FIR filter magnitude response in dB. (c) Magnitude of the error (top), and real alternating error (bottom).

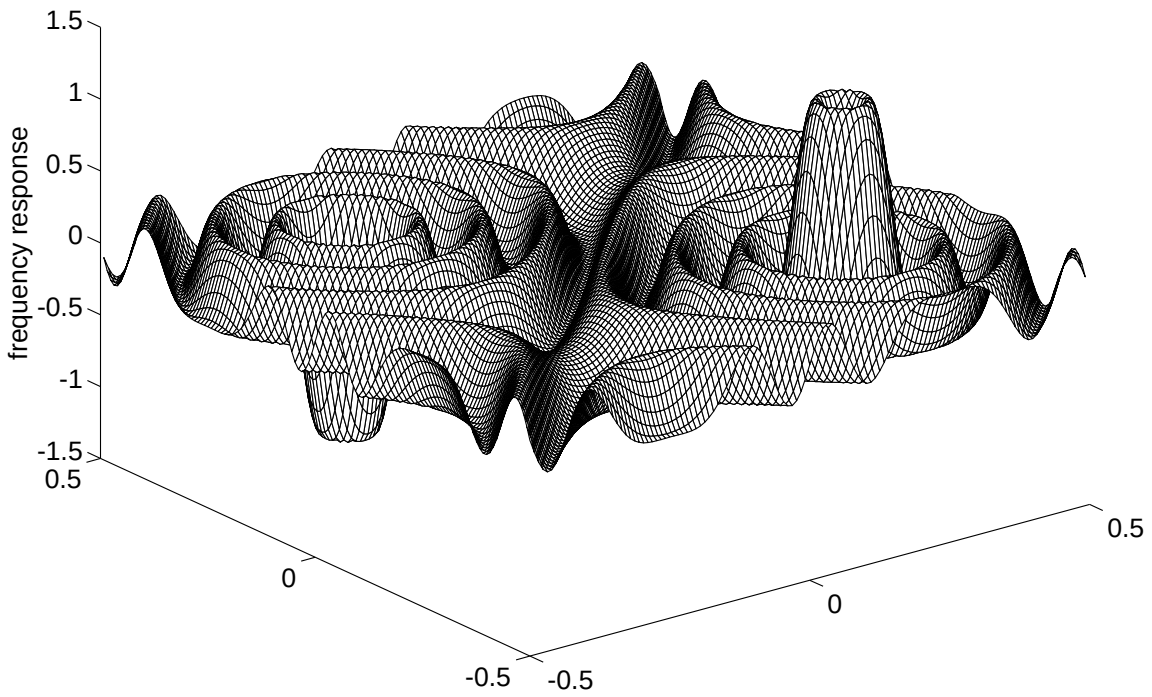


Figure 4.10. Example 1: frequency response of the transformed 2-D filter, $\omega_p = \pi/10$.

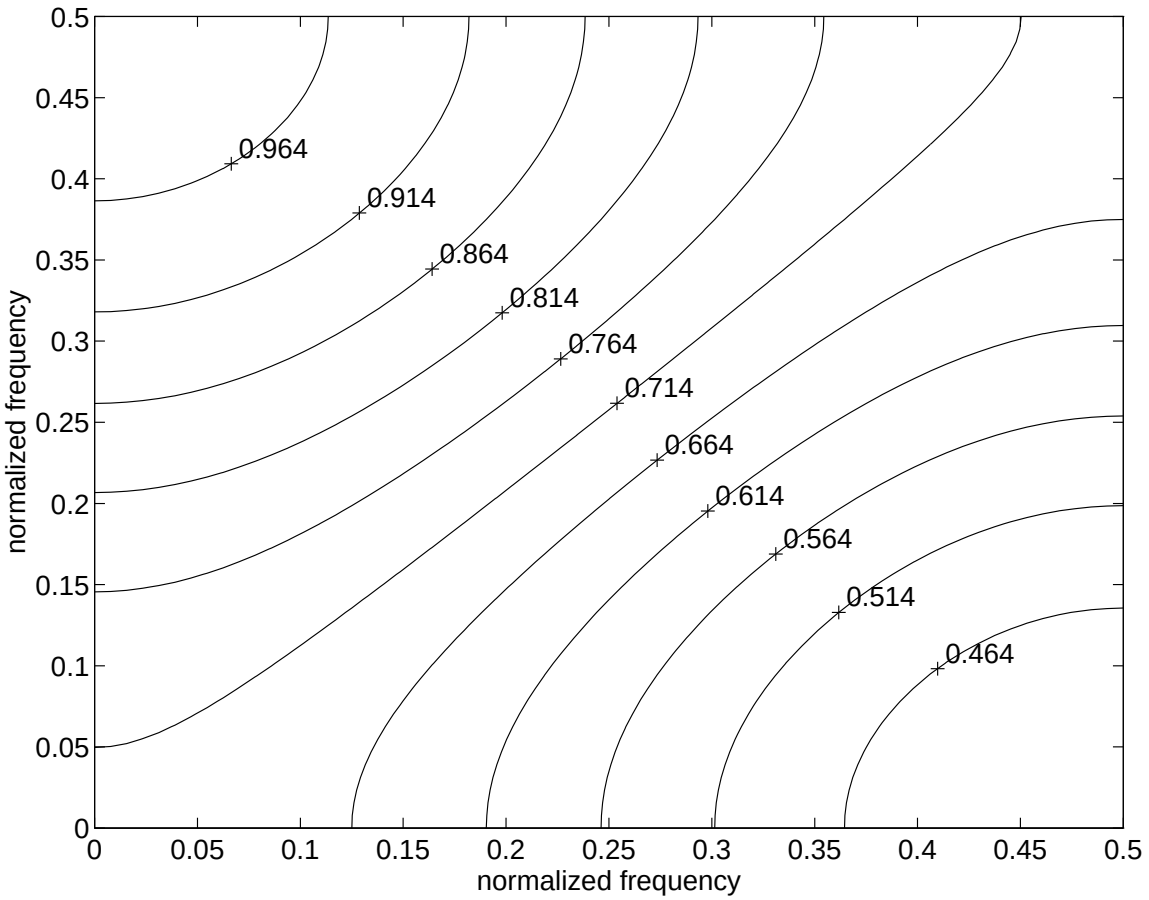


Figure 4.11. Example 2: contours of the quadrant-symmetric subfilter for designing the complex fan-shaped 2-D filter.

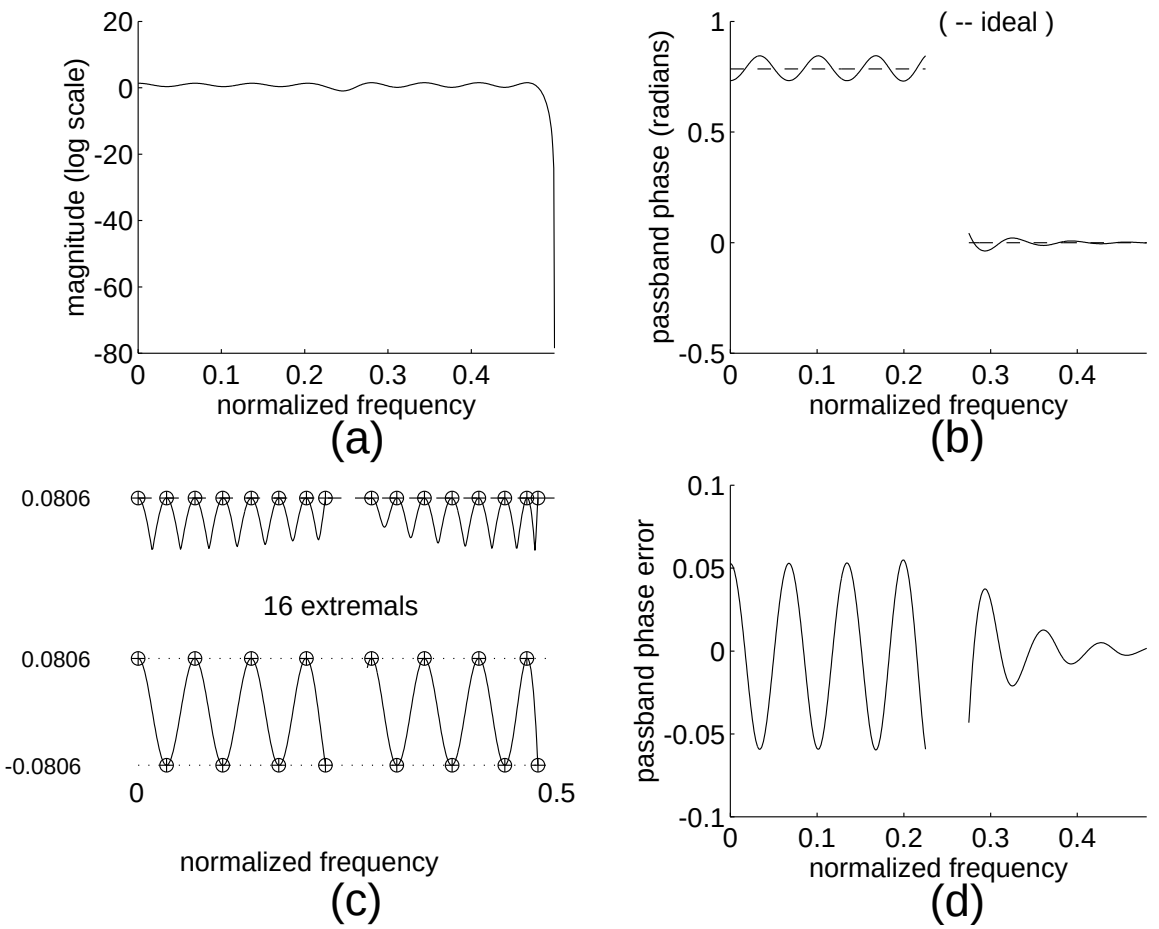
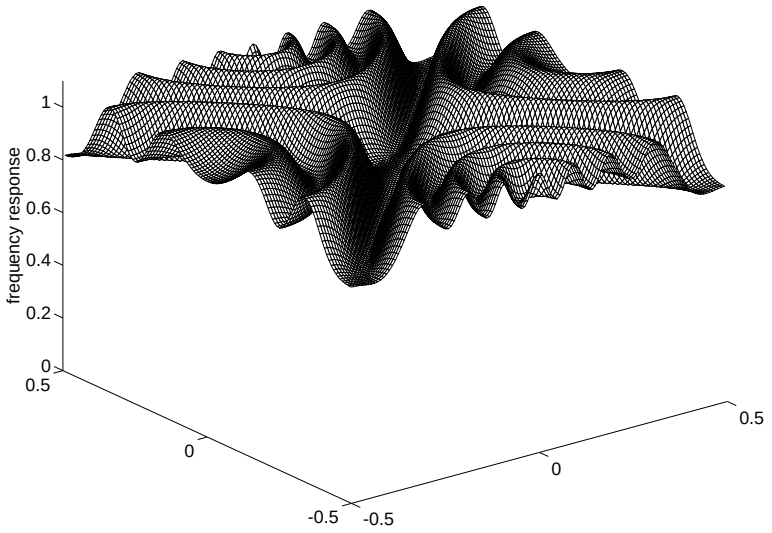
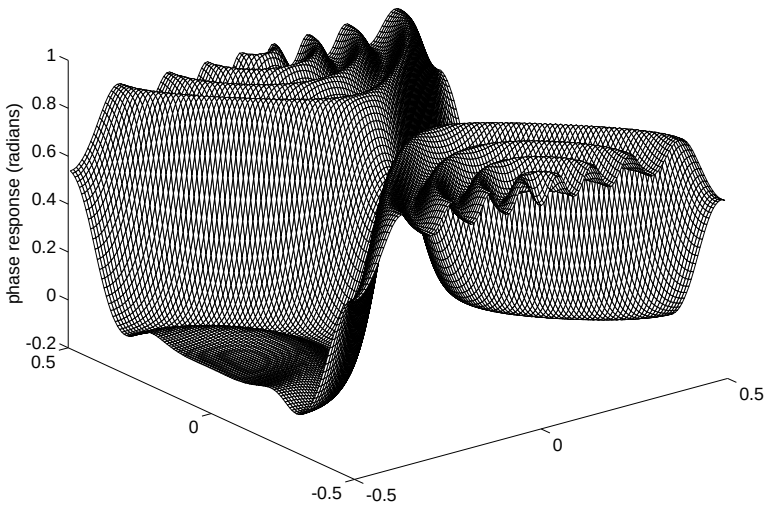


Figure 4.12. Example 2: prototype filter, $N = 30$ and $\omega_p = \pi/4$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired (dashed) passband phase responses plotted without the half-sample delay. (c) Magnitude of the error (top), and alternating real phase-rotated error (bottom). (d) Phase error in the two bands.



(a)



(b)

Figure 4.13. Example 2: two-dimensional fan-shaped phase filter designed by transforming a positive-symmetric even-length prototype with cutoff $\omega_p = \pi/4$. (a) Filter magnitude response. (b) Filter fan-shaped phase response.

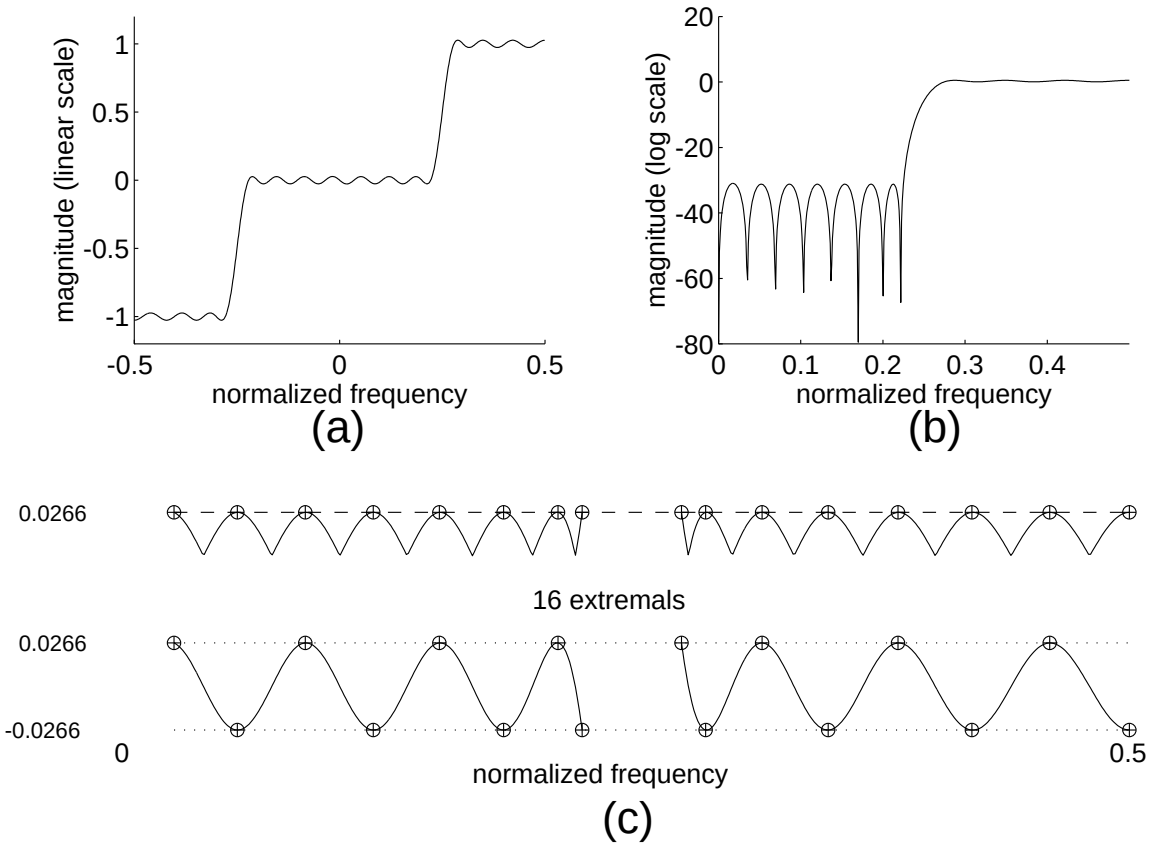


Figure 4.14. Example 3: prototype filter, $N = 30$, $\omega_{p1} = \pi/2$, and $\omega_{p2} = \pi$. (a) FIR filter magnitude response. (b) FIR filter magnitude response in dB. (c) Magnitude of the error (top), and real alternating error (bottom).

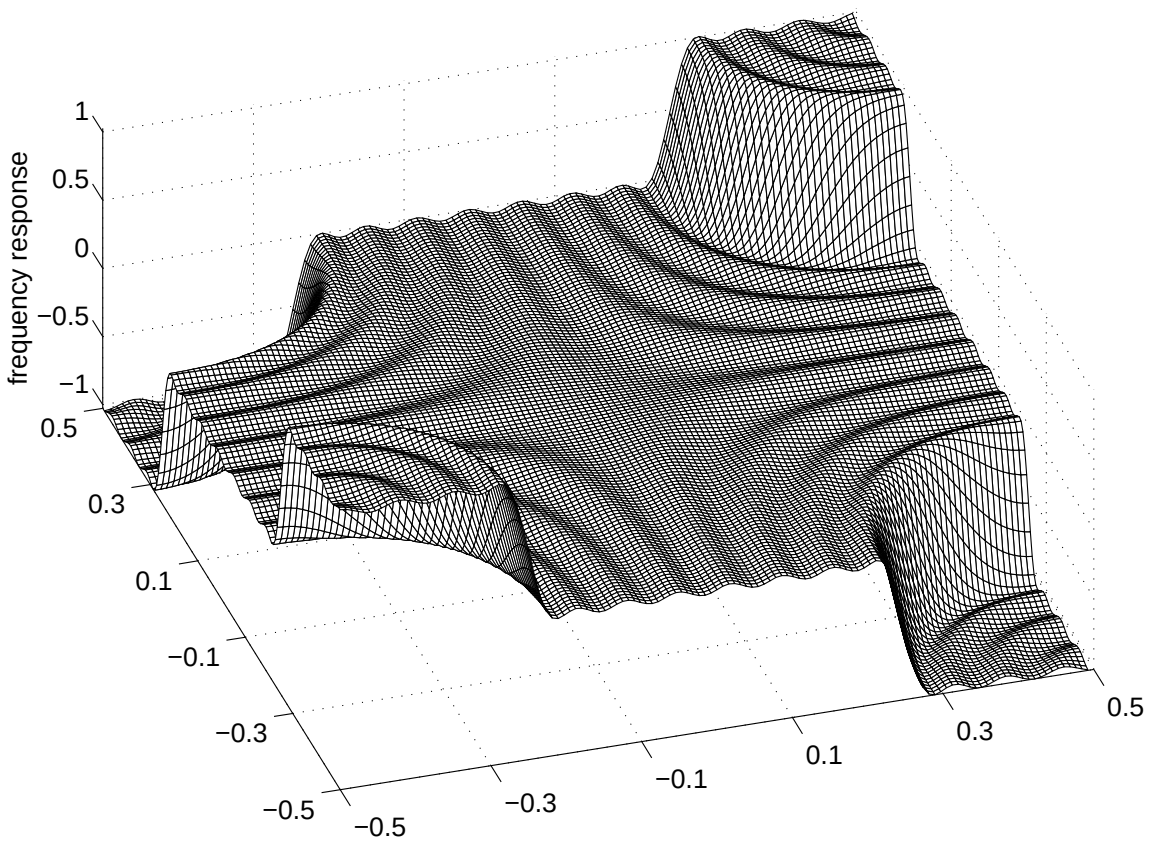


Figure 4.15. Example 3: frequency response of the transformed 2-D filter of size 30×30 ($\omega_{p1} = \pi/2$, and $\omega_{p2} = \pi$).

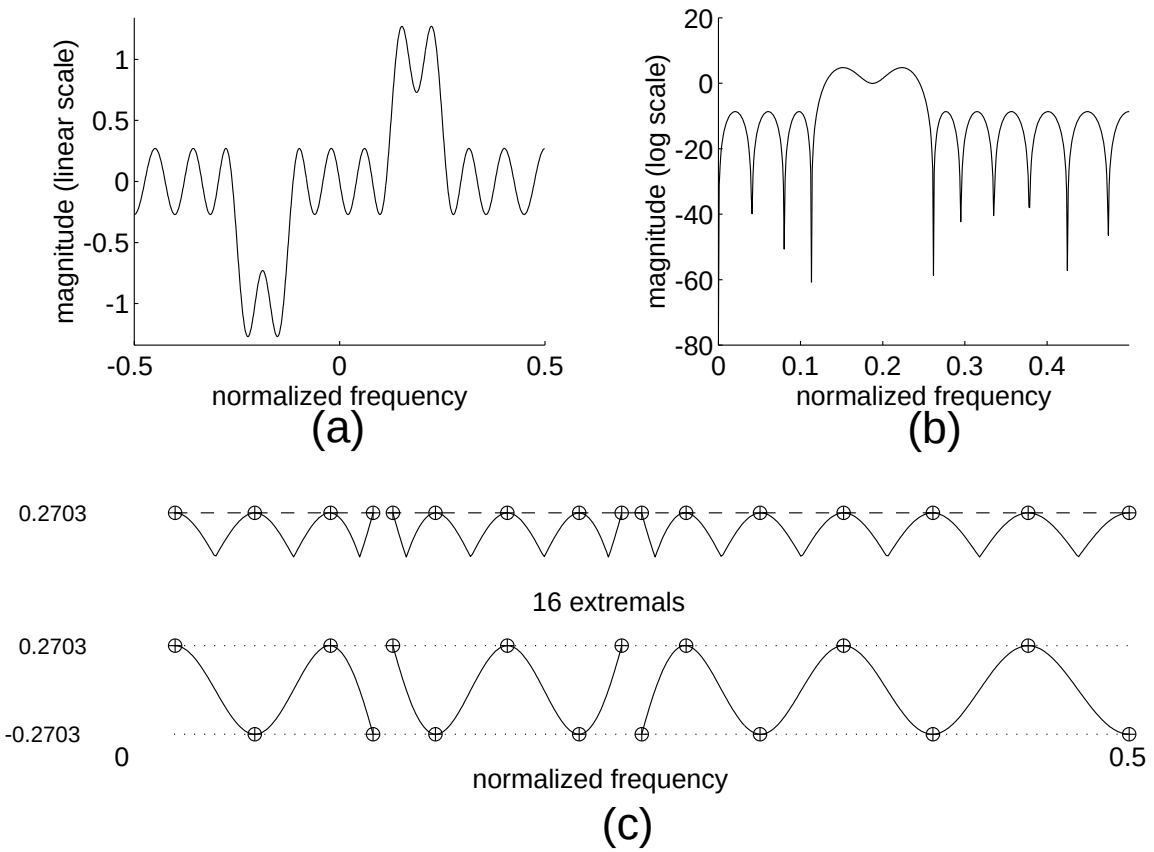


Figure 4.16. Example 3: prototype filter, $N = 30$, $\omega_{p1} = \pi/4$, and $\omega_{p2} = \pi/2$. (a) FIR filter magnitude response. (b) FIR filter magnitude response in dB. (c) Magnitude of the error (top), and real alternating error (bottom).

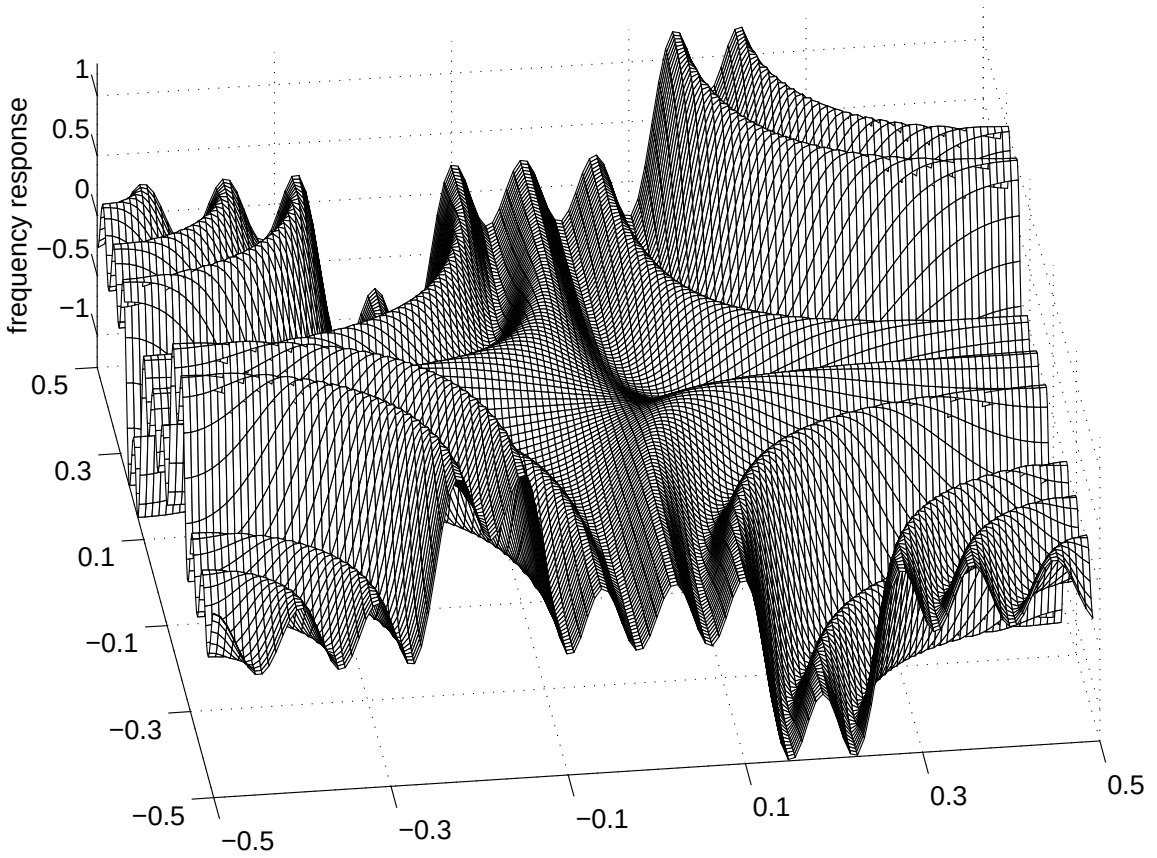


Figure 4.17. Example 3: frequency response of the transformed 2-D filter of size 30×30 ($\omega_{p1} = \pi/4$, and $\omega_{p2} = \pi/2$).

Table 4.1. The designed 2-D filter has a rectangular region of support whose size depends on the prototype filter length, the transformation subfilter size, and the transformation technique used.

	Prototype filter length	Transformation filter size	Designed 2-D filter size
odd-length prototype ↓ odd-length 2-D filter	$2L+1$	$(2Q+1)\times(2R+1)$	$(2LQ+1)\times(2LR+1)$
even-length prototype ↓ odd-length 2-D filter	$2L$	$(2Q+1)\times(2R+1)$	$(2(2L-1)Q+1)$ × $(2(2L-1)R+1)$
even-length prototype ↓ even-length 2-D filter	$2L$	$(2Q+1)\times(2R+1)$	$(2(L-1)Q+2)$ × $(2(L-1)R+2)$

4.5 Summary

This chapter presents an extension of the transformation method so that new types of multi-dimensional filters can be designed. As a result of this extension, complex, even- and odd-length, prototype filters can be transformed into complex, two-dimensional FIR filters. In addition, both positive- and negative-symmetric prototypes can be used in the 2-D design. Procedures are presented for designing both positive- and negative-symmetric 2-D filters, with complex or real-valued impulse responses. Table 4.1 gives the size of the designed 2-D impulse response in terms of the prototype filter length and the transformed subfilter size. Multi-dimensional filters, with dimensions higher than two, can be obtained by increasing the dimensionality of the transformation function.

CHAPTER 5

Efficient Design of Families of Filters by Transformation

This chapter introduces the concept of designing a complete family of filters by regarding it as a single multi-dimensional filter. In addition, the problem of designing the complete family of filters can be reduced to the design of a single one-dimensional filter which is then mapped to produce all the desired one- or multi-dimensional filters using transformations. A new efficient transformation procedure is presented for designing complete sets of complex and real, one- and multi-dimensional, variable cutoff filters. Whereas the McClellan transformation method is suitable for designing one multi-dimensional filter at a time, the proposed procedure can design a large set of one- and multi-dimensional filters by transforming a single one-dimensional prototype filter. The presented design procedure consists of three main parts:

1. The design of a subfilter approximating the contours of the desired set of filters.

2. The selection and design of a one-dimensional prototype filter.

3. The retrieval or realization of the individual filters in the set.

The proposed procedure is very fast because most of the computing time is devoted to the design of a single complex- or real-valued 1-D filter which is then transformed to produce the needed set of filters. It has the additional advantage that both the one- and multi-dimensional filters can be designed and realized very efficiently without having to explicitly compute the coefficients of the separate filters.

The new design procedure is described in details by considering the design of velocity-dependent depth migration filters (Section 5.1) and exactly linear-phase fil-

ters (Section 5.2). Although the filters designed by this technique are not optimal in general, the designer can control the Chebyshev error norm through the design of the prototype filter and the transformation subfilter.

5.1 Families of Filters for 2-D and 3-D Depth Migration

An overview of seismic migration filters was presented in Chapter 3 (section 3.4.2). It was shown that two- and three-dimensional depth migration can be performed using one- and two-dimensional migration filters, respectively. For a given velocity, the depth extrapolation is usually done one frequency at a time by convolving the seismic wavefield with a complex-valued, frequency- and velocity-dependent digital filter. This process requires the design of a complete set of extrapolation filters, one filter for each possible frequency-velocity pair. This depth extrapolation, performed for each frequency independently, is the basic operation of finite-difference migration [48]. The computational complexity of depth migration depends directly on the efficiency of this operation. In the previous methods [48, 49], the coefficients of all possible 1-D migration filters were precomputed and tabulated. The two-dimensional extrapolation filters can then be obtained from the corresponding one-dimensional migration filters using transformations. Instead of independently designing the frequency- and velocity-dependent migration filters, this section presents an efficient design procedure which produces a complete set of needed filters without having to compute the coefficients of the separate filters. This procedure has the additional advantage that both the 1-D and 2-D migration filters can be realized efficiently and need not have their coefficients precomputed and tabulated.

Given a wave velocity v and temporal frequency ω, a 1-D extrapolation filter

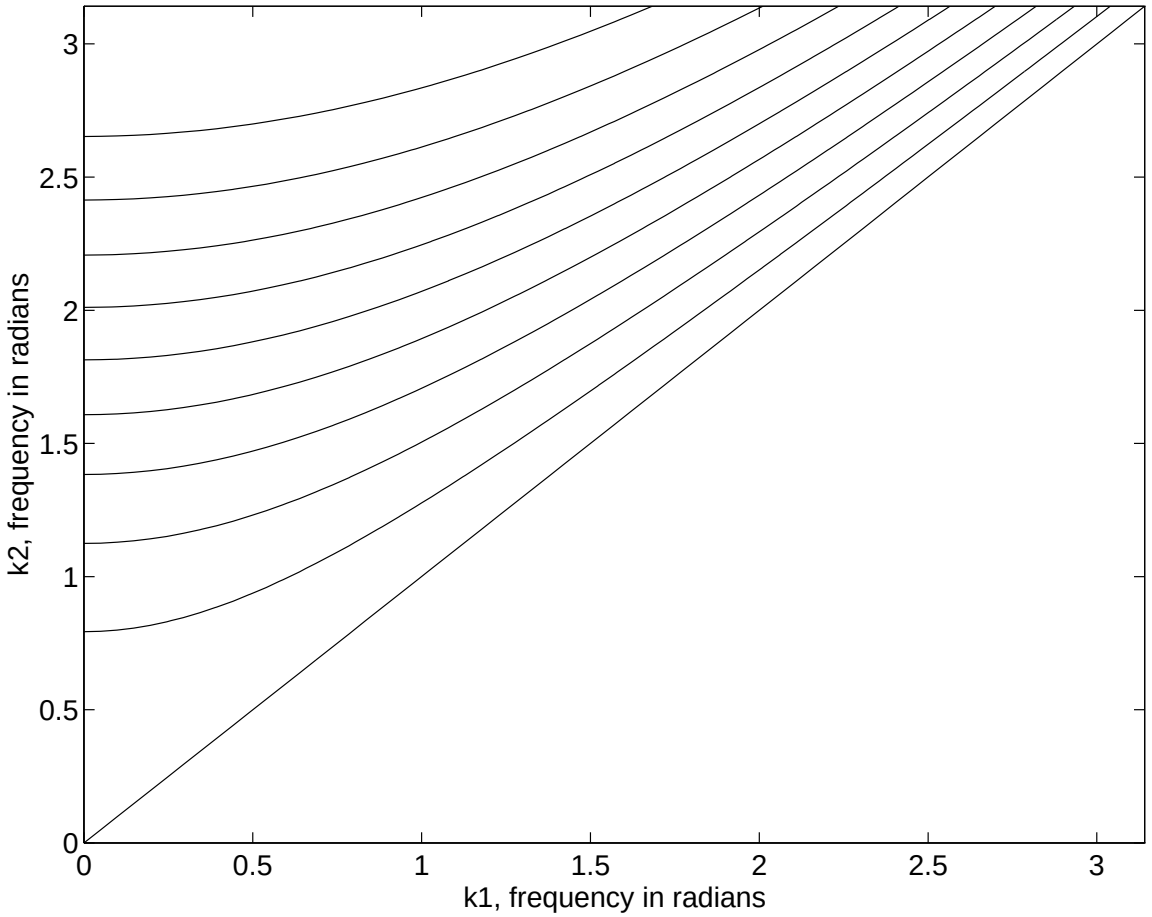


Figure 5.1. Ideal hyperbolic contours of the migration filter family.

$H(k_x)$ (3.56) has a positive symmetric frequency response that approximates

$$D(k_x) = \begin{cases} e^{jb\sqrt{(\frac{\Delta x}{\Delta t}\frac{\omega}{v})^2 - k_x^2}}, & |k_x| < |\alpha| \\ 0, & \text{otherwise} \end{cases} \tag{5.1}$$

where $\alpha = (\Delta x/\Delta t)(\omega/v)$. Because $D(k_x)$ is positive symmetric and periodic with period 2π in k_x, the approximation need only be done on the interval $0 \leq k_x \leq \pi$. In (5.1), k_x is the spatial frequency in radians, and $b = \Delta z/\Delta x$ is a constant related to the vertical and horizontal sampling periods. The dimensionless term $(\Delta x\,\omega)/(\Delta t\,v)$ represents the (temporal) frequency- and velocity-dependent cutoff

spatial frequency of $D(k_x)$. In the following, it is assumed that the normalized cutoff frequency $(\Delta x\,\omega)/(\Delta t\,v) \in [0, \pi]$. For a fixed b, an entire family of frequency- and velocity-dependent migration filters (5.1) can be approximated as a single complex-valued 2-D "fan-shaped" filter with the ideal frequency response

$$D(k_1, k_2) = \begin{cases} e^{jb\sqrt{k_2^2 - k_1^2}}, & |k_1| < |k_2| \\ 0, & \text{otherwise} \end{cases} \tag{5.2}$$

where $k_2 = (\Delta x\,\omega)/(\Delta t\,v)$ and $k_1 = k_x$. The isopotentials of $D(k_1, k_2)$ are shown in Fig. 5.1 and correspond to the hyperbolic contours

$$k_2^2 - k_1^2 = \text{constant}. \tag{5.3}$$

Each horizontal cross-section of $D(k_1, k_2)$ corresponds to a fixed value of k_2, and represents uniquely the 1-D extrapolation filter with cutoff frequency k_2. Since $D(k_1, k_2)$ is quadrant symmetric, it is sufficient to perform the approximation in the first quadrant $0 \leq k_1, k_2 \leq \pi$. Direct 2-D approximation of $D(k_1, k_2)$ is not adequate in this case not only because of the large computational complexity but also because it does not provide a simple means to retrieve the coefficients of the individual frequency- and velocity-dependent filters. These problems are eliminated if $D(k_1, k_2)$ is approximated using transformations [50].

The McClellan transformation method (see section 4.1) maps a 1-D prototype filter $P(k)$ of the form

$$P(k) = p_o + 2\sum_{n=1}^{L} p_n \cos(nk) \tag{5.4}$$

into a 2-D filter $H(k_1, k_2)$ by means of the substitution

$$\cos k = F(k_1, k_2). \tag{5.5}$$

The following 2-D filter is obtained

$$H(k_1, k_2) = p_o + 2\sum_{n=1}^{L} p_n T_n[F(k_1, k_2)]. \tag{5.6}$$

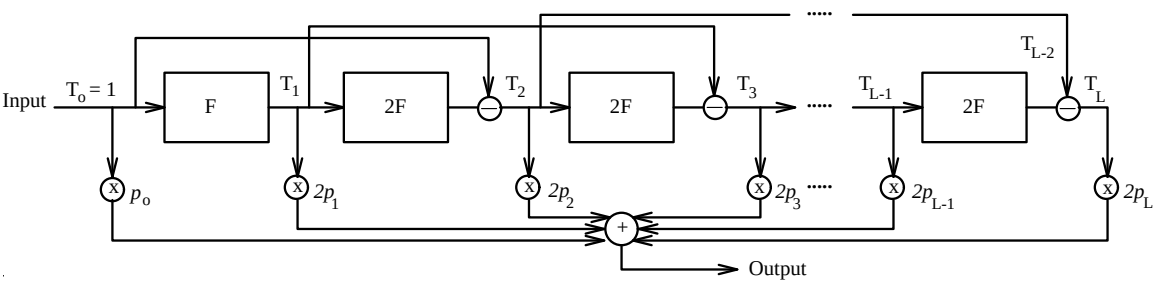


Figure 5.2. Chebyshev structure with $\{p_n\}_{n=0}^{L}$ and F representing the prototype coefficients and the transformation subfilter, respectively.

Note that the transformation function $F(k_1, k_2)$ must be the frequency response of a 2-D FIR filter to ensure that the resulting $H(k_1, k_2)$ is the frequency response of an FIR filter. Furthermore, in order to have a well-behaved mapping, $F(k_1, k_2)$ is required to take the same range of values as the *cosine* function it replaces, i.e. $|F(k_1, k_2)| \leq 1$.

As mentioned in section 4.1, the transformation method breaks down the 2-D design problem into two decoupled smaller design problems: the design of the mapping function (subfilter) $F(k_1, k_2)$ and the design of the 1-D prototype filter $P(k)$. $F(k_1, k_2)$ defines the shape of the contours of the resulting transformed 2-D filter, while $P(k)$ determines the values of the frequency response $H(k_1, k_2)$ along these contours. By virtue of the recurrence formula for Chebyshev polynomials [32], the produced filters can be realized efficiently using the Chebyshev structure shown in Fig. 5.2 [59].

5.1.1 Part 1: Subfilter Design

The contour specifications of the desired "fan-shaped" migration filter (5.2) are illustrated in Fig. 5.1. Since $D(k_1, k_2)$ is quadrant symmetric, the approximating filter $H(k_1, k_2)$ and, therefore, the transformation subfilter must be the frequency responses of quadrant symmetric 2-D FIR filters. It follows that the general form of the mapping

subfilter $F(k_1, k_2)$ may be expressed as

$$F(k_1, k_2) = \sum_{q=0}^{Q} \sum_{r=0}^{R} t_{qr} \cos(qk_1) \cos(rk_2) \tag{5.7}$$

where the $(Q+1) \times (R+1)$ coefficients $\{t_{qr}\}$ are to be determined such that $F(k_1, k_2)$ best approximates the hyperbolic isopotentials of $D(k_1, k_2)$. The integers Q and R control the size of the subfilter. Usually, the size of the subfilter is kept small since the computational complexity and the implementation cost increase with the subfilter size.

Optimization techniques have been proposed in [60, 57, 63] for designing subfilters that approximate contours with a variety of shapes. However, these techniques are not suitable for constraining the shapes of all contours simultaneously and are mostly used to approximate the shape of the cutoff line only. Design methods, which allow the designer to constrain the shape of all the contours simultaneously, have been presented in [61, 64]. In these methods, the desired response of the subfilter is represented in a functional form. The subfilter coefficients $\{t_{qr}\}$ are then computed so that $F(k_1, k_2)$ best approximates the desired functional response with respect to a specified error norm. It is not difficult to see that the following ideal subfilter response realizes the desired contour characteristics of (5.2):

$$F_D(k_1, k_2) = \sin\left(\frac{k_2^2 - k_1^2}{2\pi}\right) = \cos\left(\frac{\pi}{2} - \frac{k_2^2 - k_1^2}{2\pi}\right). \tag{5.8}$$

$F_D(k_1, k_2)$ is quadrant symmetric and has hyperbolic contours as $D(k_1, k_2)$. In addition, $F_D(k_1, k_2)$ is positive in the passband and negative in the stopband, which is a characteristic of the subfilters used in designing "fan-shaped" filters with a cutoff line $k_1 = k_2$ [65, 64].

The subfilter design problem is now briefly stated as follows. Among all the possible coefficients, find the coefficients $\{t_{qr}, \quad 0 \leq q \leq Q, \quad 0 \leq r \leq R\}$ such that the weighted error

$$E(k_1, k_2) = W(k_1, k_2)[F_D(k_1, k_2) - F(k_1, k_2)] \tag{5.9}$$

is minimized in the least square or Chebyshev (minimax) sense. $W(k_1, k_2)$ in (5.9) is a positive weighting function. The least square minimization can be easily performed using a classical least square formulation, and the Chebyshev minimization can be done by linear programming or by using the method presented in section 3.2.3 of Chapter 3. It has been observed that both norms produce similar results. Note that a useful well-behaved mapping is obtained only if

$$|F(k_1, k_2)| \leq 1. \tag{5.10}$$

The constraint (5.10) could either be incorporated into the approximation problem or satisfied by performing a simple scaling operation [57]. The scaling operation does not affect the shape of the contours but it does change the 1-D prototype frequency associated with each contour. Thus, scaling might complicate finding the specifications of the proper prototype filter. Other techniques for constructing well-behaved mappings can be found in [60].

5.1.2 Part 2: Prototype Filter Design

With the ideal subfilter specified as in (5.8), one needs to determine the ideal characteristics of the prototype filter which, when mapped, would produce $D(k_1, k_2)$. For that purpose, notice from (5.8) that

$$F_D(k_1, \pi - k_1) = \cos k_1. \tag{5.11}$$

So, the prototype filter $P(k)$ is mapped to the line

$$k_2 = \pi - k_1. \tag{5.12}$$

In fact, (5.11) and (5.6) lead to

$$H(k_1, \pi - k_1) = P(k_1) \approx D(k_1, \pi - k_1). \tag{5.13}$$

where $\approx$ means "is an approximation of". It follows that the ideal response of the prototype filter needed to produce the desired family of migration filters is given by

$$P_D(k_1) = \begin{cases} e^{jb\sqrt{\pi^2 - 2\pi k_1}}, & |k_1| < \dfrac{\pi}{2} \\ 0, & \text{otherwise} \end{cases} \tag{5.14}$$

Note also that $P_D(\frac{\pi}{2} - \frac{k_2^2 - k_1^2}{2\pi}) = D(k_1, k_2)$. Therefore, accurate approximations to (5.8) and (5.14) would result in an accurate approximation $H(k_1, k_2)$ to $D(k_1, k_2)$ given by (5.2). The exchange algorithm of Chapter 3 is used to compute the coefficients of the optimal equiripple prototype filter $P(k_1)$ approximating (5.14).

5.1.3 Part 3: Generating the 2-D and 3-D Depth Migration Filters

After computing the optimal mapping coefficients $\{t_{qr}\}$ approximating (5.8) and the optimal prototype coefficients $\{p_n\}$ approximating (5.14), one obtains an approximation to $D(k_1, k_2)$ with the following frequency response

$$H(k_1, k_2) = p_o + 2\sum_{n=1}^{L} p_n T_n[\underbrace{\sum_{q=0}^{Q}\sum_{r=0}^{R} t_{qr}\cos(qk_1)\cos(rk_2)}_{F(k_1,k_2)}]. \tag{5.15}$$

The 2-D depth migration filter $H(k_1)$ having a fixed cutoff frequency k_c and approximating

$$D(k_1) = \begin{cases} e^{jb\sqrt{k_c^2 - k_1^2}}, & |k_1| < |k_c| \\ 0, & \text{otherwise} \end{cases} \tag{5.16}$$

is equal to $H(k_1, k_c)$ and is obtained by substituting for F in the Chebyshev structure of Fig. 5.2 the 1-D transformation subfilter

$$F(k_1) = F(k_1, k_c) = \sum_{q=0}^{Q}(\sum_{r=0}^{R} t_{qr}\cos rk_c)\cos(qk_1) \tag{5.17}$$

and by substituting for $\{p_n\}$ the coefficients of the 1-D complex prototype filter approximating (5.14). Note, from (5.17), that the coefficients of the 1-D subfilter $F(k_1)$

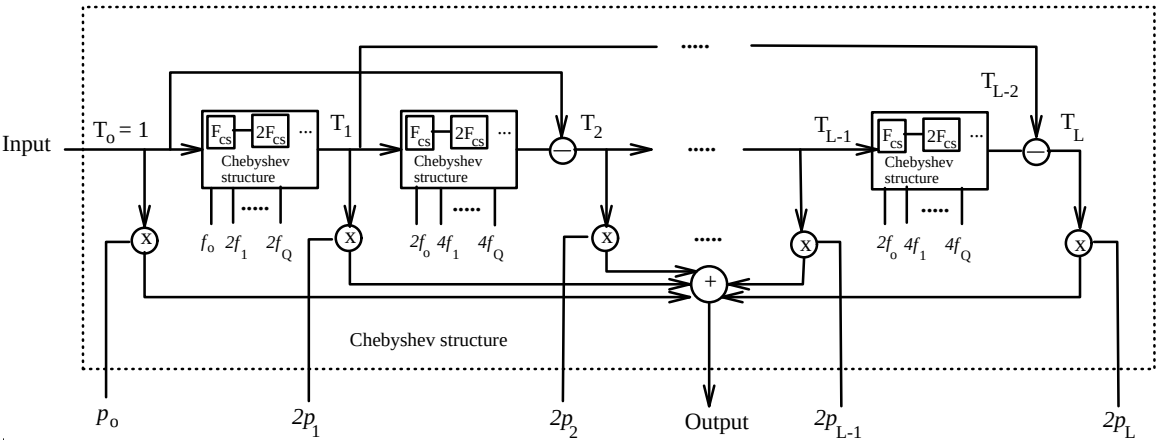


Figure 5.3. The two-in-one Chebyshev structure for realizing multi-dimensional filters.

are given by

$$f_q = \begin{cases} \sum_{r=0}^{R} t_{or} \cos(rk_c), & q = 0 \\ (\frac{1}{2}) \sum_{r=0}^{R} t_{qr} \cos(rk_c), & 1 \le |q| \le Q \end{cases} \tag{5.18}$$

The corresponding circularly-symmetric 3-D depth migration filter, with radial cutoff k_c, can be obtained by substituting for F in Fig. 5.2 the following 2-D transformation subfilter

$$F_2(k_1, k_2) = \sum_{q=0}^{Q} (\sum_{r=0}^{R} t_{qr} \cos(rk_c)) T_q[F_{\rm CS}(k_1, k_2)]. \tag{5.19}$$

$F_{\rm CS}(k_1, k_2)$ in (5.19) is a McClellan transformation which produces circularly-symmetric 2-D filters. Examples of transformations for circularly-symmetric filters were presented in [48, 49, 50, 66]. It is easy to see that $F_2(k_1, k_2)$ is obtained from $F(k_1)$ in (5.17) by means of the substitution

$$\cos k_1 = F_{\rm CS}(k_1, k_2). \tag{5.20}$$

So, the 2-D subfilter (5.19) can also be efficiently realized by means of the Chebyshev structure shown in Fig. 5.2, where F is replaced by $F_{\rm CS}$ and $\{p_n\}$ by $\{f_n\}_{n=0}^{Q}$ from (5.18). Consequently, the 2-D filters for 3-D depth migration are efficiently realized using a two-in-one Chebyshev structure, as shown in Fig. 5.3.

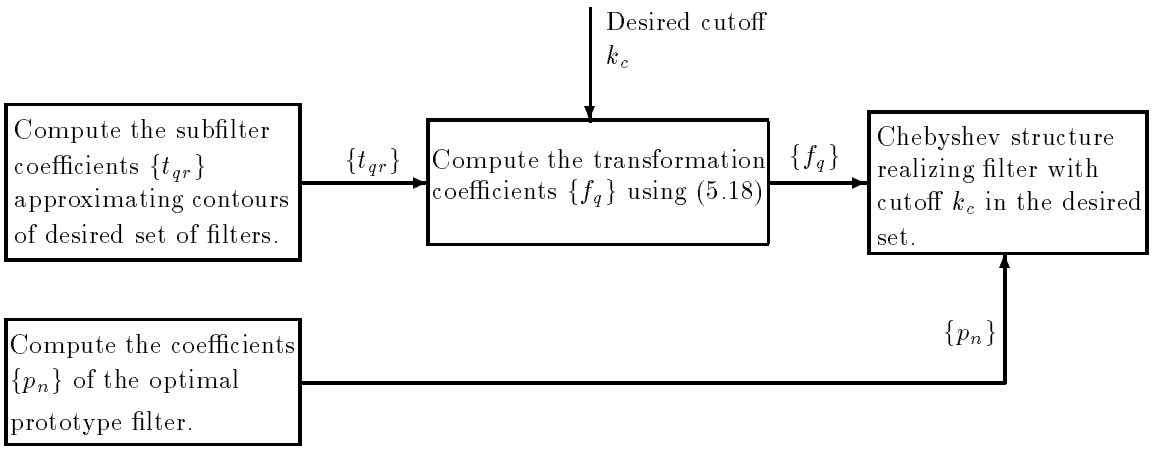


Figure 5.4. Block diagram of the design procedure.

From the above discussion, it can be concluded that the coefficients of the migration filters do not need to be computed and are directly realized by the Chebyshev structure in terms of the transformation subfilter coefficients $\{f_n\}$ (plus the coefficients of F_{CS} for the 3-D depth migration filters), and the prototype filter coefficients $\{p_n\}$. With a fixed b, the filter cutoff frequency k_c can be varied, for both the 2-D and 3-D depth migration filters, by varying only the coefficients $\{f_n\}_{n=0}^{Q}$ of the 1-D transformation subfilter. The coefficients $\{f_n\}$ can be easily computed using (5.18) in terms of the cutoff frequency k_c. On the other hand, migration filters, with the same cutoff frequency but different values of b, are obtained by varying only the coefficients $\{p_n\}$ of the prototype filter; $\{p_n\}$ are computed such that they best approximate (5.14). In this way, all the desired migration filters can be efficiently realized using the modular Chebyshev structure of Figs. 5.2 and 5.3, with the two input parameters b and k_c determining uniquely the 2-D or 3-D depth migration filter which is to be realized. The transformation coefficients $\{f_q\}$ of (5.18) depend on k_c, while the prototype filter coefficients $\{p_n\}$ depend on b. A MATLAB routine, which implements the Chebyshev structure and computes the coefficients of any migration filter with specified k_c and b, is listed in Appendix D. With an optimal prototype filter, the quality of the resulting migration filter depends mostly on how well the parameters $\{t_{qr}\}$ approximate the

ideal function (5.8) and, therefore, the hyperbolic contours (5.3). It should be noted that the subfilter coefficients $\{t_{qr}\}$ need to be computed once for the complete desired set. In addition, $\{t_{qr}\}$ do not depend on the parameter b. So, the same subfilter coefficients $\{t_{qr}\}$ can be used in designing different sets of filters corresponding to different values of b. A block diagram of the design procedure is shown in Fig. 5.1.3.

5.2 Families of Variable-Cutoff Linear-Phase Filters

It is often desirable to construct digital filters for which the cutoff frequency can be varied by controlling a small number of parameters. This section considers the design of variable-cutoff, linear-phase, lowpass filters. Variable-cutoff bandpass and highpass filters can be easily obtained from the lowpass filters [1].

One approach to implementing 1-D variable-cutoff, linear-phase, FIR lowpass filters was proposed by Oppenheim et al. [67], where the cutoff frequency of a prototype filter is controlled by varying all the parameters of a special first-order transformation function. In this section, the complete set of 1-D lowpass filters is treated as a single 2-D fan filter [65]. Using a procedure similar to the one presented in section 5.1, a single 2-D transformation function is used to design the complete desired family of lowpass filters. One additional transformation is needed to produce families of 2-D circularly-symmetric linear-phase filters. As a result, the individual 1-D and 2-D filters can be realized efficiently using Chebyshev structures. Alternatively, the complete family of 1-D (2-D) filters can be regarded and realized as a single 1-D (2-D) variable-cutoff filter.

Compared to the technique in [67], the cutoff frequency for the designed FIR filters is controlled through a *single* parameter when using first-order transformation functions. The new design procedure also supports higher order transformation functions, which allows the designer to control the error deviation and the width of the

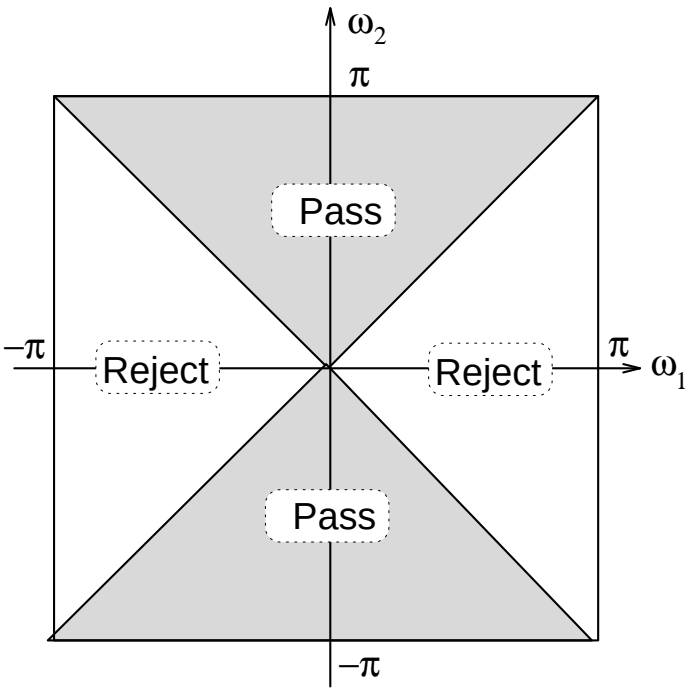


Figure 5.5. Frequency response of the ideal fan filter.

transition band by increasing the order of the transformation subfilter [64].

The entire family of 1-D zero-phase, lowpass filters can be approximated as the 2-D real-valued, fan filter having an ideal frequency response given by

$$D(\omega_1, \omega_2) = \begin{cases} 1, & |\omega_1| < |\omega_2| \\ 0, & \text{otherwise} \end{cases} \tag{5.21}$$

The ideal fan filter (5.21) has a wedge- or pie-shaped passband region as shown in Fig. 5.5. Each horizontal cross-section of $D(\omega_1, \omega_2)$ corresponds to a fixed value of ω_2 and represents uniquely the frequency response of a 1-D zero-phase lowpass filter with cutoff frequency ω_2. As in section 5.1, the transformation method is used to design the 2-D FIR filter $H(\omega_1, \omega_2)$ that approximates (5.21). Since $D(\omega_1, \omega_2)$ is quadrant symmetric, it is sufficient to consider only the first quadrant $0 \leq \omega_1, \omega_2 \leq \pi$. The prototype filter used is of the form (5.4) with the variable k replaced by ω, and the

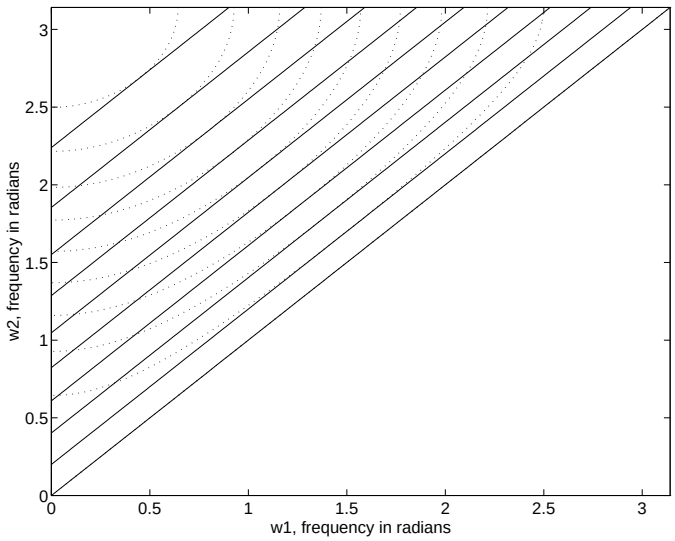


Figure 5.6. Ideal subfilter contours (solid lines) and McClellan's first-order transformation for fan filters (dotted curves).

transformation is done by means of the substitution

$$\cos(\omega) = F(\omega_1, \omega_2) \tag{5.22}$$

where $F(\omega_1, \omega_2)$ is the transformation subfilter.

5.2.1 Part 1: Subfilter Design

McClellan [3, 65] showed that an ideal 1-D zero-phase, lowpass filter is mapped into an ideal fan filter by using the first-order transformation

$$F(\omega_1, \omega_2) = 0.5(\cos\omega_1 - \cos\omega_2). \tag{5.23}$$

Note that the transformation (5.23) maps the prototype filter $P(\omega)$ to the line

$$\omega_2 = \pi - \omega_1. \tag{5.24}$$

So, ideally, a subfilter with contours shown by the solid lines in Fig. 5.6 is needed in order for the transformed fan filter to have the same cutoff band as the prototype

filter $P(\omega)$ [64]. The contours of (5.23) are shown by the dotted lines in Fig. 5.6 and indicates that the resulting fan filter has a distorted and broader transition band than that of the original 1-D filter.

It is easy to verify that the following ideal subfilter response realizes the ideal contour characteristics of Fig. 5.6 [64]:

$$F_D(\omega_1, \omega_2) = \sin(\frac{|\omega_2| - |\omega_1|}{2}), \qquad -\pi \leq \omega_1, \omega_2 \leq \pi. \tag{5.25}$$

Since $D(\omega_1, \omega_2)$ is quadrant symmetric, the approximating filter $H(\omega_1, \omega_2)$ and, therefore, the transformation subfilter must be the frequency responses of quadrant symmetric 2-D FIR filters. So, $F(\omega_1, \omega_2)$ is of the form

$$F(\omega_1, \omega_2) = \sum_{q=0}^{Q} \sum_{r=0}^{R} t_{qr} \cos(q\omega_1) \cos(r\omega_2) \tag{5.26}$$

where the $(Q+1) \times (R+1)$ coefficients $\{t_{qr}\}$ are to be determined such that $F(\omega_1, \omega_2)$ best approximates $F_D(\omega_1, \omega_2)$ in the least square or Chebyshev sense (see section 5.1). In this way, a subfilter with any order could be used and narrower transition bands could result [64]. However, the subfilter order should be kept small in order to reduce the complexity of the filter implementation. Also, note that the approximation need only be done in the first quadrant $0 \leq \omega_1, \omega_2 \leq \pi$.

5.2.2 Part 2: Prototype Filter Design

With the ideal subfilter (5.25), the required ideal prototype filter is to be determined. First, note that for $0 \leq \omega_1 \leq \pi$

$$F_D(\omega_1, \pi - \omega_1) = \cos \omega_1. \tag{5.27}$$

So, the prototype filter is mapped to the line (5.24), and it follows that

$$H(\omega_1, \pi - \omega_1) = P(\omega_1) \approx D(\omega_1, \pi - \omega_1). \tag{5.28}$$

From (5.28), it can be deduced that $P(\omega)$ should approximate the ideal frequency response

$$P_D(\omega) = \begin{cases} 1, & |\omega_1| < \dfrac{\pi}{2} \\ 0, & \text{otherwise} \end{cases} \tag{5.29}$$

The exchange algorithm of Chapter 3 is again used to compute the coefficients of the optimal prototype filter $P(\omega)$.

5.2.3 Part 3: Realizing the variable-cutoff lowpass filters

After finding the optimal mapping coefficients $\{t_{qr}\}$ and the optimal prototype coefficients $\{p_n\}$ as described in sections 5.2.1 and 5.2.2, the lowpass FIR filter $H(\omega)$ with cutoff frequency ω_c is obtained by substituting for F in Fig. 5.2 the 1-D transformation subfilter

$$F(\omega) = F(\omega, \omega_c) = \sum_{q=0}^{Q} (\sum_{r=0}^{R} t_{qr} \cos{(r\omega_c)}) \cos((q\omega)) \tag{5.30}$$

and by substituting for $\{p_n\}$ the coefficients of the 1-D prototype filter approximating (5.29). From (5.30), it follows that the coefficients of the 1-D subfilter $F(\omega)$ are given by

$$f_q = \begin{cases} \displaystyle\sum_{r=0}^{R} t_{or} \cos{(r\omega_c)}, & q = 0 \\ (\dfrac{1}{2}) \displaystyle\sum_{r=0}^{R} t_{qr} \cos{(r\omega_c)}, & 1 \leq |q| \leq Q \end{cases} \tag{5.31}$$

The corresponding 2-D circularly-symmetric lowpass filter with radial cutoff ω_c, can be obtained by replacing $\cos{(q\omega)}$ in (5.30) by $T_q[F_{\text{CS}}(\omega_1, \omega_2)]$, where $F_{\text{CS}}(\omega_1, \omega_2)$ is a transformation for circularly symmetric filters as described in section 5.1. The two-in-one Chebyshev structure of Fig. 5.3 can be used to implement these variable-cutoff 2-D filters.

The cutoff frequency ω_c is varied, for both the 1-D and 2-D filters, by varying the 1-D subfilter coefficients $\{f_n\}_{n=0}^{Q}$ given by (5.31). Now, if $R = 1$ (Q arbitrary),

(5.31) reduces to

$$f_q = \begin{cases} t_{00} + t_{01}\alpha, & q = 0 \\ 0.5(t_{q0} + t_{q1}\alpha), & 1 \leq |q| \leq Q \end{cases} \tag{5.32}$$

where $\alpha = \cos\omega_c$. So, in that case, the cutoff frequency is controlled through the single parameter α.

A variable-cutoff highpass filter can be directly realized using the same Chebyshev structure as a variable-cutoff lowpass filter except that the transformation coefficients $\{f_q\}$ of (5.31) are replaced by $\{(-1)^q f_q\}$. On the other hand, a variable-cutoff bandpass filter can be realized by cascading one lowpass and one highpass variable-cutoff filter structures as shown in Fig. 5.7. The highpass Chebyshev structure is obtained from a lowpass structure as explained above and controls the low cutoff frequency of the bandpass filter, while the lowpass Chebyshev structure controls the high cutoff frequency.

5.3 Constraints on the Transformation Subfilter

In this section, constraints, which result in a better approximation of the desired 2-D family, are imposed on the transformation subfilter. As discussed earlier in sections 5.1 and 5.2, the desired transformation subfilters (5.8) and (5.25) map exactly the one-dimensional prototype filters to the line

$$\omega_2 = \pi - \omega_1 \tag{5.33}$$

in the two-dimensional frequency domain (ω_1, ω_2). Consequently, the prototype filter is chosen to approximate the ideal two-dimensional filter, representing the desired filter family, along that line (5.33). Also, note that the ideal two-dimensional family filters (5.2) and (5.21) are "fan-shaped" filters with a cutoff edge along the line

$$\omega_2 = \pm\omega_1. \tag{5.34}$$

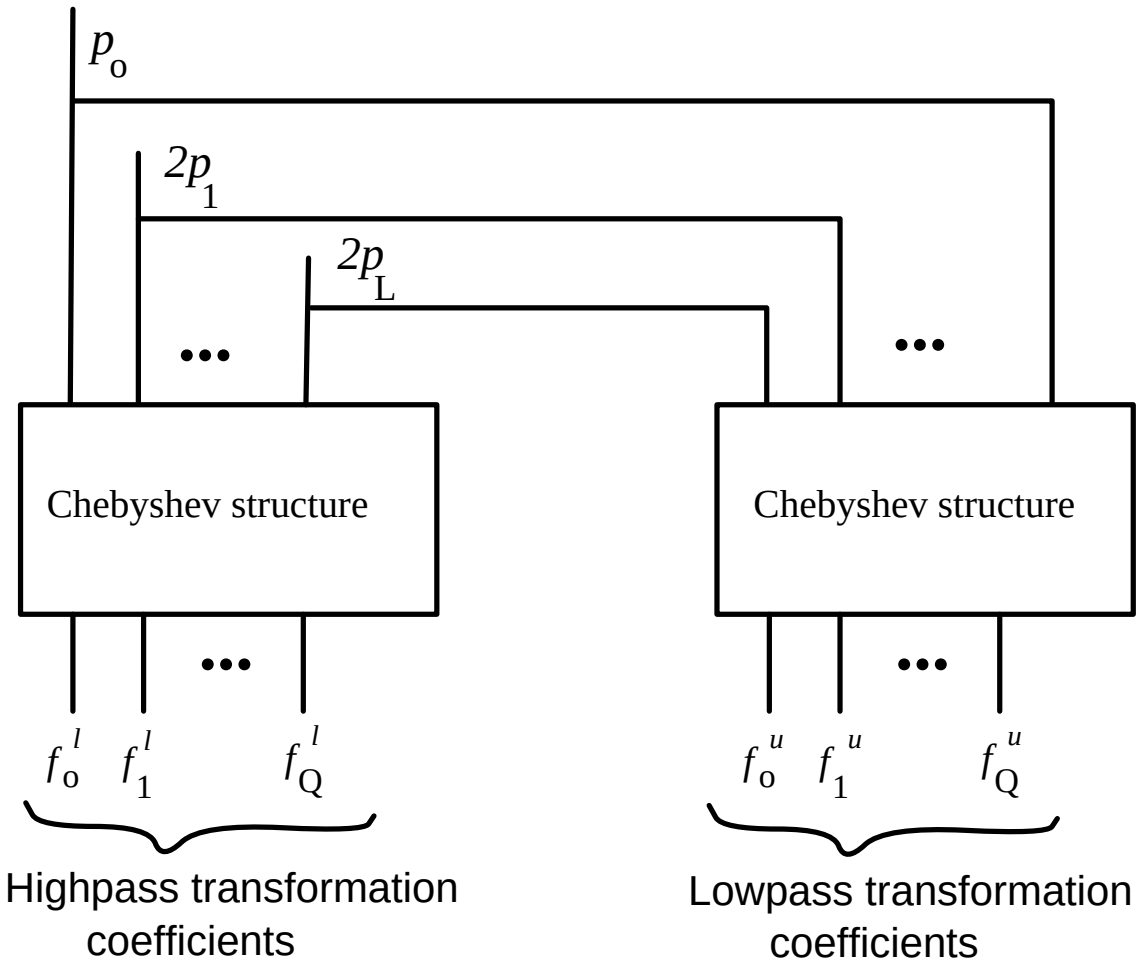


Figure 5.7. Realization of a variable-cutoff bandpass filter by means of the Chebyshev structures for the variable-cutoff lowpass filter (see Figs. 5.2 and 5.3). The lower and upper cutoff frequencies of the realized bandpass filter are controlled by $\{f_n^l\}_{n=0}^Q$ and $\{f_n^u\}_{n=0}^Q$, respectively. The highpass transformation coefficients $\{f_n^l\}_{n=0}^Q$ are obtained by toggling the signs of the lowpass transformation coefficients which are computed using (5.31) with ω_c set to be the desired lower cutoff frequency of the bandpass filter.

As a result, it is desirable to have the following two constraints imposed on the designed two-dimensional subfilter approximating (5.8) or (5.25):

$$\omega \quad \longleftrightarrow \quad (\omega, \pi - \omega) \tag{5.35}$$

$$\pi/2 \quad \longleftrightarrow \quad (\omega, \omega). \tag{5.36}$$

Constraint (5.35) guarantees that the prototype filter is mapped onto the line (5.33) while constraint (5.36) maps $\pi/2$, the cutoff frequency of the prototype filters, to the cutoff edge $\omega_1 = \omega_2$. Also, these constraints reduce the size of the approximation problem by decreasing the number of free coefficients $\{t_{qr}\}$ and, thus, result in some computational savings. However, for each given Q and R (transformation subfilter orders), one needs to derive relations between the coefficients $\{t_{qr}\}$ such that the constraints (5.35–5.36) are satisfied and explicitly determine which of these coefficients are free parameters. This derivation becomes tedious for a subfilter order (Q or R) larger than two. Therefore, it is desirable to have a general formula that would automatically produce the quadrant-symmetric subfilter of order $Q \times R$ satisfying (5.35–5.36). This formula is derived next.

Recall that the general form of a quadrant-symmetric subfilter $F(\omega_1, \omega_2)$ of order $(Q \times R)$ is given by

$$F(\omega_1, \omega_2) = \sum_{q=0}^{Q} \sum_{r=0}^{R} t_{qr} \cos(q\omega_1) \cos(r\omega_2). \tag{5.37}$$

It can be easily verified that $Q \geq 1$ and $R \geq 1$. A lower order subfilter cannot satisfy both constraints (5.35) and (5.36).

Applying constraint (5.35) leads to

$$\sum_{q=0}^{Q} \sum_{r=0}^{R} t_{qr} \cos(q\omega) \cos(r\pi - r\omega) = \cos\omega \tag{5.38}$$

which, in turn, generates the following constraint equations

$$t_{10} = 1 + t_{01} \tag{5.39}$$

$$t_{jj} = 0, \qquad 0 \leq j \leq \min(Q, R) \tag{5.40}$$

$$t_{qr} = t_{rq}\,, \qquad (q+r) \text{ is odd} \tag{5.41}$$

$$t_{qr} = -t_{rq}\,, \qquad (q+r) \text{ is even} \tag{5.42}$$

Constraint (5.36) leads to

$$\sum_{q=0}^{Q}\sum_{r=0}^{R} t_{qr}\cos(q\omega)\cos(r\omega) = 0 \tag{5.43}$$

which implies that

$$t_{qr} = -t_{rq} \tag{5.44}$$

where $\{t_{rq}\}_{r>Q} = \{t_{rq}\}_{q>R} = 0$.

So, applying constraints (5.35) and (5.36) together gives the following constraint equations

$$t_{10} = \frac{1}{2} \tag{5.45}$$

$$t_{01} = -\frac{1}{2} \tag{5.46}$$

$$t_{qr} = 0\,, \qquad (q+r) \text{ is odd} \tag{5.47}$$

$$t_{qr} = -t_{rq}\,, \qquad (q+r) \text{ is even} \tag{5.48}$$

The following theorem has, therefore, been proved.

Theorem 5.1 The quadrant-symmetric subfilter of order $Q \times R$ satisfying the constraints (5.35) and (5.36) is given by

$$F(\omega_1,\omega_2) = \sum_{q=0}^{Q}\sum_{r=0}^{R} t_{qr}\cos(q\omega_1)\cos(r\omega_2) \tag{5.49}$$

where

$$\begin{cases} t_{10} = \dfrac{1}{2} \\ t_{01} = -\dfrac{1}{2} \\ t_{qr} = 0\,, & (q+r) \text{ is odd} \\ t_{qr} = -t_{rq}\,, & (q+r) \text{ is even} \end{cases} \tag{5.50}$$

□

It should be noted that the (nonzero) coefficients $\{t_{qr}, (q+r)$ even, $0 \leq q \leq Q, 0 \leq r < q\}$, represent the free parameters that need to be computed such that (5.49) best approximates the ideal subfilter $F_D(\omega_1, \omega_2)$. Note also that some of these coefficients may be zero when $Q \neq R$ since $\{t_{rq}\}_{r>Q} = \{t_{rq}\}_{q>R} = 0$. By using Theorem 5.1, the problem becomes easily scalable.

The subfilter design problem can now be restated as follows. Find the coefficients

$$\{t_{qr},\ (q+r)\ \text{ even},\ 0 \leq q \leq Q,\ 0 \leq r < q\} \tag{5.51}$$

that minimize, in the least square or minimax sense, the weighted error

$$E(\omega_1, \omega_2) = W(\omega_1, \omega_2)[F'_D(\omega_1, \omega_2) - F'(\omega_1, \omega_2)] \tag{5.52}$$

where

$$F'_D(\omega_1, \omega_2) = F_D(\omega_1, \omega_2) - 0.5(\cos\omega_1 - \cos\omega_2) \tag{5.53}$$

$$F'(\omega_1, \omega_2) = F(\omega_1, \omega_2) - 0.5(\cos\omega_1 - \cos\omega_2) \tag{5.54}$$

$F_D(\omega_1, \omega_2)$ in (5.53) is the desired subfilter and $F(\omega_1, \omega_2)$ in (5.54) is the approximating subfilter (5.49).

5.4 Design Examples

Before presenting the design examples, some observations about the designed filters follow. For the current discussion, the migration filter families are used for illustration but all the observations apply to the variable-cutoff, linear-phase, filters of section 5.2.

Figure 5.8 shows the ideal hyperbolic contours of the migration filter families with a horizontal cross-section at $k_2 = 1.4$ represented by a dash-dotted line. As mentioned in section 5.1, the filter with a cutoff frequency k_c is given by the horizontal cross-section $k_2 = k_c$. The contours can be thought of as "propagating", in the two-dimensional frequency domain, the optimal prototype filter values which are exactly mapped into the line (5.12) corresponding to the dotted line in Fig. 5.8. For

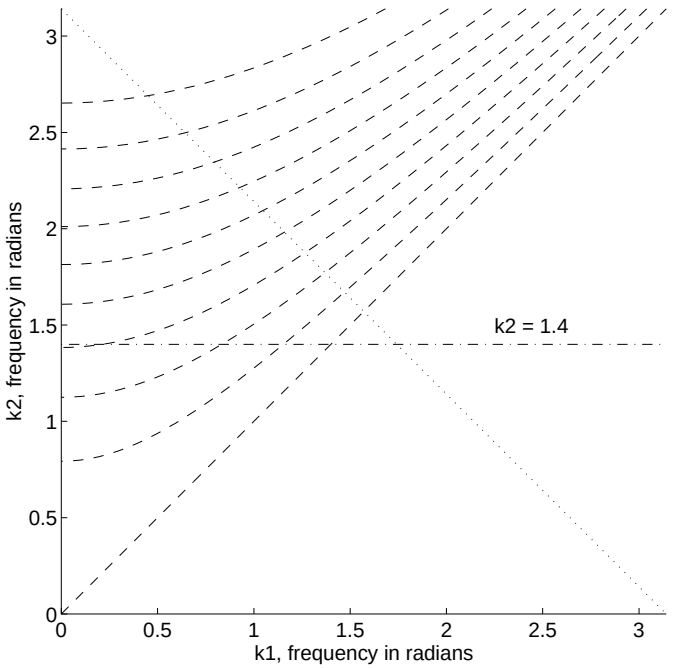


Figure 5.8. Ideal hyperbolic contours of the migration filter family (solid curves) with the horizontal cross-section (dash-dotted line) representing the migration filter having a cutoff frequency $k_c = 1.4$. The dotted line represents the line into which the prototype filter is mapped.

convenience, the horizontal cross-section and the line (5.12) will be called the "filter line" and the "prototype line", respectively. For a given filter with cutoff frequency k_c, the best possible approximating filter results if the designed contours match the ideal contour segments joining the filter line ($k_2 = k_c$) to the prototype line. These segments control the "flow" of values from the prototype line to the corresponding filter line. Consequently, for a given subfilter order, better approximations to a subset of filters can be achieved by assigning a higher weight to selected contour segments. In the remainder of this chapter, the following terminology is used:

- Prototype line: the line $k_2 = \pi - k_1$.
- Filter line: any horizontal cross-section representing a filter in the family.
- Flow segment: a contour segment joining a filter line to the prototype line; one

end-point of the segment is on the filter line and the other end-point belongs to the prototype line.

In the following design examples, the frequency values in the plots are normalized by a factor of 2π.

5.4.1 Families of migration filters

As discussed in section 5.1, the same transformation subfilter (5.7) can be used to generate the different families of migration filters. The subfilter needs to be designed only once so that it best approximates the hyperbolic contours of the migration filter families. Which family is generated depends on the value of b and, therefore, on the prototype filter. The design of the prototype filter is easily done using the algorithm presented in Chapter 3.

For illustration purposes, a continuous family of migration filters with $b = 1.0$ (a common value used in seismic processing) is generated by designing a prototype filter with the following specifications

$$P_D(k) = \begin{cases} e^{j\sqrt{\pi^2 - 2\pi k}}, & |k| < \dfrac{\pi}{2} \\ 0, & \text{otherwise} \end{cases} \tag{5.55}$$

$$\text{Weight: } W(k) = \begin{cases} 50, & |k| < \dfrac{\pi}{2} \\ 1, & \text{otherwise} \end{cases} \tag{5.56}$$

Approximating filter length: $N = 29$

The designed prototype filter is then mapped by the subfilter (5.7) to produce an approximation of the desired family. The characteristics of the optimal prototype filter are shown in Fig. 5.9. The multiple-exchange ascent algorithm converged to the optimal solution in 6 iterations of the complex Remez stage with $\|E_{opt}\| = 0.9153$. The design of the subfilter is discussed next.

Since it is desirable to keep the order of the subfilter as small as possible, a first order transformation subfilter is considered first. The order is then increased if a

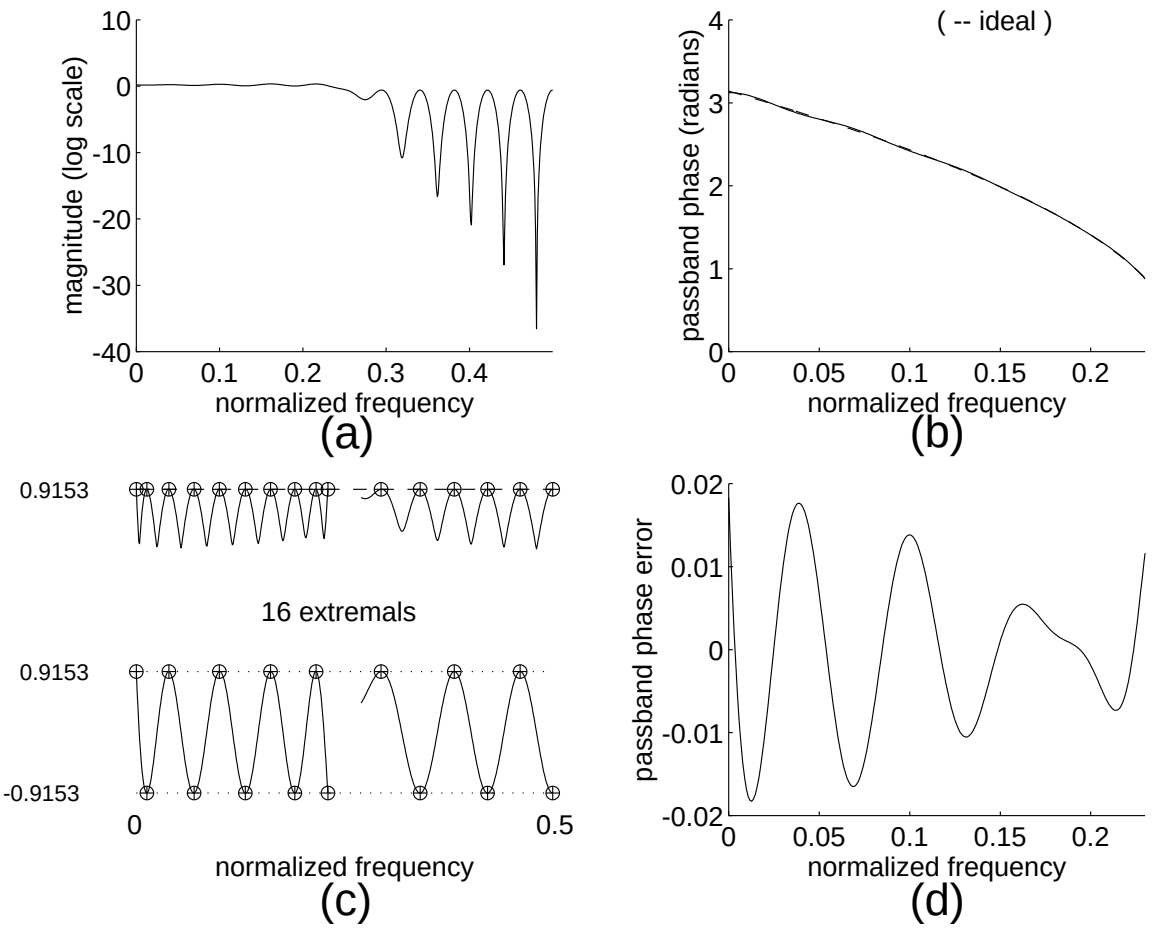


Figure 5.9. Migration Filter Family Prototype, $N = 29$. (a) Magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Magnitude of the error (top), and alternating real rotated error (bottom). (d) Phase error in passband.

better approximation is needed. Using the results of section 5.3, it can be easily seen that the needed first order ($Q = R = 1$) subfilter $F(k_1, k_2)$ is given by:

$$F(k_1, k_2) = 0.5(\cos k_1 - \cos k_2) \tag{5.57}$$

which is McClellan's transformation for the design of real-valued fan filters.

Figure 5.10 compares the contours of the first order subfilter (solid curves) to the ideal subfilter contours (dashed curves). It is clear that the first order subfilter does not approximate well the desired hyperbolic contours nor flow segments of these contours. Some of the resulting 1-D filters are shown in Figs. 5.11–5.12, for $k_c = 0.6\pi$ and 0.8π, respectively. The coefficients of these filters were computed using the coefficients of the subfilter and of the prototype filter approximating (5.55) by means of the Chebyshev structure shown in Fig. 5.2. It should be noted that no free parameters are available in this case to improve the resulting contour approximation. So, a higher order transformation is needed.

The second order subfilter is of the form

$$F(k_1, k_2) = 0.5(\cos k_1 - \cos k_2) + t_{20}(\cos(2k_1) - \cos(2k_2)). \tag{5.58}$$

With a unity weighting function (i.e., no weighting), the minimax error criterion resulted in $t_{20} = -0.1251$. A very close result, $t_{20} = -0.1230$, is obtained with the least square error criterion. The respective contours are shown in Fig. 5.13 and are obviously very similar. From the contour plots and the flow segments, four cases are distinguished:

- $0 < k_c \leq 0.45\pi$: flow segments do not match the ideal contours. The resulting filters in this set are not good approximations to the ideal migration filters.

- $0.45\pi < k_c \leq 0.67\pi$: flow segments match closely the ideal contours in the passband region near the cutoff edge. The resulting filters of this set approximate well the ideal migration filters on the high passband frequencies.

- $0.67\pi < k_c \leq 0.85\pi$: flow segments match the ideal contours (except near $k_2 = 0$). The designed filters of this set approximate well the ideal migration filters in most of the passband.

- $0.85\pi < k_c \leq \pi$: flow segments match the ideal contours in the passband region near $k_2 = 0$. The resulting filters approximate well the ideal migration filters on the low passband frequencies.

In order to illustrate the above observations, Figs. 5.14(a)–(d) show the properties of the filters with $k_c = 0.4\pi$, 0.6π, 0.8π and 0.95π, respectively. It follows that the obtained second order subfilter provides a good approximation for the continuous set of migration filters with $0.67\pi < k_c \leq 0.85\pi$. The phase responses of filters from this set are shown in Fig. 5.15. The characteristics of the resulting 3-D depth migration filter with cutoff 0.8π are shown in Fig. 5.16. The coefficients of this two-dimensional filter were computed, by means of the two-in-one Chebyshev structure (Fig. 5.3), using the prototype filter approximating (5.55), the second order subfilter (5.58), and the first order McClellan transformation

$$F_{\rm CS}(\omega_1, \omega_2) = 0.5(\cos\omega_1 + \cos\omega_2 + \cos\omega_1 \cos\omega_2 - 1). \tag{5.59}$$

A good approximation for other sets of filters can be obtained with the second order subfilter (5.58) by applying a suitable weighting function in the region of interest. Using the following weighting function

$$W(k_1, k_2) = \begin{cases} 50, & k_2 > k_1 \text{ and } k_2 < 1.0 \\ 1, & \text{otherwise} \end{cases} \tag{5.60}$$

results in $t_{20} = -0.0565$ for the minimax error norm and $t_{20} = -0.0508$ for the least square error norm. The contours generated using the different error norms differ slightly and are shown in Fig. 5.17. The weighting (5.60) produces a good approximation of the set of migration filters with $0.3\pi \leq k_c \leq 0.75\pi$. Using the minimax result $t_{20} = -0.0565$, the properties of designed filters from this set are

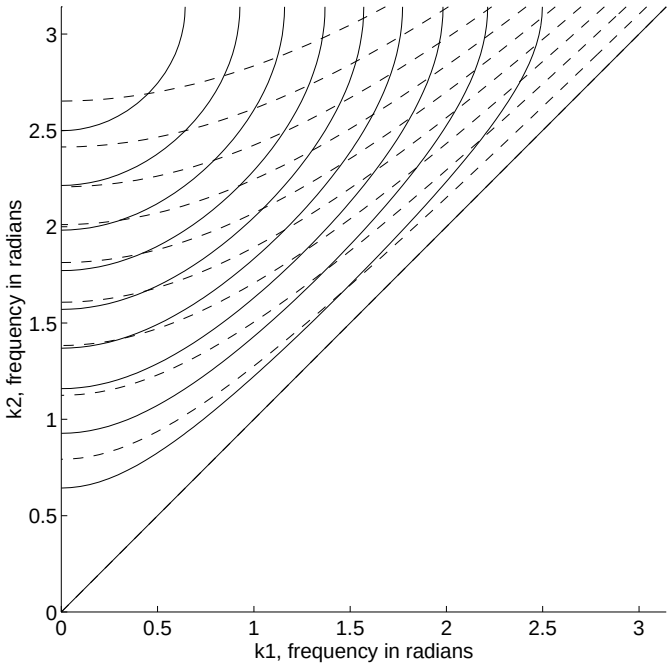
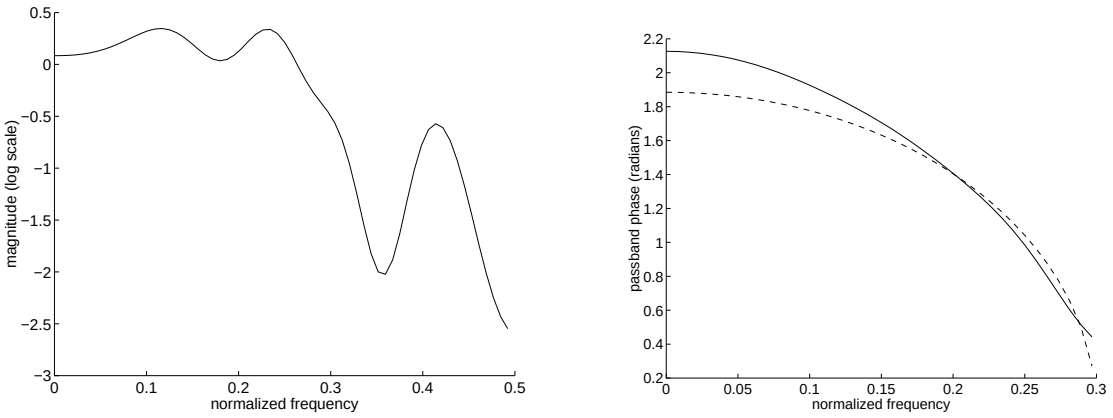


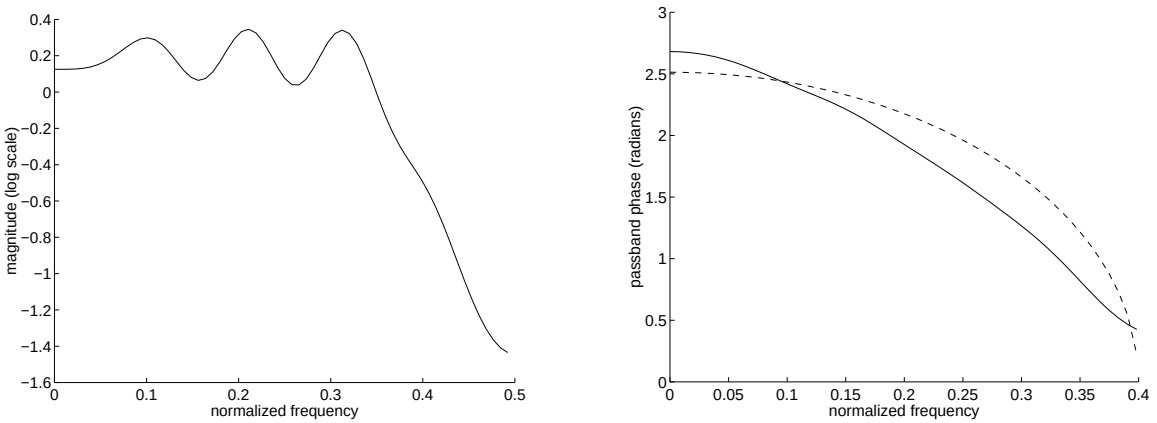
Figure 5.10. Comparison of the first-order subfilter contours (solid curves) with the contours of the ideal migration subfilter (5.8)(dashed curves).

shown in Figs. 5.18 and 5.19. Similarly, a large weighting can be applied in the passband regions corresponding to filters with high cutoff frequencies to get better approximation of these filters. Higher order subfilters could also be used to improve the approximation of the ideal contours and, therefore, of the desired set of filters.



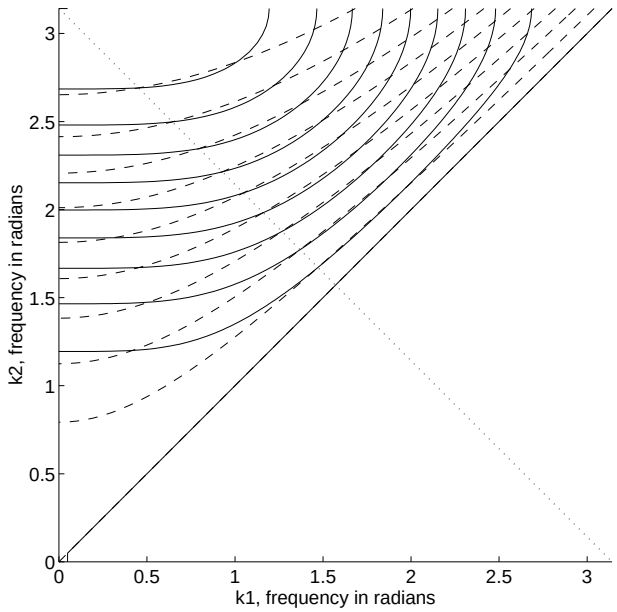
(a) (b)

Figure 5.11. Migration filter obtained using a first-order subfilter, $k_c = 0.6\pi$. (a) Magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses.

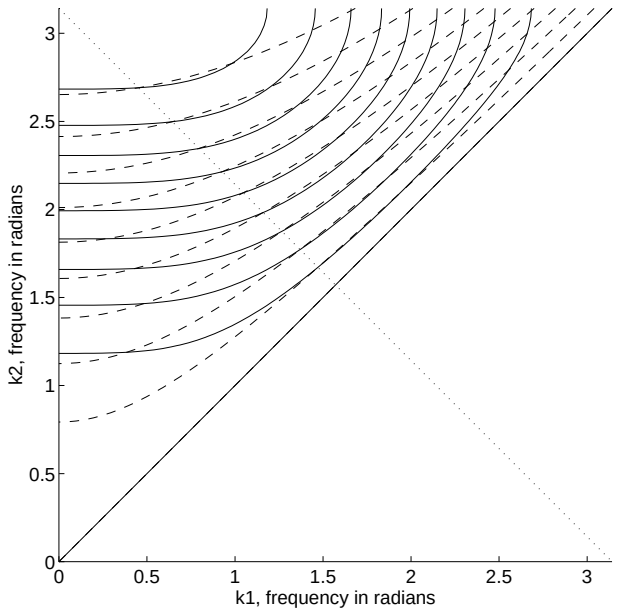


(a) (b)

Figure 5.12. Migration filter obtained using a first-order subfilter, $k_c = 0.8\pi$. (a) Magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses.

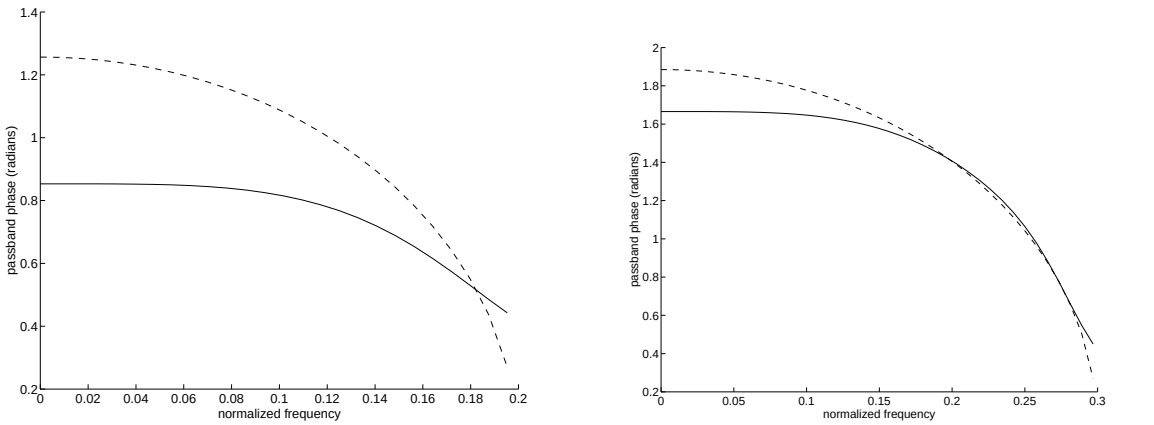


(a)



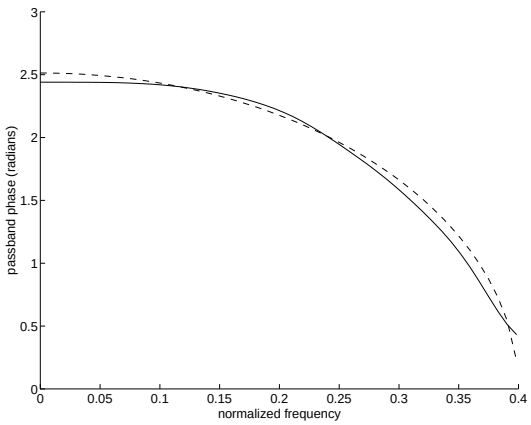
(b)

Figure 5.13. Contours of the second-order subfilter (solid curves) and of the ideal subfilter (dashed curves). (a) Contours resulting from the minimax error criterion. (b) Contours resulting from the least square error criterion.

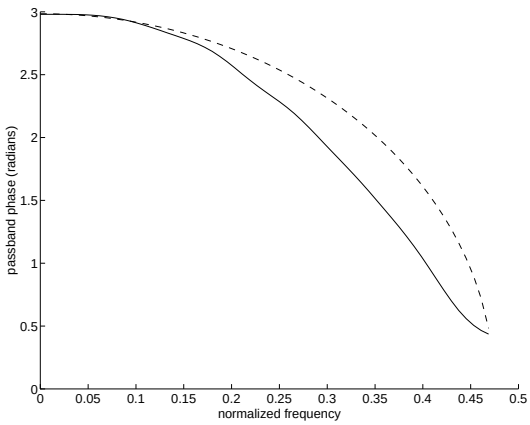


(a) $k_c = 0.4\pi$.

(b) $k_c = 0.6\pi$.



(c) $k_c = 0.8\pi$.



(d) $k_c = 0.95\pi$.

Figure 5.14. Passband phase responses of the desired filters (dashed curves) and of the FIR filters (solid curves) obtained by using a non-weighted second-order subfilter. Four cases are distinguished: (a) $0 < k_c \leq 0.45\pi$: obtained contours do not match ideal. (b) $0.45\pi < k_c \leq 0.67\pi$: obtained contours match ideal ones in passband region near the cutoff edge. (c) $0.67\pi < k_c \leq 0.85\pi$: obtained contours match ideal ones in most passband region. (d) $0.85\pi < k_c \leq \pi$: obtained contours match ideal ones near the line $k_2 = 0$.

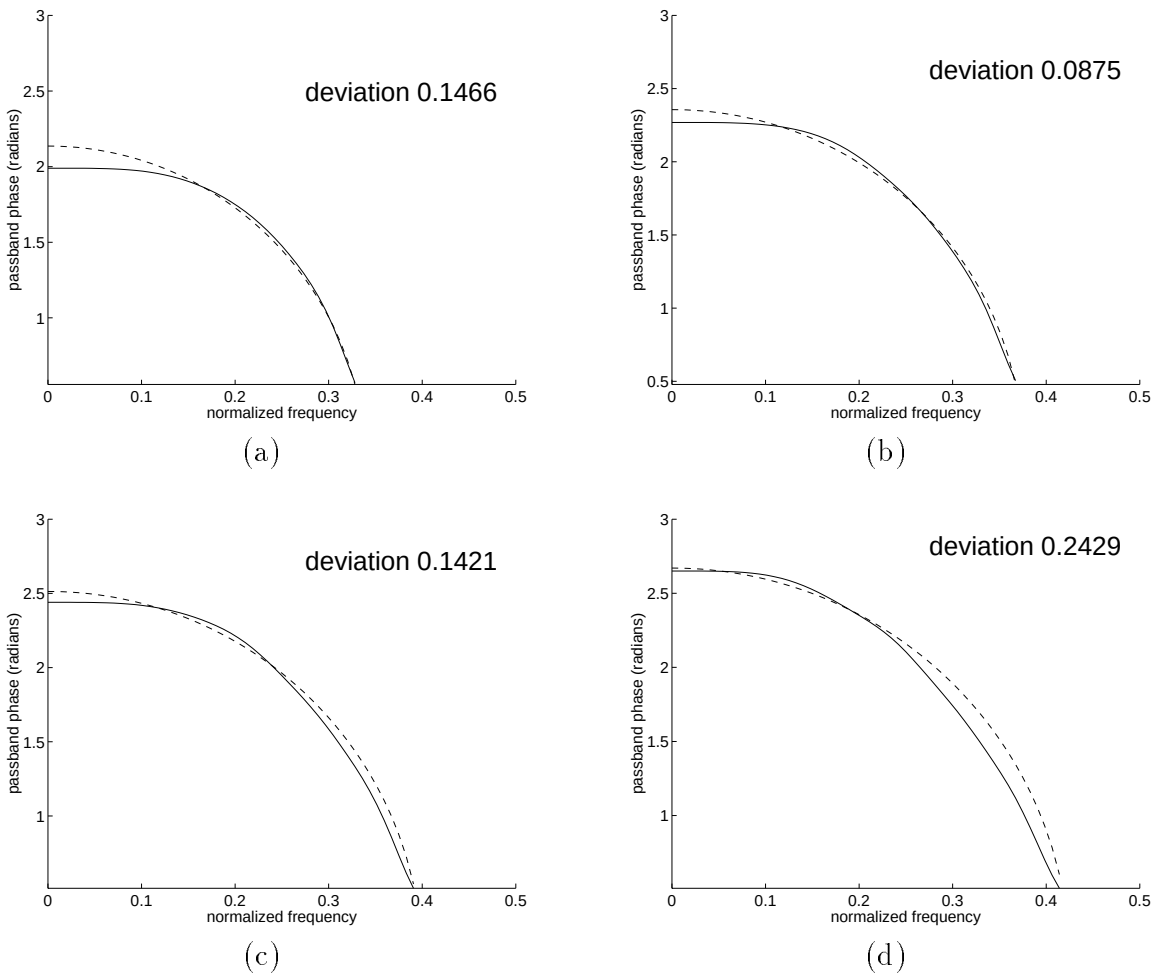
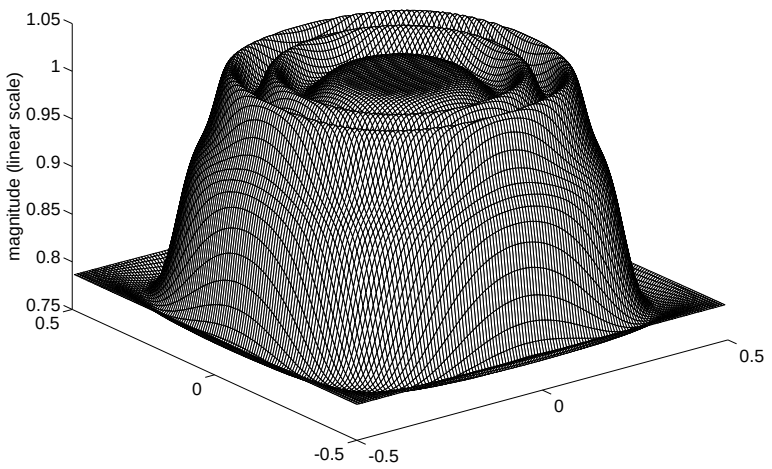
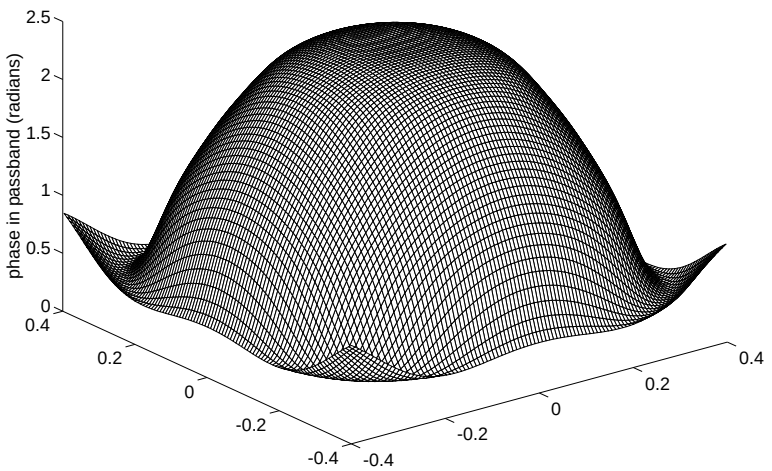


Figure 5.15. The non-weighted second-order subfilter provides a good approximation of the continuous set of filters with $0.68\pi \leq k_c \leq 0.85\pi$. FIR filter (solid) and desired filter (dashed) passband phase responses: (a) $k_c = 0.68\pi$. (b) $k_c = 0.75\pi$. (c) $k_c = 0.8\pi$. (d) $k_c = 0.85\pi$.

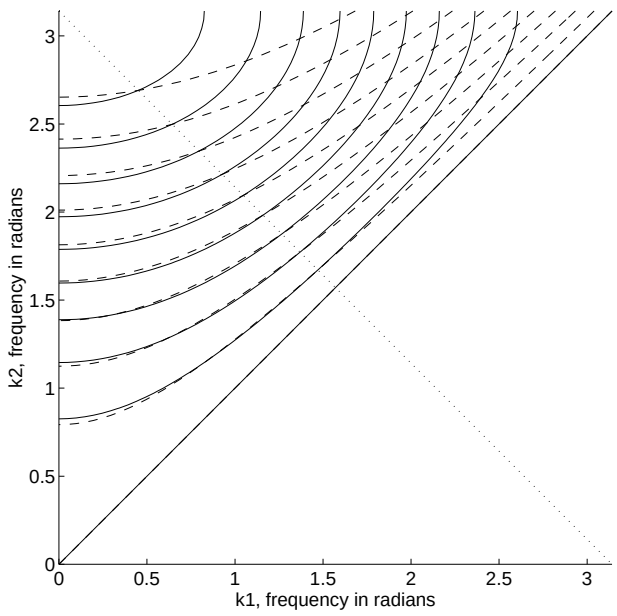


(a)

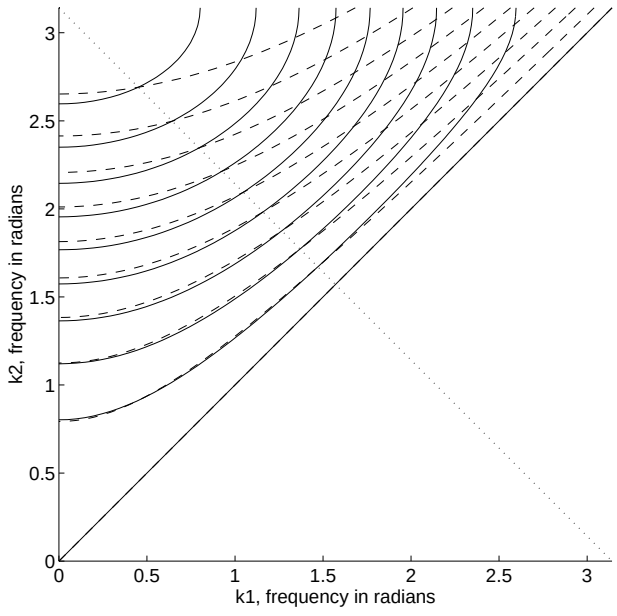


(b)

Figure 5.16. 3-D depth migration filter obtained using a non-weighted second-order subfilter, $k_c = 0.8\pi$. (a) Magnitude response. (b) Phase response in the passband.



(a)



(b)

Figure 5.17. Contours of the weighted second-order subfilter (solid curves) and of the ideal subfilter (dashed curves). (a) Contours resulting from the weighted minimax error criterion. (b) Contours resulting from the weighted least square error criterion.

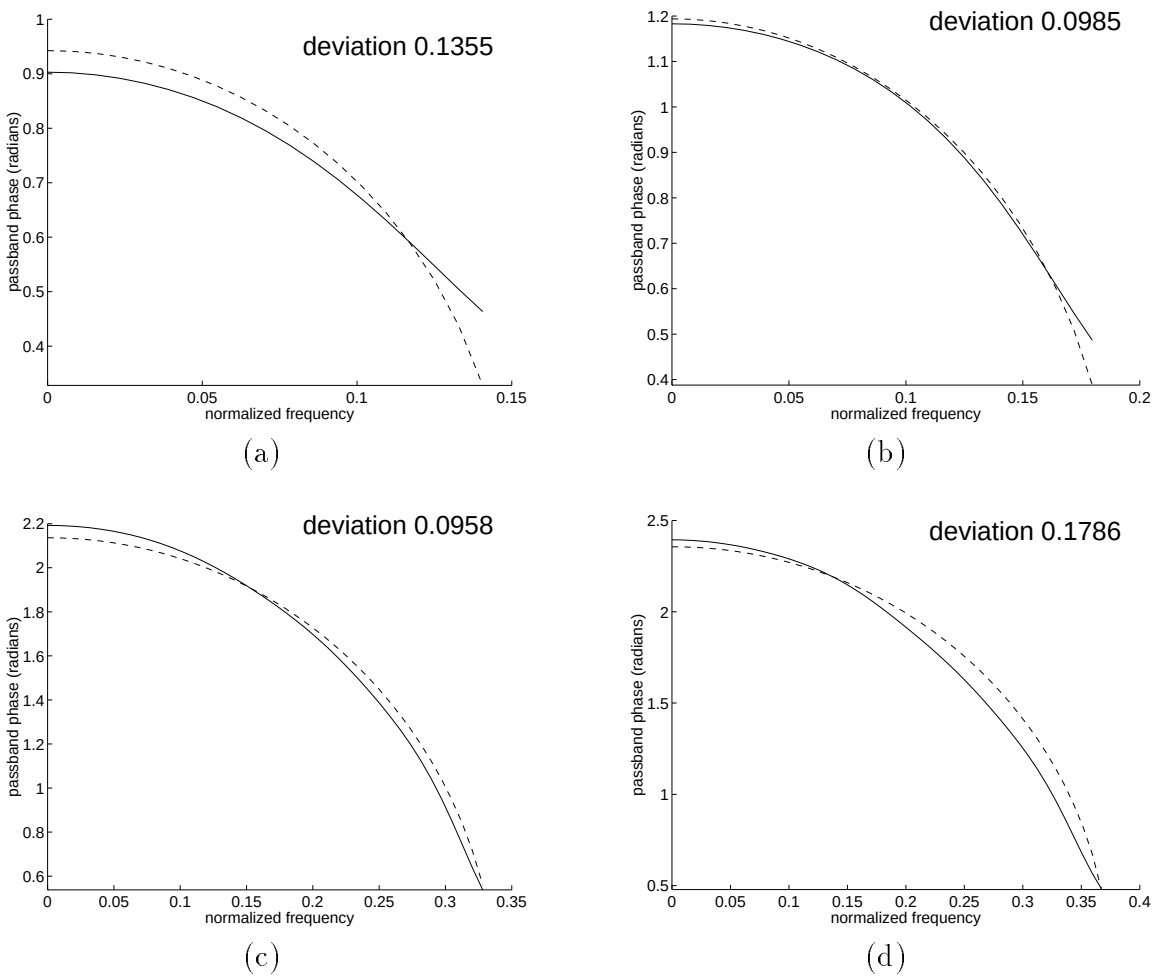


Figure 5.18. A suitably weighted second-order subfilter providing a good approximation of the continuous set of filters with $0.3\pi \leq k_c \leq 0.75\pi$. FIR filter (solid) and desired filter (dashed) passband phase responses: (a) $k_c = 0.3\pi$. (b) $k_c = 0.38\pi$. (c) $k_c = 0.68\pi$. (d) $k_c = 0.75\pi$.

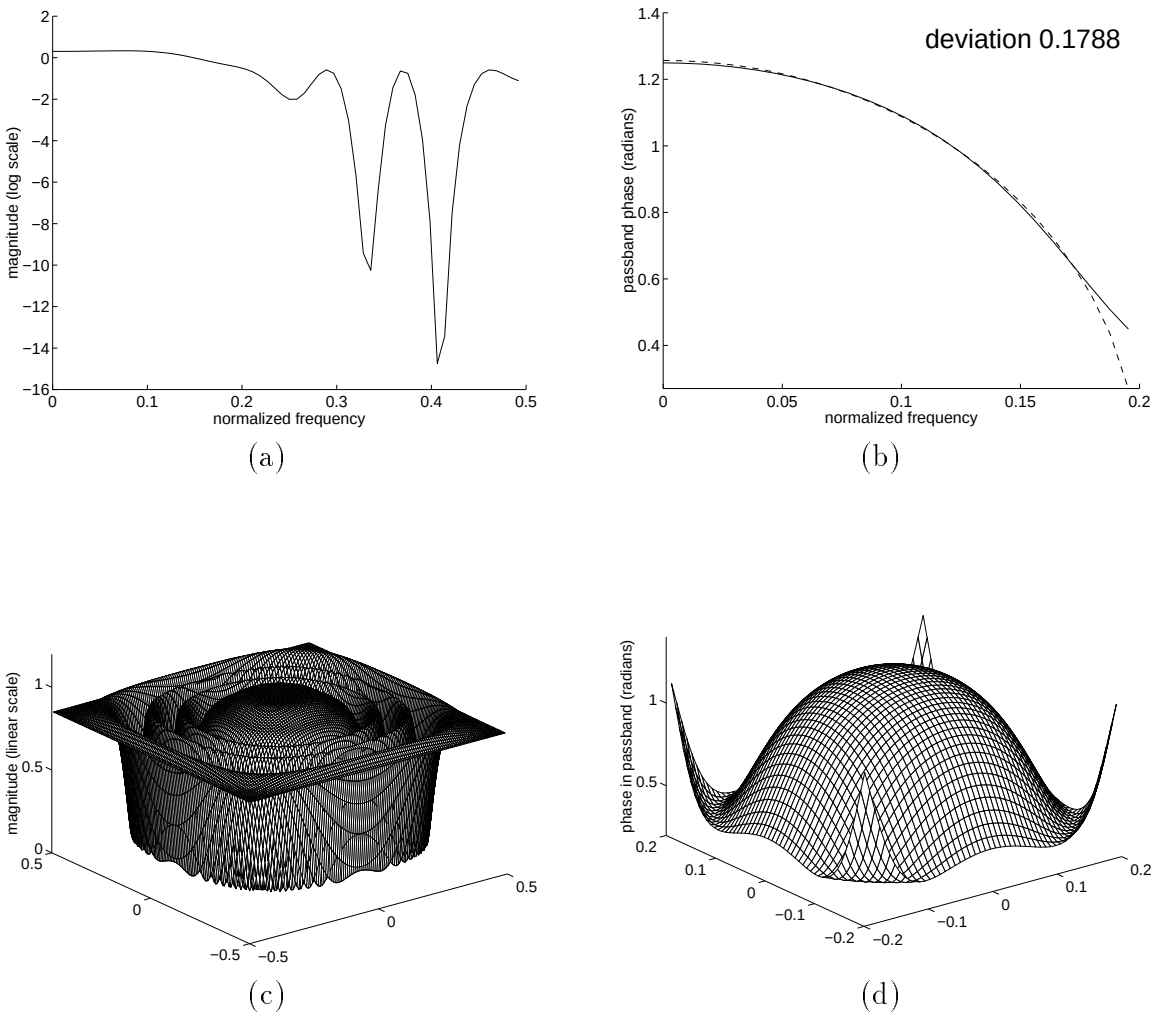


Figure 5.19. One- and two-dimensional migration filters obtained using a weighted second-order subfilter, $k_c = 0.4\pi$. (a) Magnitude response of the 1-D migration filter. (b) Designed (solid) and desired (dashed) passband phase responses of the 1-D migration filter. (c) Magnitude response of the 2-D migration filter. (d) Phase response of the 2-D migration filter in the passband.

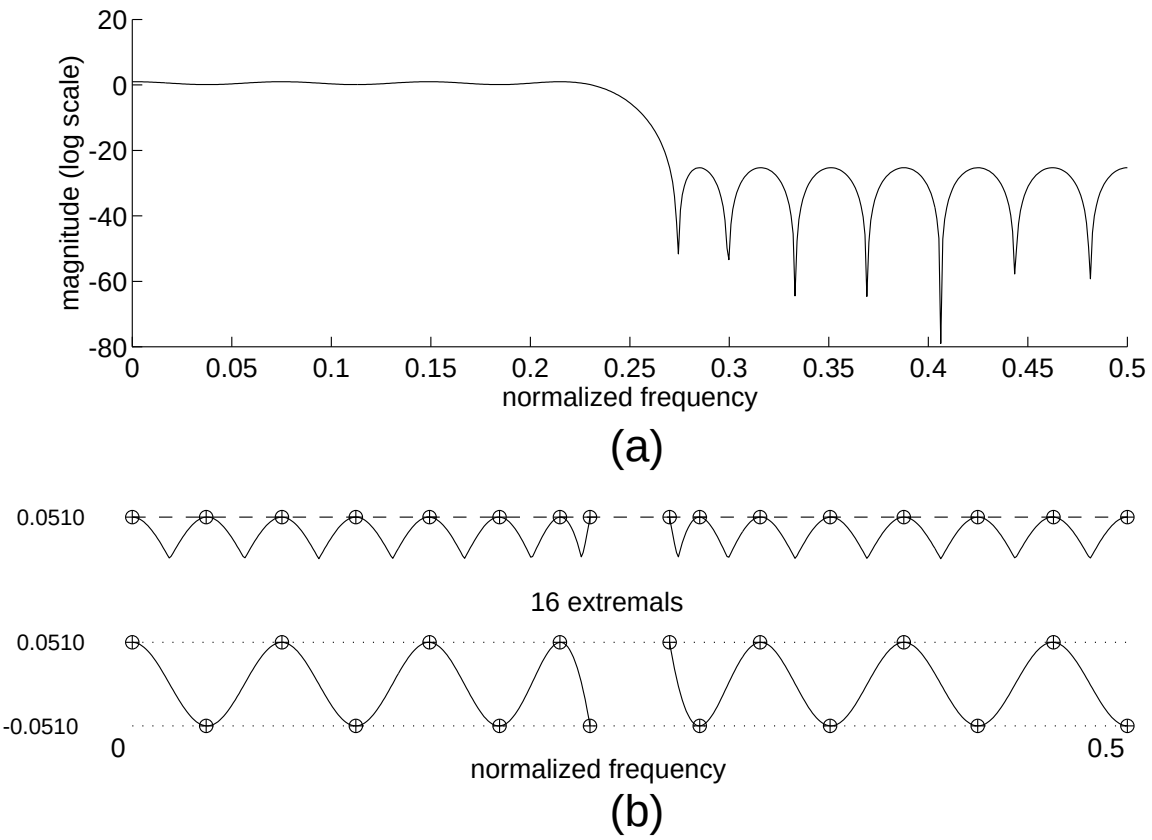


Figure 5.20. Lowpass Filter Family Prototype, $N = 29$. (a) Magnitude response in dB. (b) Magnitude of the error (top), and alternating real rotated error (bottom).

5.4.2 Linear-Phase Variable-Cutoff Filters

A complete family of linear-phase variable-cutoff filters is generated by using the first-order subfilter (5.23) and designing the prototype filter (5.29) with the following design specifications

$$P_D(\omega) = \begin{cases} 1, & |\omega_1| < \dfrac{\pi}{2} \\ 0, & \text{otherwise} \end{cases} \tag{5.61}$$

Approximating filter length: $N = 29$.

The weighting is the same for the passband and stopband. The multiple-exchange ascent algorithm converged to the optimal solution in 6 iterations of the complex

Remez stage with $\|E_{opt}\| = 0.051$. The properties of the resulting prototype filter are shown in Fig. 5.20.

Figures 5.21 and 5.22 display the properties of the resulting 1-D and 2-D filters for different cutoff frequencies ω_c. The coefficients of these filters were obtained using the Chebyshev structure shown in Figs. 5.2 and 5.3. It should be noted that the filter with low cutoff $\omega_c = 0.1\pi$ has a frequency response which is attenuated in the passband. In addition, the transition band is larger for the filters with $\omega_c = 0.1\pi$ and $\omega_c = 0.75\pi$. The reason for the attenuation and transition band stretch can be clearly seen from Fig. 5.6 which shows that the first-order contours deviate most from the ideal line contours at the high and low values of $\omega_2 = \omega_c$. Since the value in the passband is constant, filters with middle cutoff frequencies (e.g. $1 \leq \omega_c \leq 2.25$) are approximated very well because the flow segments match the ideal contours near the cutoff edge. Furthermore, these filters have a transition width comparable to the prototype filter. As the cutoff frequency decreases or increases with respect to the middle-range frequencies, the flow segments deviate from the ideal contours and map the transition band to a larger part of the passband. Higher order subfilters, satisfying the constraints (5.35) and (5.36), should improve the approximation of the ideal subfilter (5.25).

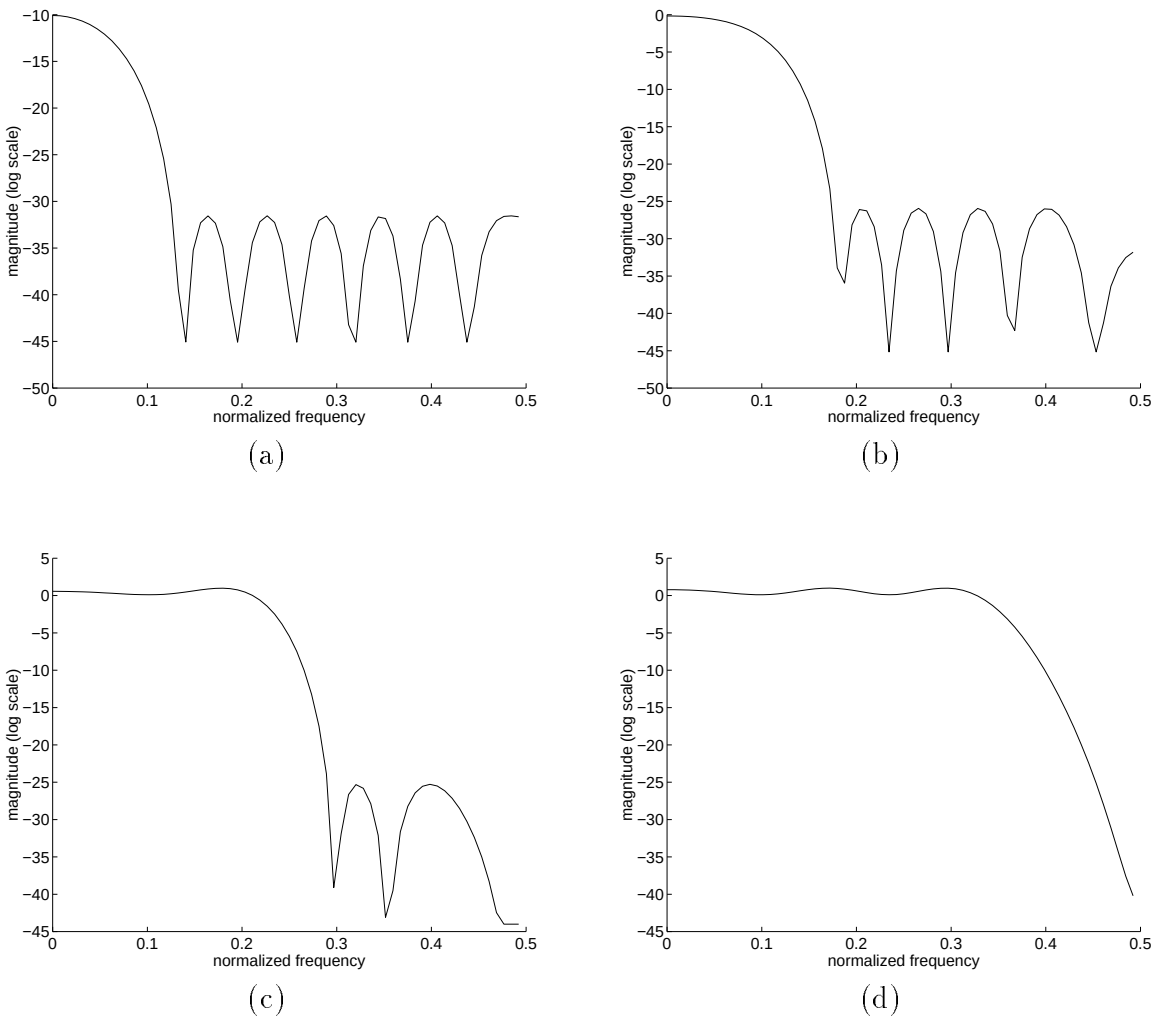


Figure 5.21. Frequency response of the designed 1-D, zero-phase, variable-cutoff filter shown for different cutoff frequencies. (a) $\omega_c = 0.1\pi$. (b) $\omega_c = 0.25\pi$. (c) $\omega_c = 0.5\pi$. (d) $\omega_c = 0.75\pi$.

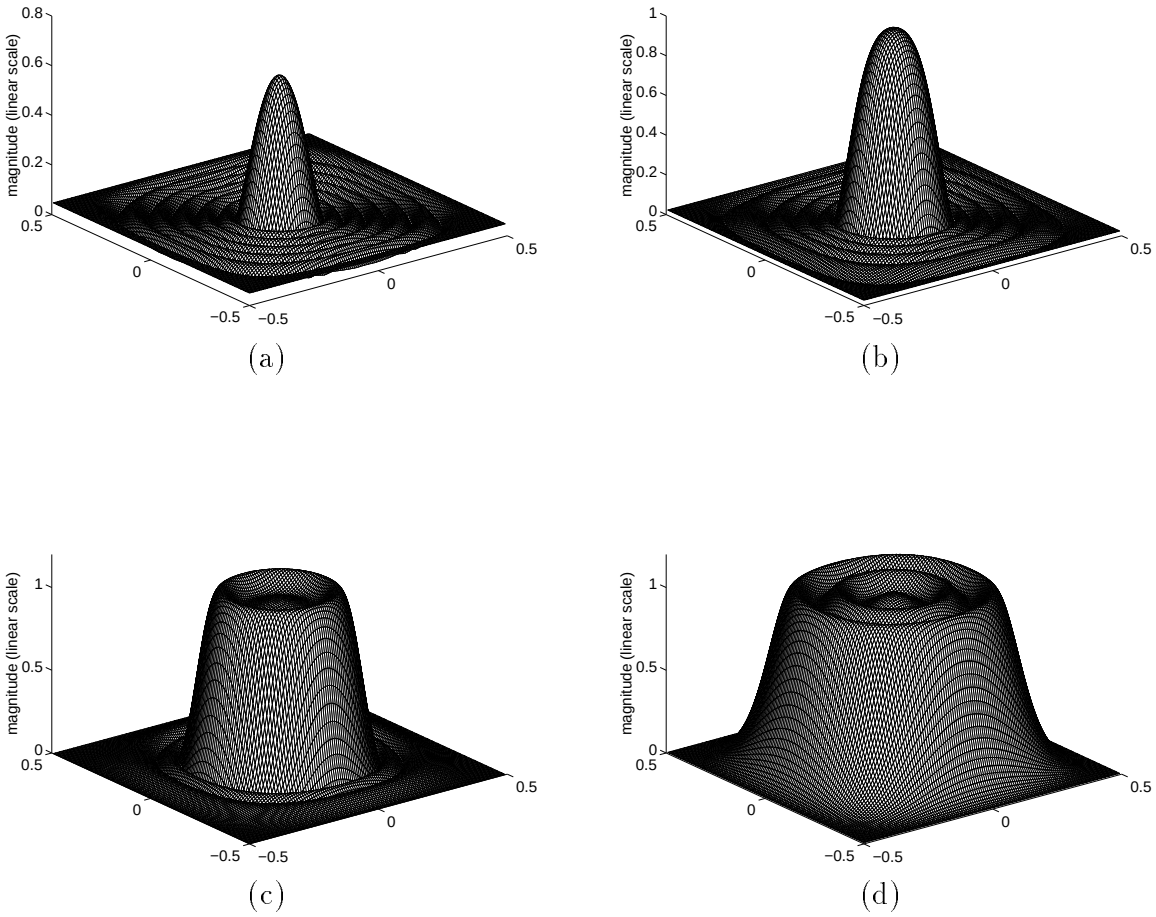


Figure 5.22. Frequency response of the designed 2-D, zero-phase, variable-cutoff filter shown for different cutoff frequencies. (a) $\omega_c = 0.1\pi$. (b) $\omega_c = 0.25\pi$. (c) $\omega_c = 0.5\pi$. (d) $\omega_c = 0.75\pi$.

CHAPTER 6

Conclusion

As a result of this research, digital filters with arbitrary magnitude and phase responses can now be efficiently designed and realized. Such capability should provide improvement in several digital signal processing areas including seismic migration, magnitude and phase equalization, low delay filtering, and possibly image processing. On the theoretical level, this research advances the theory of Chebyshev approximation and its application to filter design. A well-defined set of theoretical questions remain, but this work provides new means by which such research directions can be explored.

6.1 Contributions of the Thesis

6.1.1 Chebyshev Approximation Theory

To exploit the fact that only bounded finite sets of points need to be considered in computing the best Chebyshev approximation of a function on any compact domain, theorems for constructing the optimal Chebyshev approximations on finite sets of points are presented and proved. New optimality criteria for complex Chebyshev approximation are also derived.

Optimal Solutions on Finite Multi-dimensional Sets

Equations were derived for solving the complex Chebyshev approximation problem on arbitrary finite sets of points in a compact multi-dimensional space. The derived equations were shown to hold even in the absence of the Haar condition. A proof of

the validity of the derived equations was also presented. The difficulty of deriving an efficient point-exchange strategy for solving the general complex Chebyshev problem was reported and the reason behind this difficulty was identified. The importance of the case where the Haar condition is satisfied and the finite set of interest consists of a minimal number of points was demonstrated.

Efficient Characterization of Optimal Solutions

The known optimality criteria for complex Chebyshev approximation are of great theoretical interest but are not convenient for practical use. In order to derive a simple and exploitable characterization of optimal Chebyshev solutions, constraints on the general Chebyshev problem were identified and a new formulation of the one-dimensional Chebyshev problem was derived. The key element of this reformulation is to convert the complex Chebyshev problem into one in terms of real basis functions with complex coefficients.

New simple and practical optimality criteria were presented and proved. In particular, conditions were derived under which the alternation theorem for real Chebyshev approximation extends to the complex case. As a result, complex alternation theorems were presented and proved. The alternation property was shown to be a sufficient condition for optimality in the complex case. An important case where the complex alternation property is both necessary and sufficient for optimality was identified.

A simple and efficient way was presented for computing the optimal complex Chebyshev approximation on finite sets containing $n+1$ points, where n is the dimension of the approximating space. By exploiting the complex alternation property, very efficient multiple-exchange rules were derived for constructing a sequence of sets containing $n+1$ points such that the best Chebyshev approximations on these sets have increasing error norms. It was shown that, by transforming the complex error function, the very efficient second Remez exchange algorithm can be used in comput-

ing the optimal Chebyshev approximation.

6.1.2 Filter Design Theory

New formulations of the Filter Design Problem

A new formulation was derived for the one-dimensional complex filter design problem. This formulation was used to show that the one-dimensional filter design problem is a constrained Chebyshev problem and, thus, satisfies the Haar condition. This fact is important because it shows that, contrary to what has been reported previously in the literature [17, 27], the optimal designed filter is unique and no ill-conditioned behavior (called *degeneracy*) should be expected. It was also shown that the one- and multi-dimensional complex filter design problems can be formulated and solved as the minimization of a non-differentiable convex function.

Applicability and limitations of the Transformation Method

It was shown that only a very restricted class of well-behaved mappings could be used to transform an arbitrary prototype filter which has no particular symmetry, and that the resulting multi-dimensional filters were restricted to have only linear contours. Therefore, a general expression for the frequency response of a one-dimensional complex filter was derived in terms of Chebyshev polynomials and other new polynomials satisfying recurrence formulae. Using this derived expression, a method was proposed for obtaining multi-dimensional filters with general contours from arbitrary prototypes, but at the expense of distorting the values of the frequency response of the prototype filter. However, it is usually desirable to use only well-behaved mappings which do not alter but exactly map the values of the prototype frequency response to contours in the multi-dimensional frequency domain. Well-behaved transformation functions are important because they allow the designer to easily control the values of the designed filter and, therefore, to control the Chebyshev approximation error norm. Consequently, it was shown that a general well-behaved mapping, which places no re-

strictions on the form of the generated contours, can only be used when the prototype filter satisfies some symmetry conditions. These required symmetry conditions were derived using the general expression mentioned earlier for the prototype frequency response.

The types of positive and negative symmetries that can be designed using the transformation method were also identified. As a result, for each identified type of symmetry, conditions were derived on the mapping function and on the prototype filters for obtaining the complex two-dimensional filter with the desired symmetry. It was shown that both odd- and even-length prototype filters can be used to obtain any of the possible symmetries.

Another contribution is showing that even-length prototype filters can be used to design complex multi-dimensional filters with arbitrary symmetries and contours and whose rectangular region of support has odd-length sides. A general formula was also derived for generating an important class of transformation functions which were then used in designing seismic migration filters, fan filters, and variable-cutoff filters.

6.1.3 Chebyshev Approximation Algorithms

A basic efficient strategy for computing the optimal Chebyshev solution is to work on small finite sets of points and generate a sequence of best approximations on these sets with increasing error norms. When the maximal deviation is attained the computed solution is the desired optimal Chebyshev solution. Two multiple-exchange ascent algorithms were presented for constructing a sequence of best approximations on finite sets with increasing error norms: an extremely efficient complex Remez algorithm, which was shown to be a generalization of the classic second Remez (Parks–McClellan) algorithm to the complex case, and a robust ascent-descent algorithm which makes use of a descent method suitably adapted for minimizing the non-differentiable Chebyshev error norm. The complex Remez algorithm was derived based on the new optimality criteria presented in this thesis. The performance of this algorithm was compared

to the other existing techniques. This comparison shows that the proposed algorithm is significantly more efficient and superior in terms of memory requirements, computational complexity, and speed of convergence. Although very efficient, the complex Remez algorithm may not converge to the optimal Chebyshev approximation on the desired approximation domain. This algorithm converges to the optimal solution when the complex error satisfies an alternation property. In the general case, it converges to the optimal Chebyshev solution over a subset of the specified approximation domain. The reason for which the complex Remez algorithm does not always result in the optimal Chebyshev solution on the domain of interest was identified and explained. It was also shown that the complex Remez algorithm reduces to the classic real Remez algorithm when real-valued functions and frequency responses are approximated. The ascent-descent algorithm is more computationally expensive and less efficient than the complex Remez algorithm but it always results in the optimal Chebyshev approximation on the domain of interest. It was also shown that the ascent-descent algorithm can be generalized to solve the complex multi-dimensional Chebyshev approximation problem.

A robust and efficient multiple-exchange ascent algorithm was presented for designing optimal one-dimensional Chebyshev FIR filters with arbitrary magnitude and phase specifications and, more generally, for computing the best Chebyshev approximation of an arbitrary complex-valued function. This new approximation algorithm is the result of combining the complex Remez and the ascent-descent algorithms in a two-stage fashion: the complex Remez algorithm as the first stage followed by the ascent-descent algorithm as the second stage. In this way, the efficiency of the complex Remez method is exploited while guaranteeing convergence to the optimal solution on the desired domain. The proposed approximation algorithm was implemented and used to design low-delay, exactly linear-phase, and seismic migration filters. Complex prototype filters, which were used to design two-dimensional filters using transformations, were also designed using the new approximation algorithm. Compared to other

techniques [22, 19], the presented two-stage algorithm was shown to exhibit better convergence characteristics in terms of the number of major iterations it requires for computing the optimal solution and it was found to be significantly more efficient when the optimal solution is computed by the complex Remez stage, in which case no major iterations were required.

The following observation is made to point out the important fact that the general ascent structure of the algorithm as well as its convergence (in terms of major iterations) is independent of the method used in the second stage to compute the best approximations on the constructed finite sets of points. It should be noted that any method which can be used for computing the best Chebyshev approximation on a finite set of more than $n+1$ points (n being the dimension of the approximating space) can be substituted for the descent method in the second stage of the proposed algorithm. This substitution will not affect the total number of major iterations, which was shown to be smaller than the number of major iterations needed by other techniques (e.g., [22]). However, it does affect the computational complexity per major iteration. Also the main structure of the algorithm remains unchanged by this substitution. The descent method was presented in this work as a new attractive approach to solving the filter design problem, especially that it can be directly generalized to the multi-dimensional case.

6.1.4 Design of Filters by Transformation

The McClellan transformation method was extended so that new types of multi-dimensional filters can be designed and new types of prototype filters can be transformed. A design procedure was presented for transforming even-length, complex and real, positive- and negative-symmetric prototype filters into two-dimensional filters which can have arbitrary contours and whose rectangular region of support has odd-length sides. The procedure by Mersereau et al. [57] for designing quadrant-symmetric two-dimensional filters with a rectangular region of support having even-length sides

was extended so that both positive- and negative-symmetric, complex and real, two-dimensional filters can be designed. Practical rules were also presented specifying how to select the appropriate pair of prototype filter and transformation mapping that would result in some desired symmetry properties. Positive and negative centro symmetries as well as positive and negative quadrant symmetries were considered. The proposed design procedures and rules were used to design both complex and real, positive- and negative-symmetric, two-dimensional filters whose rectangular regions of support have odd- or even-length sides.

A new efficient transformation method for designing families of one- and multi-dimensional filters was presented. While the McClellan transformation method is suitable for designing one multi-dimensional filter at a time, the proposed method designs a complete set of one- and multi-dimensional filters by transforming a single one-dimensional prototype filter. The new method consists of three main parts: the design or selection of a two-dimensional transformation subfilter approximating the contours of the desired family of filters, the selection and design of a one-dimensional prototype filter which is mapped by the subfilter to produce the desired set of filters, and the retrieval of the individual filters in the set. The subfilter needs to be designed only once per desired filter set. Procedures for implementing each of the aforementioned three parts were presented for the design of families of seismic migration filters and exactly linear-phase filters. A new two-in-one Chebyshev structure was introduced for the efficient realization of the multi-dimensional filters in a set designed by the proposed method. The presented method is very fast because most of the computing time is devoted to the design of a single one-dimensional filter and a single low-order mapping subfilter. Moreover, the low-order subfilter can, in many cases, be selected based on experience. This new method has the additional advantage that the one- and multi-dimensional filters in the designed set can be realized efficiently without having to explicitly compute the coefficients of the separate filters.

6.2 Directions for Future Research

This research leads to several new possible research directions.

Digital filtering is a basic operation in digital signal processing. The applications considered in this thesis are just a few of the many ones possible for filters with a complex-valued frequency response. It would be of interest to explore new applications for complex filters. One possible direction would be to investigate how complex filters could be useful for processing the phase information in signals. Another possible direction is to study whether complex filters can be utilized to construct low-delay filter banks [68]. The low-delay feature is important, especially for real-time applications.

In this thesis, a very efficient complex Remez algorithm was presented. Formal convergence proofs and assessment of the rate of convergence of this algorithm are yet to be developed. It is known that this algorithm has a quadratic convergence rate at least when real-valued functions are approximated [69]. It remains to show whether this quadratic convergence behavior holds when complex functions are approximated.

An efficient non-iterative procedure was presented to compute the complex best Chebyshev approximation on sets of $n+1$ points, where n is the dimension of the approximating space, and used as a part of the complex Remez algorithm. An open question in Chebyshev approximation is whether a procedure with a comparable efficiency can be devised for computing the complex best approximation on finite sets with more than $n+1$ but less than $2n+1$ points. Such a procedure could be used in place of the descent method in the second stage of the approximation algorithm presented in this thesis. Another open question in Chebyshev approximation is whether efficient point-exchange rules can be derived for sets of more than $n+1$ points. While such exchange rules were developed in this thesis for sets having a fixed size of $n+1$ points, it is not clear if these rules can be generalized for finite sets with a larger number of points.

Some of the results and methods presented in this thesis deserve further study.

Equations were derived in Chapter 2 for computing the optimal Chebyshev solution on finite sets with an arbitrary number of points in a multi-dimensional space. It would be worthwhile to investigate efficient ways to solve these equations. Chapter 3 proposed an algorithm for solving the multi-dimensional complex Chebyshev problem. The implementation issue of this general algorithm will be considered in a future work. Chapter 4 proposed a method for designing multi-dimensional filters by transforming an arbitrary prototype filter. The practical use of this method still needs to be investigated. Chapter 5 presented an efficient procedure for designing families of seismic migration and linear-phase filters. One worthwhile line of research is to explore what additional types of filters could be designed by this procedure. In particular, it would be of interest to examine if this procedure could be used for designing families of directional fan filters.

APPENDIX A

Notation and Terminology

A.1 Notation

- **Boldface** lower case letters (e.g. $\mathbf{x}$) correspond to vectors or multi-dimensional variables.
- Regular lower case letters (e.g. x) correspond to a scalar variable.
- $|\delta|$: lower bound computed by the complex Remez algorithm of Chapter 3, and optimal deviation on sets of $n+1$ points, where n is the dimension of the approximating space.
- ω: frequency variable in radians.
- N: length of a one-dimensional FIR filter.
- h: impulse response of an FIR filter.
- B: domain of approximation.
- $\|f\|$: is used to denote the Chebyshev norm of a function f unless it is otherwise indicated. The Chebyshev norm of a function f defined on a domain B is given by

$$\|f\| = \max_{\mathbf{x} \in B} |f(\mathbf{x})|$$

  where $|f(\mathbf{x})|$ is the magnitude of $f(\mathbf{x})$.
- ε_o: optimal approximation error on the domain of interest B.
- 1-D: abbreviation for "one-dimensional".

- 2-D: abbreviation for "two-dimensional".
- multi-D : abbreviation for "multi-dimensional".

A.2 Terminology

- The term **real filter** refers to a filter with a real-valued frequency response. Thus, a real filter can have a complex (conjugate symmetric) impulse response.
- The term **complex filter** refers to a filter with a complex-valued frequency response. A complex filter can thus have a real impulse response.
- The term **real case** refers to the case where the functions or frequency responses being approximated are real-valued.
- The term **complex case** refers to the case where the functions or frequency responses being approximated are complex-valued. Note that the complex case includes the real case.
- The term **well-behaved** mapping function refers to a transformation function that exactly maps (without distortion) the values of the frequency response of a prototype filter to contours in the multi-dimensional space.

APPENDIX B

Mathematical Derivations

B.1 Addendum to Chapter 3

B.1.1 Derivation of (3.14)

$$\delta_{\text{new}} = e^{j\rho\pi}\left(\delta(1-|\lambda'_{i_o}|)+|\lambda'_{i_o}|(\text{sgn}\{\lambda'_{i_o}\}e^{j\rho\pi}\varepsilon(\zeta))\right)$$

Derivation: From (3.13),

$$\begin{aligned}\delta_{\text{new}} &= \textstyle\sum_{i=1}^{n+1}\lambda'_i\varepsilon(x'_i)\\ &= \textstyle\sum_{\substack{i=1\\ i\neq i_o}}^{n+1}\lambda'_i\varepsilon(x'_i)+\lambda'_{i_o}\varepsilon(\zeta)\end{aligned}$$

Note that $|\varepsilon(x'_i)| = |\delta|$ and $\theta_{\varepsilon(x'_i)} = \theta_{\varepsilon(x_i)} + \rho\pi$ for $i \neq i_o$, leading to

$$\begin{aligned}\delta_{\text{new}} &= \textstyle\sum_{\substack{i=1\\ i\neq i_o}}^{n+1}(-1)^{i+1}|\lambda'_i||\delta|e^{j\theta_{\varepsilon(x_i)}}e^{j\rho\pi}+\lambda'_{i_o}\varepsilon(\zeta)\\ &= e^{j\rho\pi}\textstyle\sum_{\substack{i=1\\ i\neq i_o}}^{n+1}|\lambda'_i||\delta|e^{j\theta_\delta}+\lambda'_{i_o}\varepsilon(\zeta)\\ &= e^{j\rho\pi}\textstyle\sum_{\substack{i=1\\ i\neq i_o}}^{n+1}|\lambda'_i|\delta+\lambda'_{i_o}\varepsilon(\zeta)\\ &= e^{j\rho\pi}\delta(1-|\lambda'_{i_o}|)+\lambda'_{i_o}\varepsilon(\zeta).\end{aligned}$$

Factoring out $e^{j\rho\pi}$ and rewriting λ'_{i_o} as $|\lambda'_{i_o}|\text{sgn}\{\lambda'_{i_o}\}$ result in (3.14).

B.2 Addendum to Chapter 4

B.2.1 Proof of Theorem 4.1

Proof: Equation (4.8) implies that

$$|F(\omega_1, \omega_2)|^2 = 1 \tag{B.1}$$

and, therefore,

$$AC(q, r) = \delta(q, r) \tag{B.2}$$

where $\delta(q, r)$ is the dirac delta function and $AC(q, r)$ is the autocorrelation of $f(q, r)$ in (4.7). $AC(q, r)$ is given by

$$AC(q, r) = f(q, r) * f^{\star}(-q, -r) \tag{B.3}$$

where $f^{\star}(-q, -r)$ is the complex conjugate of $f(-q, -r)$ and $*$ refers to the linear convolution operation [1].

Now, assume that $f(q, r)$ has nonzero samples over the region $\{Q_1 \leq q \leq Q_2\,,$ $R_1 \leq r \leq R_2\}$ with $f(Q_1, R_1) \neq 0$ and $f(Q_2, R_2) \neq 0$. Then (B.3) can be rewritten as

$$\begin{cases} \displaystyle\sum_{m=Q_1}^{Q_2-|q|} \sum_{n=R_1}^{R_2-|r|} f(m, n) f^{\star}(|q| + m, |r| + n)\,, & 0 \leq |q| \leq Q_2 - Q_1 \\ & 0 \leq |r| \leq R_2 - R_1 \\ 0\,, & \text{otherwise} \end{cases} \tag{B.4}$$

In partcular, note that

$$AC(Q_2 - Q_1, R_2 - R_1) = AC(Q_1 - Q_2, R_1 - R_2) = f(Q_1, R_1) f^{\star}(Q_2, R_2) \neq 0.$$

So, if $Q_2 \neq Q_1$ or $R_2 \neq R_1$, $AC(q, r)$ would have at least three non-zero samples $AC(0)$, $AC(Q_2 - Q_1, R_2 - R_1)$, and $AC(Q_1 - Q_2, R_1 - R_2)$. It follows that (B.2) is satisfied only if $Q_2 = Q_1$ and $R_2 = R_1$, which implies that $f(q, r)$ has exactly one non-zero sample $f(Q_1, R_1)$. In addition, $|f(Q_1, R_1)| = 1$ since, from (B.2),

$$AC(0, 0) = |f(Q_1, R_1)|^2 = 1. \tag{B.5}$$

Consequently,

$$\begin{aligned} F(\omega_1,\omega_2) &= f(Q_1,R_1)\, e^{-j(\omega_1 Q_1+\omega_2 R_1)} \\ &= e^{-j(\omega_1 Q_1+\omega_2 R_1-\theta_f)} \end{aligned} \tag{B.6}$$

where θ_f is the phase of $f(Q_1,R_1)$. Setting $n_1 = Q_1$, $n_2 = R_1$, and $C = -\theta_f$ in (B.6) gives (4.9). □

B.2.2 Proof of recurrence formula (4.14)

$$\begin{cases} S_k[x] &= \begin{cases} 2S_{k-1}[x](1-x^2) - S_{k-2}[x], & k \text{ odd} \\ 2S_{k-1}[x] - S_{k-2}[x], & k \text{ even} \end{cases} \\ S_0[x] &= 0 \\ S_1[x] &= x \end{cases}$$

Proof: Using the known mathematical relations for the trigonometric function $\sin(k\omega)$ where k is an integer parameter greater than 1 and ω is a real variable, one can write:

$$\sin(k\omega) = \sin((k-1)\omega)\cos\omega + \cos((k-1)\omega)\sin\omega \tag{B.7}$$

$$\sin((k-2)\omega) = \sin((k-1)\omega)\cos\omega - \cos((k-1)\omega)\sin\omega. \tag{B.8}$$

Now, adding (B.7) and (B.8) gives

$$\sin(k\omega) = 2\sin((k-1)\omega)\cos\omega - \sin((k-2)\omega). \tag{B.9}$$

Two cases are distinguished:

- k odd

  From (4.12),

$$\begin{cases} \sin(k\omega) &= S_k[\sin\omega] \\ \sin((k-1)\omega) &= S_{k-1}[\sin\omega]\cos\omega. \end{cases} \tag{B.10}$$

Equations (B.9) and (B.10) lead to

$$S_k[\sin\omega] = 2S_{k-1}[\sin\omega]\cos^2(\omega) - S_{k-2}[\sin\omega]. \tag{B.11}$$

Setting $x = \sin\omega$, (B.11) becomes

$$S_k[x] = 2S_{k-1}[x](1-x^2) - S_{k-2}[x] \tag{B.12}$$

which is the recurrence relation (4.14) when k is odd.

- k even

  From (4.12),

$$\begin{cases} \sin(k\omega) &= S_k[\sin\omega]\cos\omega \\ \sin((k-1)\omega) &= S_{k-1}[\sin\omega]. \end{cases} \tag{B.13}$$

  Equations (B.9) and (B.13) lead to

$$S_k[\sin\omega]\cos\omega = 2S_{k-1}[\sin\omega]\cos\omega - S_{k-2}[\sin\omega]\cos\omega. \tag{B.14}$$

  Setting $x = \sin\omega$, (B.14) becomes

$$S_k[x] = 2S_{k-1}[x] - S_{k-2}[x] \tag{B.15}$$

  which is the recurrence relation (4.14) when k is even. □

B.2.3 Proof of relation (4.15)

$$S_k[x] = \begin{cases} (-1)^m T_{2m+1}[x], & k = 2m+1 \ (k \text{ odd}) \\ 2(-1)^{m-1}\sum_{p=1}^{m} T_{2p-1}[x], & k = 2m \ (k \text{ even}) \text{ and } k \neq 0 \end{cases}$$

Proof:

- k odd

  Let $k = 2m+1$, where m is an integer. Using (4.12),

$$S_k[\sin\omega] = \sin(k\omega)$$

$$
\begin{aligned}
&= \cos(k\omega - \frac{\pi}{2}) \\
&= (-1)^m \cos(k\omega - \frac{\pi}{2} - m\pi) \\
&= (-1)^m \cos(k\omega - \frac{\pi}{2} - 2m\frac{\pi}{2}) \\
&= (-1)^m \cos(k\omega - k\frac{\pi}{2}) \\
&= (-1)^m T_k[\cos(\omega - \frac{\pi}{2})] \\
&= (-1)^m T_k[\sin(\omega)]. \qquad \text{(B.16)}
\end{aligned}
$$

Now, substituting x for $\sin\omega$ and $2m+1$ for k in (B.16) leads to the relation (4.15) for k odd.

- k even

  With $k = 2m$, the proof is carried out by induction on m.

  The relation (4.15) holds for $m = 1$ and $m = 2$. In fact, from (4.14),

$$
\begin{aligned}
S_2[x] &= 2S_1 - S_0 \\
&= 2T_1[x] \\
&= 2(-1)^0 \sum_{p=1}^{1} T_{2p-1}[x].
\end{aligned}
$$

  For $m = 2$, the relation (4.15), which was proved above for $S_k[x]$ when k is odd, is used in addition to (4.14) leading to

$$
\begin{aligned}
S_4[x] &= 2S_3 - S_2 \\
&= -2T_3[x] - 2T_1[x] \\
&= 2(-1)^1 \sum_{p=1}^{2} T_{2p-1}[x].
\end{aligned}
$$

  Now, assuming that (4.15) holds for $m-1$, i.e.,

$$
S_{2(m-1)} = 2(-1)^{m-2} \sum_{p=1}^{m-1} T_{2p-1}[x]\,, \qquad \text{(B.17)}
$$

  it is required to prove that (4.15) also holds for m. From (4.14),

$$
S_{2m} = 2S_{2m-1} - S_{2(m-1)}. \qquad \text{(B.18)}
$$

Using (4.15) for k odd and (B.17), (B.18) becomes

$$\begin{aligned} S_{2m} &= 2(-1)^{m-1} T_{2m-1}[x] - 2(-1)^{m-2} \sum_{p=1}^{m-1} T_{2p-1}[x] \\ &= 2(-1)^{m-1} \{T_{2m-1}[x] + \sum_{p=1}^{m-1} T_{2p-1}[x]\} \\ &= 2(-1)^{m-1} \sum_{p=1}^{m} T_{2p-1}[x]. \end{aligned}$$

□

B.2.4 Proof of relation (4.16)

$$T_{2m}[\cos\omega] = (-1)^m T_{2m}[\sin\omega]$$

Proof:

$$\begin{aligned} T_{2m}[\cos\omega] &= \cos(2m\omega) \\ &= (-1)^m \cos(2m\omega - m\pi) \\ &= (-1)^m \cos(2m(\omega - \frac{\pi}{2})) \\ &= (-1)^m T_{2m}[\cos(\omega - \frac{\pi}{2})] \\ &= (-1)^m T_{2m}[\sin\omega]\,. \end{aligned}$$

□

B.2.5 Proof of relation (4.17)

$$S_{2m}[\sin\omega]\cos\omega = (-1)^{m+1} S_{2m}[\cos\omega]\sin\omega$$

Proof: From (4.12),

$$\begin{aligned}
S_{2m}[\sin\omega]\cos\omega &= \sin(2m\omega) \\
&= (-1)^m \sin(2m\omega - m\pi) \\
&= (-1)^m \sin(2m(\omega - \frac{\pi}{2})) \\
&= (-1)^{m+1} \sin(2m(\frac{\pi}{2} - \omega)) \\
&= (-1)^{m+1} S_{2m}[\sin(\frac{\pi}{2} - \omega)]\cos(\frac{\pi}{2} - \omega) \\
&= (-1)^{m+1} S_{2m}[\cos\omega]\sin\omega.
\end{aligned}$$

□

APPENDIX C

Descent Method Based on the Kolmogoroff Criterion

A new descent method is presented here where the descent directions $\{\mathbf{p}_k\}$ are efficiently computed by exploiting the Kolmogoroff optimality criteria (Theorem 2.1) for Chebyshev approximation. It is assumed that the approximating space V_n satisfies the Haar condition.

If n_e denotes the number of extremal points, descent directions can be computed efficiently when $n_e \leq n$. This is due to the fact that a function $\phi(\mathbf{x}) = \sum_{i=1}^{n} c_i \phi_i(\mathbf{x})$ can be interpolated through the n_e extremal points such that the Kolmogoroff optimality criterion is violated. This is no longer true for $n_e > n$ since $\phi(\mathbf{x})$, which is a generalized polynomial of degree n, cannot be specified at more than n points.

The Kolmogoroff descent method is based on the fact that any function $p(\mathbf{x}) = \Phi^T \mathbf{c}$ violating (2.4–2.5) corresponds to a descent direction $\mathbf{p} = \mathbf{c}$ [31, pp. 14–15]. The proposed algorithm requires n_e to be less than n. It has been observed that n_e is usually $\leq n$ throughout the descent iterations.

Before proceeding with the Kolmogoroff-based descent algorithm description, some additional notations are needed:

k: refers to the kth iteration.

ϵ_m, ϵ_s,ϵ_t: tolerances between 0 and 1.

δ: lower bound from ascent stage.

$\varepsilon_k(\mathbf{x})$: error $\varepsilon_k(\mathbf{x}) = f(\mathbf{x}) - \phi^{(k)}(\mathbf{x})$.

$\mathbf{x}_e$: extremal points at which error attains its maximal norm.

$\mathbf{x}_{lm}$: points corresponding to local maxima of $|\varepsilon_k(x)|$ (including $\mathbf{x}_e$).

$\mathbf{x}_{\epsilon_m}$: $\mathbf{x}_{lm}$ satisfying $|\varepsilon_k(\mathbf{x}_{lm})|/|\varepsilon_k(\mathbf{x}_e)| \geq (1-\epsilon_m)$.

n_e,n_{lm},n_{ϵ_m}: number of points $\mathbf{x}_e$, $\mathbf{x}_{lm}$, and $\mathbf{x}_{\epsilon_m}$ respectively.

$p^{(k)}(\mathbf{x})$: perturbation function violating (2.4–2.5).

t_k: step corresponding to the descent direction $p^{(k)}(\mathbf{x})$.

$\theta_i^{(k)}$: phase of $\varepsilon_k(\mathbf{x}_i)$.

mn_k: $\sum_{i=1}^{n_{lm}} |\varepsilon_k(x_{lm,i})|/n_{lm}$.

$r_k(\alpha)$: defined as $r_k(\alpha) = (\|\varepsilon_k\| - |\alpha|)/|\alpha|$.

Initially, let $\phi^{(0)}(\mathbf{x})$ and $\varepsilon_0(\mathbf{x}) = f(\mathbf{x}) - \phi^{(0)}(\mathbf{x})$ be, respectively, the starting approximation and error obtained from the ascent stage with $n_{lm} \geq n$ and $n_e \leq n$. It is assumed that $n_e \leq n$ throughout the iterations. A convenient starting value for ϵ_m is 0.005.

At each iteration k, the descent procedures identifies n points (Step 1) and sets up an interpolation problem (Step 2) to generate the descent function $p^{(k)}(\mathbf{x})$ (Step 3). In all the iterations, the local maxima $\mathbf{x}_{lm}$ are ordered such that $|\varepsilon(\mathbf{x}_{lm,1})| \leq |\varepsilon(\mathbf{x}_{lm,2})| \ldots \leq |\varepsilon(\mathbf{x}_{lm,n_{lm}})|$. Since the passage from $\mathbf{c}_k$ to $\mathbf{c}_{k+1}$ $(k = 0, 1, \ldots)$ is the generic update, suppose that the k-th approximation $\phi^{(k)}(\mathbf{x})$ is already available. The kth iteration of the descent method constitutes one execution of the following steps:

Step 1 Find all $\mathbf{x}_e^{(k)}$, $\mathbf{x}_{lm}^{(k)}$, and $\mathbf{x}_{\epsilon_m}^{(k)}$; compute mn_k.

Step 2 If $r_k(\delta) \leq \epsilon_s$ or $r_k(mn_k) \leq \epsilon_s$, then take $\phi^{(k)}$ as the best approximation of f on B. Done

Step 3 If $(n_e > n)$, go to Step 11.

Step 4 If $(n_{\epsilon_m} > n)$, $\epsilon_m = \epsilon_m/2$ and repeat from Step 1.

Step 5 If $n_{lm} < n$, add $(n - n_{lm})$ distinct frequencies from $f_{lm}^{(k-1)}$ to $f_{lm}^{(k)}$.

Step 6 Select n points $\mathbf{x} \in B$ consisting of $\{$ (all $\mathbf{x}_{\epsilon_m}^{(k)}$) $+(\mathbf{x}_{lm,1}^{(k)}, \ldots, \mathbf{x}_{lm,N-n_{\epsilon_m}^{(k)}}^{(k)})\}$.

Step 7 Specify $p^{(k)}(\mathbf{x})$ at the n points determined in Step 6

$$p^{(k)}(\mathbf{x}_{\epsilon_m,i}) = \varepsilon_k(\mathbf{x}_{\epsilon_m,i}) - |\delta| e^{j\theta_{\epsilon_m,i}}, \qquad i = 1, \ldots, n_{\epsilon_m} \tag{C.1}$$

and

$$p^{(k)}(\mathbf{x}_{lm,i}) = \begin{cases} \varepsilon_k(\mathbf{x}_{lm,i}) - |\delta| e^{j\theta_{lm,i}}, & |\varepsilon_k(\mathbf{x}_{lm,i})| < |\delta| \\ 0\,, & \text{otherwise} \end{cases} \tag{C.2}$$

for $i = 1, \ldots, (n - n_{\epsilon_m})$.

Step 8 Solve the interpolation problem of Step 7.

Step 9 Consider the ray $\phi_t(\mathbf{x})$ defined as

$$\phi_t(\mathbf{x}) = \phi^{(k)}(\mathbf{x}) + t\, p^{(k)}(\mathbf{x}) \tag{C.3}$$

and determine the step t_k such that

$$\|f(\mathbf{x}) - \phi_{t_k}(\mathbf{x})\| = \min_{t \geq 0} \|f(\mathbf{x}) - \phi_t(\mathbf{x})\|. \tag{C.4}$$

Step 10 If $t_k \leq \epsilon_t$, go to Step 11.

Step 11 Stop and use the ascent-descent method described in Chapter 3 with $\phi^{(k)}(\mathbf{x})$ serving as an initial solution.

Step 12 Set the $(k+1)$-th approximation to be

$$\phi^{(k+1)}(\mathbf{x}) = \phi_{t_k}(\mathbf{x}). \tag{C.5}$$

For the first iteration, if $|\varepsilon_k(\mathbf{x}_{lm,i})| \geq |\delta|$ for all i, it was found beneficial in some cases to replace δ by mn_k in (C.2) for that iteration.

Note that, while the choice in (C.1) is needed to violate the Kolmogoroff criterion (2.4–2.5), the choice in (C.2) is heuristic and seemed to be a suitable one. In

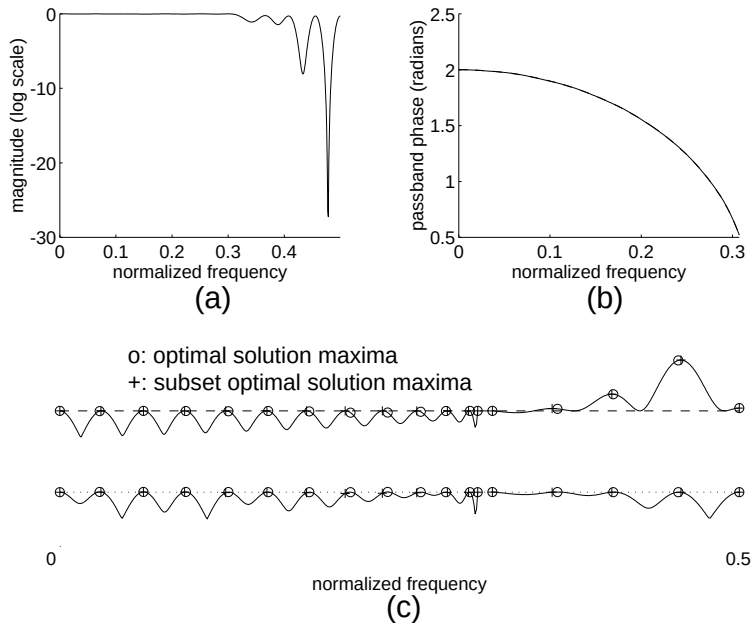


Figure C.1. Design Example: seismic migration filter, $N = 31$. (a) FIR filter magnitude response in dB. (b) FIR filter (solid) and desired filter (dashed) passband phase responses. (c) Magnitude of the weighted subset optimal error (top), and magnitude of the weighted optimal error (bottom).

fact, $p^{(k)}(\mathbf{x})$ could correspond to a poor descent direction because of the selection made in (C.2). The terminating condition $t_k < \epsilon_t$ corresponds to the case where the generated descent function $p^{(k)}(\mathbf{x})$ is so feeble that the procedure could take endless steps without reducing $\|\varepsilon(\mathbf{x})\|$ much.

Design Example

This example corresponds to a non-causal seismic migration filter with the following specifications:

$$\text{Desired: } D(f) = \begin{cases} e^{j\sqrt{(2)^2-(2\pi f)^2}} & \text{if } |f| \leq 2\sin(75°)/2\pi \\ 0 & \text{if } 2/2\pi \leq |f| \leq 0.5 \end{cases}$$

$$\text{Weight: } W(\omega) = \begin{cases} 500 & \text{if } |f| \leq 2\sin(75^o)/2\pi \\ 1 & \text{if } 2/2\pi \leq |f| \leq 0.5 \end{cases}$$

Approximating filter length: $N = 31$

The starting impulse response index is $N_1 = -15$. The large passband weighting is used to force almost perfect match in the passband. In fact, for migration filters, the approximation need only be accurate in the passband as long as the stopband magnitude deviation is not larger than unity [4, p. 361].

The optimal solution was obtained in 64 iterations (9 complex Remez stage and and 55 Kolmogoroff-based descent iterations). The filter properties are shown in Figure C.1. Since $D(f)$ is symmetric with respect to $f = 0$, the resulting optimal filter is also symmetric with an optimal error $\|E_o\|$=0.9772. The lower bound $|\delta|$, which is obtained from the ascent method, is 0.9704. Figure C.1(c) shows the magnitude of the subset-optimal error resulting from the ascent multiple-exchange stage and of the optimal error at the end of the descent stage. Note that the local maximal frequencies of the subset-optimal error matches the extremal frequencies of the optimal error. The design took 12 seconds to complete on a SUN SPARC 10 workstation.

APPENDIX D

Code Listing

D.1 Multiple-Exchange Design Algorithm (Chapter 3)

```
%%%%%% RUNNING INSTRUCTIONS:
%%%%%%     1) Run Matlab (version 4.0 or higher)
%%%%%%     2) At prompt, type:
%%%%%%                crmz_adesc
%%%%%%     3) You will be prompted for an input file
%%%%%%        name which contains the desired design
%%%%%%        specifications. Refer to the SAMPLE
%%%%%%        input file crmz_in.m below to generate
%%%%%%        your own input file.
%%%%%%
%%%%%%  NOTE: The second stage "adesc.m" is slow because
%%%%%%        of the current MATLAB implementation.
%%%%%%        It is recommended to have the descent
%%%%%%        step of the second stage inplemented as
%%%%%%        a C mex file. This will significantly
%%%%%%        shorten the running time of that stage.
%%%%%%

%%%%%%%%%% crmz_adesc.m %%%%%%%%%%
% crmz_adesc.m
% Two-stage Multiple-exchange Filter Design Algorithm
%      designs FIR filters with arbitary magnitude and
%      phase specifications.
%
% crmz_adesc.m: version 5.0 created on 02/95
%
% (c) 1995 by Lina Karam and James McClellan
```

```
%

%main program
clc; disp(' ');
disp('             ********************************************');
disp('             *****    Design of FIR Digital Filters    *****');
disp('             *****  by Complex Chebyshev Approximation *****');
disp('             *****  Version 5.0 (Matlab 4.0 or higher) *****');
disp('             ********************************************');
%%%%%%%%%%%%%% Read Design Specifications %%%%%%%%%%%%%%
J=sqrt(-1);
crmz_datain;
if ~exist('sym'), sym = 0; end;
if ~exist('N1'), N1 = 0; end;
N2 = N1 + L - 1;   % Last index of impulse response
is_odd = mod(L,2);
%%%%%%%%%%%%%%    Run Design Algorithm    %%%%%%%%%%%%%%
crmz;              %%% Complex Remez Stage
if (not_optimal)   %%% optimality flag set by Complex Remez (crmz.m)
  adesc;           %%% Ascent-descent Stage
end;
%%%%%%%%%%%%%%    Final result Display    %%%%%%%%%%%%%%
clf;
if (prg_dspl)
  crmz_plerror;
  keyboard
end;
H = HH .* exp(-J*2*pi*tgrid*(N1+N2)/2);
E = EE .* exp(-J*2*pi*grd*(N1+N2)/2);
E_pl = EE_pl .* exp(-J*2*pi*grd_pl*(N1+N2)/2);  %% Needed for plot
crmz_plresult;
impfile = crmz_input(' File to save coefficients (0 = none) : ','0');
if ~strcmp(impfile,'0')
  impfile(find(impfile=='.'):length(impfile)) = [];
  eval(['save ' impfile ' h']);
end;
%%%%%%%%%%%%%%          End Program       %%%%%%%%%%%%%%
```

```
%%%%%%%%%% SAMPLE INPUT FILE %%%%%%%%%%
% crmz_in.m
% Sample input file for crmz.m
% Version 5.0
% Input Data File (Reduced delay Linear-phase filter)
%
% NOTE:
% Only entries having the expression "USER SPECIFIED"
% next to them need to be changed by the user.
% All the other entries and variables must be left
% unchanged.
% Please, make sure to read the paragraphs starting with
% the word NOTE before you create your own design
% specifications.
%
% [lk, 01/95]

% Designed filter specifications
L = 31;             % USER SPECIFIED
                    % filter length

N1 = 0;             % USER SPECIFIED
                    % Starting index of impulse response

sym = 0;          % USER SPECIFIED
                  % 0 = no-symmetry,1 = even-symmetry,2 = odd-symmetry
                  % NOTE:
                  %  Specifies the symmetry of the function
                  %  A(w) (NOT the desired Des(w)) which is
                  %  equal to
                  %
                  %      A(w) = exp(J*w*(2*N1+L-1)/2)*Des(w)
                  %
                  %  where Des(w) is the ideal frequency response.
                  %  sym = 1 => A(w) = A(-w)
                  %  sym = 2 => A(w) = -A(-w)
                  %  Note that A(w) is in general a complex function.
                  %

% Edge band frequencies in [-0.5, 0.5] (USER SPECIFIED)
% NOTE: To ensure that the Haar condition is satisfied the
%       following rules must be followed:
```

```
%       1) if sym = 1 or sym = 2, only specify band edges which
%          are greater or equal to 0 (i.e., between 0 and 0.5).
%          Also one of the following rules may apply.
%       2) if sym = 1 and L even, last element of edges must be
%          strictly less than 0.5 (i.e., edges(length(edges)) < 0.5).
%       3) if sym = 2 and L odd, first element of edges must be
%          strictly greater than zero (i.e., edges(1) > 0), and
%          last element of edges must be strictly less than
%          0.5 (i.e., edges(length(edges)) < 0.5).
%       4) if sym = 2 and L even, first element of edges must be
%          strictly greater than zero (i.e., edges(1) > 0).
%
  edges = [-0.5 -0.12; -0.06 0.06; 0.12 0.4995];  % USER SPECIFIED

% Passbands Edge frequencies (USER SPECIFIED)
% Needed for plotting
% NOTE: if sym = 1 or sym = 2, only specify passbands edges which
%       are greater or equal to 0.
%
  pas_edges = [-0.06 0.06];                       % USER SPECIFIED

if (sym)
  str = 'half ';      % 'half' = [0,0.5]
else
  str = 'whole';      % 'whole' = [-0.5,0.5)
end;

% Generate frequency grid
[tgrid, delf, vec_edges, indx_edges] = crmz_grid(edges, L, str);
tgrid = tgrid';

% Initialization
Des = [];
WT = [];
grd = [];

% Specify desired frequency response and weighting function
% here.
% NOTE: The number of bands must be half the number of
%       edge frequencies given by "edges" above.
%       User must add bands as needed. This can be done
%       by:
```

```
%       (1). replicating the script corresponding to the
%            last band.
%       (2). incrementing by 1 the indices
%            of "indx_edges" in the first line of the script.
%       (3). specifying the values of the desired frequency
%            response, "bnd_des", and the weighting function,
%            "bnd_wt", in the considered band(s).
%
%%%% All bands are specified in a similar way.
%%%%
%%%% Band 1
% USER SPECIFIED: only specify indices (in this case, 1 and 2)
%
  bnd_grid = tgrid(indx_edges(1):indx_edges(2));
%
% USER SPECIFIED: desired frequency response in terms of "bnd_grid";
%                 "bnd_grid" represent the NORMALIZED frequency
%                 variable "f", (i.e. f = w/(2*pi) ).
%                 write the desired function in terms of "bnd_grid".
%
  bnd_des = 0 * bnd_grid;
%
% USER SPECIFIED: weighting function of the form
%                      bnd_wt = Wt *ones(length(bnd_grid),1);
%                 where "Wt" is a positive real number specified
%                 by the USER. Only "Wt" needs to be specified.
  bnd_wt = 10 * ones(length(bnd_grid),1);
%
Des = [Des; bnd_des];
WT = [WT; bnd_wt];
grd = [grd;bnd_grid];
Des_pl = [bnd_des;nan];

% Band 2
bnd_grid = tgrid(indx_edges(3):indx_edges(4)); % USER SPECIFIED indices
bnd_des = exp(-J*2*pi*bnd_grid*12);           % USER SPECIFIED
bnd_wt = ones(length(bnd_grid),1);             % USER SPECIFIED
Des = [Des; bnd_des];
WT = [WT; bnd_wt];
grd = [grd;bnd_grid];
Des_pl = [Des_pl;bnd_des;nan];
```

```
%Band 3
bnd_grid = tgrid(indx_edges(5):indx_edges(6)); % USER SPECIFIED indices
bnd_des = 0*bnd_grid;                          % USER SPECIFIED
bnd_wt = 10 * ones(length(bnd_grid),1);        % USER SPECIFIED
Des = [Des; bnd_des];
WT = [WT; bnd_wt];
grd = [grd;bnd_grid];
Des_pl = [Des_pl;bnd_des];

%%%%%%%%%% End INPUT FILE %%%%%%%%%%



%%%%%%%%%%%%%%%%%% PROGRAM ROUTINES %%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%% crmz_datain.m %%%%%%%%%%
% Prompts user for input for the filter design
% program.
%

infile = crmz_input(' Input Data File (0 = crmz_in) : ','0');
if strcmp(infile,'0'),
   infile = 'crmz_in';
else
   infile(find(infile=='.'):length(infile)) = [];
end;
prg_dspl = 2;
disp('');
disp(' Program Results : ');
disp('        (0) Graphical and Numerical Display at each iteration');
disp('        (1) Numerical Display at each iteration');
disp('        (2) Graphical and Numerical Display at last iteration');
prg_dspl = crmz_input(' Select one of the above (0, 1, or 2): ',...
                      prg_dspl);
eval(infile);
%%%%%%%%%% End crmz_datain.m %%%%%%%%%%


%%%%%%%%%% crmz_input.m %%%%%%%%%%
function y=crmz_input(str, default);
% Version 3.0
% y = crmz_input(str, default)
```

```
%        prompts the user with the string "string" and suggested
%        default value "default". Waits for input from the
%        keyboard. Returns the user input (if any) or the default
%        value if only the <return> key is pressed.
%
if nargin <1
  disp('Error: Function needs at least one argument');
elseif nargin <2
  default = [];
end;
if isstr(default)
        str = [str '( ' default ' ): '];
elseif max(size(default)) >= 1;
        str = [str '( ' num2str(default) ' ): '];
else
        str = [str ': '];
end;
y = default;
strin = input(str, 's');
if max(size(strin))
  if isstr(default)
    y = strin;
  else
    y = eval(strin);
  end;
end;
%%%%%%%%%% End crmz_input.m %%%%%%%%%%


%%%%%%%%%% crmz.m (Complex Remez) %%%%%%%%%%
% crmz.m
% Complex Remez: Multiple-exchange Filter Design Algorithm
%      designs FIR filters with arbitary magnitude and
%      phase specifications; reduces to the Remez
%      (Parks-McClellan) algorithm in the real case.
%
% @(#) crmz.m: version 5.0 created on 02/95
% (c) 1995 by Lina Karam and James McClellan
%
% The author can be reached at:
%
% Lina Karam
```

```
% School of Electrical and Computer Engineering
% Goergia Institute of Technology
% Atlanta, GA 30332
% (404) 894-2910
% karam@eedsp.gatech.edu
%main program
disp(' ');
disp('             ***********************************************');
disp('             *****           Complex  Remez            *****');
disp('             ***********************************************');
if ~exist('sym'), sym = 0; end;
if ~exist('N1'), N1 = 0; end;
N2 = N1 + L - 1;   % Last index of impulse response
is_odd = mod(L,2);
if (is_odd)
  Lf = (L+1)/2;
  Ws = ['W(:,2:Lf)'];
  hr_str = ['[a([Lc:-1:2])/2; a(1); a([2:Lc])/2]'];
  hi_str = ['[-a([Lb:-1:(Lc+1)])/2;    0; a([(Lc+1):Lb])/2]'];
  ph_str = ['ones(length(tgrid),1)'];
else
  Lf = L / 2;
  Ws = ['W'];
  hr_str = ['[a([Lc:-1:1])/2; a([1:Lc])/2]'];
  hi_str = ['[-a([Lb:-1:(Lc+1)])/2; a([(Lc+1):Lb])/2]'];
  ph_str = ['exp(-J*pi*tgrid)'];
end;
Lc = Lf;
Ls = L - Lf;
if (~sym)
  Lb = L;
  M_str = ['[cos(W)  sin(' Ws ')]'];
  h_str = [hr_str '+J*' hi_str];
  HH_str = ['HH = fftshift( fft(hc,Ng) ).*' ph_str ';'];
elseif (sym == 1)
  Lb = Lc;
  M_str = ['[cos(W)]'];
  h_str = [hr_str];
  HH_str = ['HH=fft(hc,Ng); HH((Ng/2+2):Ng)=[]; HH=HH.* ' ph_str ';'];
elseif (sym == 2)
  Lb = Ls;
  Lc = 0;
```

```
  M_str = ['[sin(' Ws ')]'];
  h_str = ['J*' hi_str];
  HH_str = ['HH=fft(hc,Ng); HH((Ng/2+2):Ng)=[]; HH=HH.* ' ph_str ';'];
end;
flag = ~sym;
A = Des .* exp(J*2*pi*grd*(N1+N2)/2);
clf;
plot(grd,abs(Des),'w--')
hold on
plot(grd, angle(Des),'w-')
hold off
title('Desired Filter Frequency Response')
xlabel('normalized frequency')
ylabel('magnitude (dashed) and phase (solid)')
disp('Press <return> to start ...')
pause
Ng = round(1.0/delf);
ifgrid = round((grd+0.5*flag)/delf) + 1;
if (max(ifgrid) > Ng)
  ifgrid(length(ifgrid)) = Ng;
end;
disp('Calculating initial solution .....')
[fext, iext] = crmz_guess(edges, grd, Lb);
it = 0; delta = 0.0; delta_old = -1;
%while ( abs(delta) > delta_old )
no_stp = 1;
while (no_stp)
  it = it + 1;
  delta_old = abs(delta);
  fext = grd(iext);
  W = 2*pi*fext*([0:(Lf-1)]+(~is_odd)*0.5);
  Mb = eval(M_str);
  M = [ Mb ((-1).^[0:Lb]')./WT(iext) ];
  a = M \ A(iext);
  delta = a(Lb+1);
  h = eval(h_str);
  hc = crmz_rotate(zeropad(h,Ng-L), -Ls);
  eval(HH_str);
  W = 2*pi*vec_edges*([0:(Lf-1)]+(~is_odd)*0.5);
  Mb = eval(M_str);
  HH(indx_edges) = Mb * a(1:Lb);
  EE = WT .* (A - HH(ifgrid));
```

```
  EE(iext) = delta*( (-1) .^ [2:length(iext)+1] );
  [jext,EEj] = crmz_find(EE,iext);
  if (prg_dspl < 2)
    fprintf('(%3.0f): ', it)
    fprintf('delta = %8.5f+j%8.5f, ', real(delta), imag(delta))
    fprintf('|delta| = %5.2e, ', abs(delta));
    fprintf('e_max = %5.2e\n',max(abs(EE)));
     if (~prg_dspl)
      crmz_plerror;
      keyboard
    end;
  end;
  if (all(iext==jext'))
    no_stp = 0;
  end;
  iext = jext';
end;  %%% end of Complex Remez algorithm %%%%
%%%%% Assessing Optimality of the Solution %%%%
e_max = max(abs(EE));
tlr = abs(delta)/100;  %% Needed due to limited machine accuracy %%
if (e_max <= (abs(delta)+tlr))
  not_optimal = 0;
  fprintf('Optimal Solution Obtained !\n')
  if (prg_dspl == 2)
    fprintf('(%3.0f): ', it)
    fprintf('delta = %8.5f+j%8.5f, ', real(delta), imag(delta))
    fprintf('|delta| = %5.2e, ', abs(delta));
    fprintf('e_max = %5.2e\n',max(abs(EE)));
  end;
else
  crmz_chckopt;
  not_optimal = 1;
  fprintf('(%3.0f): ', it)
  fprintf('delta = %8.5f+j%8.5f, ', real(delta), imag(delta))
  fprintf('|delta| = %5.2e, ', abs(delta));
  fprintf('e_max = %5.2e\n',max(abs(EE)));
end;
%%%%%%%%%%%%  End Complex Remez Stage  %%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%% adesc.m (Second Stage) %%%%%%%%%%
% adesc.m
```

```
% Ascent-descent algorithm for FIR Filter Design
% Solve problem on sets of extremal points using a
% descent method based on the work of Demjanov & Malozemov
% and Wolfe.
%
% @(#) adesc.m: version 5.0 created on 02/95
% (c) 1995 by Lina Karam and James McClellan
%
%main program
adesc_flg=crmz_input('Press <return> to continue and 0 to stop: ',1);
if (adesc_flg)
  disp('           ********************************************');
  disp('           *****    Invoking Second Ascent  Stage     ****');
  disp('           ********************************************');

% Set the variable "bands".
 bands = zeros(size(vec_edges));
 for k = 1:length(vec_edges)
     if vec_edges(k)~=1
       bands(k) = find(vec_edges(k)==grd);
     else
       bands(k) = length(grd);
     end
 end
 J = sqrt(-1);
 no_stp = 1;
 a = a(1:Lb);
 a = a(:);
 n2 = length(a);
 n = 2*length(a);
 mxl = 2*L+1;
 maj_it = 0;
 alpha = 1;
 nu = 0;
 %%%%%%%%%% Set initial accuracy parameters %%%%%%%%%%%%
 acc_scale = max(abs(WT.*A));
 acc_min = 0.012;         %%% ACCURACY: USER SPECIFIED %%%
 acc = acc_min;
 if (acc < acc_min)
   acc = acc_min;
 end;
 r_min = 0.5;
```

```
r_o = acc_scale * r_min;
if (r_o < r_min)
  r_o = r_min;
end;
r = r_o;
epsi_min = 0.012;
epsi_o = acc_scale * epsi_min;
epsil = epsi_o;
epsilon = 1-0.005;
%%%%%%%%%% Construct initial subset %%%%%%%%%%%%%%%%%%%%
HH_o = HH;
e_max = max(abs(EE));
[iext] = adesc_findset(EE,iext,bands,delta);
sub_EE = EE(iext);
sub_grd = grd(iext);
sub_WT = WT(iext);
sub_max = max(abs(sub_EE));
fmax = iext;
[jext] = adesc_findextr(EE,bands,epsilon);
%%%%%%%%%% Check optimality of initial approximation %%%%%
gext = adesc_grad(EE, jext, grd, WT, M_str, Lf, is_odd);
[rext] = adesc_minpolytope(gext);
no_stp = (norm(rext) > acc);
%%%%%%%%%% Begin major iteration %%%%%%%%%%%%%%%%%%%%%%%%
while (no_stp)
    jext = find((max(abs(sub_EE))-abs(sub_EE)) <= epsil);
    G = adesc_grad(sub_EE, jext, sub_grd, sub_WT, M_str, Lf, is_odd);
    [NrG] = adesc_minpolytope(G);
    norm_NrG = norm(NrG);
    if (norm_NrG <= r)
     nu = nu + 1;
     epsil = epsi_o./(2^nu);
     r = r_o./(2^nu);
     f_extr = find(abs(sub_EE)/sub_max >= epsilon);
     gext = adesc_grad(sub_EE,f_extr,sub_grd,sub_WT,M_str,Lf,is_odd);
     [rext] = adesc_minpolytope(gext);
     if (norm(rext) < acc)
       maj_it = maj_it + 1;
       nu = 0;
       epsil = epsi_o;
       r = r_o;
       fprintf('iter: %3.0f   ',maj_it)
```

```
        fprintf('peak wgt error:  %5.2e  ',e_max)
        fprintf('subset peak wgt err: %5.2e\n',sub_max)
        [iext] = adesc_findset(EE,iext,bands,sub_max);
        sub_EE = EE(iext);
        sub_grd = grd(iext);
        sub_WT = WT(iext);
        sub_max = max(abs(sub_EE));
        if (length(iext) == length(fmax))
          no_stp = ~(all(iext==fmax));
        end;
        fmax = iext;
      end;
     else
      d = - NrG./norm_NrG;
      d2 = d(1:n2)+J*d(n2+1:n);
      [HH_d] = adesc_reconst(d2,h_str,HH_str,M_str,tgrid,Ng,...
               L,Lf,Lc,Ls,Lb,is_odd,vec_edges,indx_edges);
      [HH_n,EE,e_max,alpha] = adesc_linsearch(alpha,iext,HH_o,HH_d,...
                              sub_max,A,WT,ifgrid);
      a = a + alpha*d2;
      HH_o = HH_n;
      sub_EE = EE(iext);
      sub_max = max(abs(sub_EE));
     end;             %%% if %%%
 end;                 %%% while no_stp %%%
 %%%%%%%%%%   end major iteration %%%%%%%%%%%%%%%%%%%%%%%%%
 HH = HH_n;
 a = a(:);
 h = eval(h_str);
 epsilon = 0.9*sub_max/e_max;
 iext = adesc_findextr(EE,bands,epsilon);
 jext = iext';
 delta = max(abs(EE));
 fprintf('Optimal Solution Obtained !\n')
end;       %%%%%%% end if %%%%%%%
%%%%%%%%%%   End Ascent-Descent Stage  %%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%% crmz_guess.m %%%%%%%%%%
function  [fext,iext] = crmz_guess( edges, grid, nfcns )
% REMGRID   generate initial guess of "EXTREMAL FREQS"
%            for remez exchange algorithm [jMc 11/92]
```

```
%   usage:
%        [Fext,Iext] = rmz_guess( Edges, Grid, Nfcns )
%
%     Edges:  band-edges moved onto the grid (Revised edges)
%     Grid:   frequencies on the fine grid
%     Nfcns:  number of basis functions in the approx
%     Fext:   initial guess of "extremal frequencies"
%     Iext:   indices for the "extremal frequencies"
%
TOL = 5*eps;
next = nfcns+1;
Nbands = length(edges(:,2));
tt = edges;
merged = tt;
if( Nbands > 1 )
  jkl = find( abs(tt(1:(Nbands-1),2)-tt(2:Nbands,1)) > TOL );
  merged = [ [tt(1,1); tt(jkl+1,1)] , [tt(jkl,2); tt(Nbands,2)] ];
end
Nbands = length(merged(:,1));
bw = merged(:,2)-merged(:,1);
if( any(bw<0) )
   edges
   error('REM_GUESS: negative bandwidth!')
end
percent_bw = bw / sum(bw);
fext = zeros(next,1);
n = 0;  i = 1;
while( n<next & i<=Nbands )
   nfreqs_i = min( next-n, ceil( percent_bw(i)*next ) );
   if( nfreqs_i == 0 )
      n=n+1;   fext(n) = merged(i,1);
   else
      fext(n+[1:nfreqs_i]) = ...
           linspace(merged(i,1),merged(i,2),nfreqs_i);
      n = n+nfreqs_i;
   end
   i = i+1;
end
iext = 0*fext;
for i=1:next
   [tt,iext(i)] = min( abs(fext(i)-grid) );
end
```

```
if( any(diff(iext)==0) )
   error('RMZ_GUESS: two extremal freqs at same grid pt, NOT!')
end
fext = grid(iext);
%%%%%%%%%% End crmz_guess.m %%%%%%%%%%


%%%%%%%%%% crmz_grid.m %%%%%%%%%%
function  [grid,del_f,vec_edges,inedges] = crmz_grid(edges, L, whole)
% REMGRID    generate grid for remez exchange  [jMc 11/92] [lk 05/93]
% Version 3.0
%   usage:
%        [Grid,Delta_f,Rev_edges] = remgrid( Edges, L, Whole )
%
%     grid:       frequencies on the fine grid
%     del_f:      spacing between grid pts
%     vec_edges:  specified edges as a vector
%     inedges:    index of specified band-edges on grid
%     edges:      specified band-edges, [Nbands X 2] array
%     L:          filter Length
%     Whole:      domain for approximation: default is [0,0.5]
%                 'whole' means [-0.5,0.5),
%
if( nargin < 3 )
   whole = 'half ';
end
mf = 25;
Ngrid = 2^(nextpow2( mf*L ));
del_f = 1.0/Ngrid;
vec_edges = edges';
vec_edges = vec_edges(:);
if ( all(whole == 'whole') )
  grid = [-0.5:del_f:0.5-del_f];
  inedges = round((vec_edges+0.5)/del_f)+1;
elseif( all(whole == 'half ') )
  grid = [0:del_f:0.5];
  inedges = round(vec_edges/del_f)+1;
else
   error('RMZ_GRID: whole is neither half nor whole')
end
if (inedges(length(inedges)) > length(grid))
  inedges(length(inedges)) = length(grid);
```

```
end;
Ng = length(grid);
grid(inedges) = vec_edges;
%%%%%%%%%% End crmz_grid.m %%%%%%%%%%


%%%%%%%%%% crmz_rotate.m %%%%%%%%%%
function rotated = rotate(x,num_places)
%ROTATE                                                  [jMc 2/89]
%    rotate(V,r) circularly shifts the elements in the columns of V
%        by r places right (r>0); or r places left (r<0).
%        (Right is down; left is up.)
%        If the input is a row or column vector, the shift is
%        performed on the vector.
%        If the input is a signal matrix, each column is shifted
%  see also SHIFTM, SHIFT, ZEROPAD
[M,N] = size(x);
if M > 1              % ------- rotate columns ----------------
   num_places = mod(num_places,M);  % make num_places in range [0,M-1]
   rotated = [ x(M-num_places+1:M,:); x(1:M-num_places,:) ];
elseif N > 1          % ------- rotate row vector -------------
   num_places = mod(num_places,N);  % make num_places in range [0,N-1]
   rotated = [ x(N-num_places+1:N) x(1:N-num_places) ];
end
%%%%%%%%%% End crmz_rotate.m %%%%%%%%%%


%%%%%%%%%% crmz_find.m %%%%%%%%%%
function  [fnew,enew] = crmz_find( error, fold )
%  Construct new subset of points using the exchange
%  rules derived in Chapter 3 of Thesis. (see also
%  "Complex Chebyshev approximation for FIR filter design"
%  by Lina J. Karam and James H. McClellan in the IEEE
%  Trans. on Circuits and Systems II - March 1995.
%
fold = fold(:);
Nx = length( fold );
Ngrid = length( error );
sgn_error = error(fold(1))/abs(error(fold(1)));
error = real( conj(sgn_error)*error );
delta = min( abs( error(fold) ) ); %--- present value of delta
up = sign(error(fold(1))) > 0;
```

```
if (up)
  fence = [ [1;fold(1:2:Nx)]  [fold(1:2:Nx);Ngrid] ];
else
  fence = [ [1;fold(2:2:Nx)]  [fold(2:2:Nx);Ngrid] ];
end;
Lf = length(fence(:,1));
emn = zeros(Lf,1); imn = emn; emx = emn; imx = emn;
for i=1:Lf
   [emn(i),imn(i)] = min( error(fence(i,1):fence(i,2) ));
   imn(i) = imn(i) + fence(i,1) -1;
end;
imn(find((emn > -delta))) = [];
fence = [ [1;imn]  [imn;Ngrid] ];
 Lf = length(fence(:,1));
for i=1:Lf
   [emx(i),imx(i)] = max( error(fence(i,1):fence(i,2) ));
   imx(i) = imx(i) + fence(i,1) -1;
end;
imx(find((emx < delta))) = [];
fnew = sort([imx;imn]);
Nf = length(fnew);
if ( Nf > Nx)
 if ( abs(error(fnew(1))) >= abs(error(fnew(Nf))) )
   fnew = fnew(1:Nx);
 else
   fnew = fnew(Nf-Nx+1:Nf);
 end;
end;
fnew = fnew';
%%%%%%%%%% End crmz_find.m %%%%%%%%%%


%%%%%%%%%% crmz_plerror.m %%%%%%%%%%
% plot error magnitude, real phase rotated error,
%      error real part, error imaginary part.
%
clf;
EE1 = EE(iext(1));
EE_pl = []; grd_pl = []; i = 1; l = 0;
while (i < (length(indx_edges)-1))
  lo = l + 1;
  l = l + length([indx_edges(i):indx_edges(i+1)]);
```

```
  grd_pl = [grd_pl;grd(lo:l);nan];
  EE_pl = [EE_pl;EE(lo:l);nan];
  i = i + 2;
end;
lo = l + 1;
l = l + length([indx_edges(i):indx_edges(i+1)]);
grd_pl = [grd_pl;grd(lo:l)];
EE_pl = [EE_pl;EE(lo:l)];
EEE_pl = [real(EE_pl * conj(EE1/abs(EE1)) )-3*abs(delta),...
abs(EE_pl), real(EE_pl)-6*abs(delta),...
imag(EE_pl)-9*abs(delta)];  %-- plottting offset
EEE = [real(EE * conj(EE1/abs(EE1)) )-3*abs(delta),...
abs(EE), real(EE)-6*abs(delta),...
imag(EE)-9*abs(delta)];  %-- plottting offset
delr = real(delta);   deli = imag(delta);
axis('auto')
plot( grd_pl, EEE_pl, '-', [grd(1), 0.5], abs(delta)*[1;1], 'w--',...
 [grd(1), 0.5], abs(delta)*[1 -1;1 -1]-3*abs(delta), 'w:',...
 [grd(1), 0.5], delr*[1 -1;1 -1]-6*abs(delta),'w:',...
 [grd(1), 0.5], deli*[1 -1;1 -1]-9*abs(delta),'w:',...
 grd(jext), EEE(jext,:), 'o', grd(iext), EEE(iext,:), '+')
V = axis;
axis([-0.5*flag 0.5 V(3) V(4)])
xlabel('normalized frequency')
title('Weighted Error: 1.magnitude  2.real-rotated  3.real part '...
'4.imag. part')
%%%%%%%%%% End crmz_plerror.m %%%%%%%%%%


%%%%%%%%%% crmz_chckopt.m %%%%%%%%%%
% Checks degree of suboptimality of solution not optimal
% on desired frequency bands.
%
subint = find( abs(EE) <= (abs(delta)+tlr) );
rel_size = round((length(subint)/length(grd))*100);
disp([sprintf('Optimal solution found on a frequency subset ') ...
sprintf('comprising %3.0f percent \n of the original ',rel_size) ...
sprintf('frequency set (see Error plots)!')])
%%%%%%%%%% End crmz_chckopt.m %%%%%%%%%%


%%%%%%%%%%  adesc_findset.m  %%%%%%%%%%
```

```
function [fmax]=thesis_findset(error,iext,indx_edges,delta)
%
% Construct new set fmax of frequency points.
% fmax: All local maxima including the set "iext".
%       Points with small deviation removed.
%
error = error(:);
indx_edges = indx_edges(:);
Ngrid = length(error);
abs_e = abs(error);
abs_e = [abs_e(2); abs_e; abs_e(Ngrid-1)];
fmax = find( (abs_e(2:Ngrid+1) >= abs_e(1:Ngrid)) & ...
             (abs_e(2:Ngrid+1) >= abs_e(3:Ngrid+2)) );
fmax = fmax(:);
fmax = sort([fmax; indx_edges; iext]);
abs_e([1;Ngrid+2]) = [];
fmax(find(~(fmax(1:length(fmax)-1)-fmax(2:length(fmax)))))=[];
fmax(find(abs_e(fmax) < 0.9*abs(delta))) = [];
%%%%%%%%%% End adesc_findset.m %%%%%%%%%%


%%%%%%%%%% adesc_findextr.m %%%%%%%%%%
function [fmax]=adesc_findextr(error,indx_edges,epsilon)
%
% returns indices of the extremals in "fmax".
%
error = error(:);
indx_edges = indx_edges(:);
Ngrid = length(error);
abs_e = abs(error);
abs_e = [abs_e(2); abs_e; abs_e(Ngrid-1)];
fmax = find( (abs_e(2:Ngrid+1) >= abs_e(1:Ngrid)) & ...
             (abs_e(2:Ngrid+1) > abs_e(3:Ngrid+2)) );
fmax = fmax(:);
fmax = sort([fmax; indx_edges]);
abs_e([1;Ngrid+2]) = [];
fmax(find(~(fmax(1:length(fmax)-1)-fmax(2:length(fmax)))))=[];
if (length(fmax) > 1)
  fmax(find(abs_e(fmax)/max(abs_e) < epsilon)) = [];
end;
%%%%%%%%%% End adesc_findextr.m %%%%%%%%%%
```

```
%%%%%%%%%% adesc_grad.m %%%%%%%%%%
function [G] = adesc_grad(EE, iext, grd, WT, M_str, Lf, is_odd)
%
%  Calculates gradients at the points iext
%
G = [];
J = sqrt(-1);
fext = grd(iext);
W = 2*pi*fext*([0:(Lf-1)]+(~is_odd)*0.5);
Mb = eval(M_str);
Mb = -Mb';
MWT = diag(2.*(WT(iext).*EE(iext)));
M = Mb*MWT;
G = [real(M);imag(M)];
%%%%%%%%%% End adesc_grad.m %%%%%%%%%%


%%%%%%%%%% adesc_minpolytope.m %%%%%%%%%%
function [Ptmin] = crmz_minpolytope(P);
% Finds the nearest point Ptmin in the convex hull of
% the set of points P1,...,PM given by the column of
% the N*M matrix P.
% Implementation of a method by P. Wolfe presented in
% "Mathematical Programming", vol. 11, 1976, pp. 128-149.
%
S = []; w = []; no_stp = 1;
[N,M] = size(P);
if (M==1)
  no_stp = 0;
  X = P;
end;
%Z1 = 10^(-12);
Z1 = 10^(-10);
Z2 = 10^(-10);
Z3 = 10^(-10);
P_norm = sum(abs(P).^2);
[pmn,J] = min(P_norm);
S = [S;J];
w = [w;1];
while (no_stp)
  X = P(:,S)*w;
```

```
  [pmn,J] = min(X'*P);
  PS_norm = max(sum(abs(P(:,S)).^2));
  PJ_norm = P(:,J)'*P(:,J);
  no_stp = (X'*P(:,J) <= (X'*X - Z1*max([PJ_norm;PS_norm])));
 if (no_stp)
    I = find(~(S-J));
    if (length(I) > 0)
      no_stp = 0;
      disp('temporary disaster: see Note 3')
    end;
 end;
 if (no_stp)
  S = [S;J];
  w = [w;0];
  flg = 1;
  while (flg)
    e = ones(size(S));
    A = ones(length(e),length(e)) + P(:,S)'*P(:,S);
    u = A \ e;
    v = u./(e'*u);
    I = find(v <= Z2);
    if ~(length(I))
      w = v;
      flg = 0;
    else
      I = find((w-v) > Z3);
      t = w(I)./(w(I)-v(I));
      theta = min([1;1-min(t)]);
      w = theta*w + (1-theta)*v;
      I = find(w <= Z2);
      if (length(I) > 0)
        w(I) = zeros(length(I),1);
        w(I(1)) = [];
        S(I(1)) = [];
      end;
    end;
  end;
 end;
end;
Ptmin = X;
%%%%%%%%% End adesc_minpolytope.m %%%%%%%%%
```

```
%%%%%%%%%% adesc_reconst.m %%%%%%%%%%
function [HH] = adesc_reconst(at,h_str,HH_str,M_str,tgrid,Ng,...
                L, Lf, Lc, Ls, Lb,is_odd, v_edges, in_edges)
%  Reconstructs the frequency response "HH" from the basis
%  coefficients "at".
%
J = sqrt(-1);
a = at(:);
h = eval(h_str);
hc = crmz_rotate(zeropad(h,Ng-L), -Ls);
eval(HH_str);
W = 2*pi*v_edges*([0:(Lf-1)]+(~is_odd)*0.5);
Mb = eval(M_str);
HH(in_edges) = Mb * a;
%%%%%%%%%% End adesc_reconst.m %%%%%%%%%%


%%%%%%%%%% adesc_linsearch.m %%%%%%%%%%
function [HHt,EEt,emxt,t] = adesc_linsearch(t0,iext,HH_o,d,emx,...
                                            DD,WT,ifgrid)
%  Performs a simple line search for step size.
%  Efficiency of search can be improved by implementing
%  method suggested in book "Introduction To Minimax"
%  by V. F. Demjanov and V. N. Malozemov, pp. 109-112.
%
t = t0; no_stp = 1; HH_o = HH_o(:); d = d(:);
r_flg = 0; t_flg = 1; h_flg = 0; i_flg = 0;
grtr = 1; c = 2; acc = 1 -  10^(-0.1/10);
while (grtr)
   HHt = HH_o + t*d;
   EEt = WT .* (DD - HHt(ifgrid));
   emxt = max(abs(EEt(iext)));
   if (emxt <= emx)
     grtr = 0;
   else
     t = t/2;
   end;
end;
tmin = t; HHt_min = HHt; EEt_min = EEt;
emin = emxt; I = 2*t*ones(2,1);
while (no_stp)
```

```
  if (~i_flg) t = c*t; end;
  HHt = HH_o + t*d;
  EEt = WT .* (DD - HHt(ifgrid));
  emxt = max(abs(EEt(iext)));
  R = (emxt <= emin);
  if (R)
    tmin = t; HHt_min = HHt; EEt_min = EEt; emin = emxt;
    t_flg = 0; h_flg = 0;
    if (i_flg)
      I = sort([t;I(2)]);
      t = (t+I(2))/2;
    end;
    r_flg = 1;
  elseif ((~R) & (i_flg))
      I = sort([I(1);t]);
      t = (t+I(1))/2;
  elseif (t_flg)
    t1 = t;
    t_flg = 0;
  elseif (r_flg)
    if (r_flg) t1 = t; end;
    I = sort([tmin,t1]);
    t = (tmin+t1)/2;
    i_flg = 1;
  else
    no_stp = 0;
  end;
  if  ( i_flg & ((I(2) - t) < acc*I(2)) ) no_stp = 0; end;
end;
HHt = HHt_min; EEt = EEt_min; emxt = max(abs(EEt)); t = tmin;
%%%%%%%%%% End adesc_linsearch.m %%%%%%%%%%


%%%%%%%%%% crmz_plresult.m %%%%%%%%%%
% crmz_plresult
% Version 5.0
%            [lk, 01/95]
% Generates subplots of final result with
%    1. Magnitude response
%    2. Passband Phase
%    3. Error magnitude and Real rotated error
%    4. Phase error in passband
```

```
% Uses grp_delay.m and crmz_plerror2.m
%
clf;
subplot(111)
axis('auto');
subplot(221)
dbMax = 20*log(max(abs(H(:))));
%dbMin = 20*log(min(abs(H(:))));
%dbRange = dbMax - dbMin;
dbRange = 80;
plot(tgrid, db(abs(H(:)),dbRange,dbMax), '-')
ph = gca;
set(ph,'box','off');
xlabel('normalized frequency'), ylabel('magnitude (log scale)')
subplot(222);
pas_edges = pas_edges';
pas_edges = pas_edges(:);
i = 1;
grd_pl = [];
aH_pl = [];
aD_pl = [];
while (i < (length(pas_edges)-1))
   psb = find((grd >= pas_edges(i)) & (grd <= pas_edges(i+1)));
   grd_pl = [grd_pl;grd(psb);nan];
   aH_pl = [aH_pl;angle(H(ifgrid((psb))));nan];
   aD_pl = [aD_pl;angle(Des(psb));nan];
   i = i + 2;
end;
psb = find((grd >= pas_edges(i)) & (grd <= pas_edges(i+1)));
grd_pl = [grd_pl;grd(psb)];
aH_pl = [aH_pl;angle(H(ifgrid((psb))))];
aD_pl = [aD_pl;angle(Des(psb))];
plot(grd_pl, aH_pl, '-')
hold on
plot(grd_pl, aD_pl, 'w--')
ph = gca;
set(ph,'box','off');
xlabel('normalized frequency'), ylabel('passband phase (radians)')
V = axis;
V(1) = pas_edges(1);
V(2) = pas_edges(length(pas_edges));
axis([V(1) V(2) V(3) V(4)]);
```

```
x = (V(1)+V(2))/(2.0);
text(x,V(4),'( -- ideal )')
hold off
subplot(224)
e_angle = mod(aD_pl,2*pi)-mod(aH_pl,2*pi);
plot(grd_pl,e_angle)
xlabel('normalized frequency'), ylabel('passband phase error')
ph = gca;
set(ph,'box','off');
V = axis;
V(1) = pas_edges(1);
V(2) = pas_edges(length(pas_edges));
axis([V(1) V(2) V(3) V(4)]);
subplot(223)
crmz_plerror2;
ph = gca;
set(ph,'box','off');
axis('off')
text(0.48,-4.5*abs(delta),'0.5')
text(-0.0108,-4.5*abs(delta),'0')
if (~sym)
 text(-0.58,-4.5*abs(delta),'-0.5')
 str = sprintf('%5.4f',abs(delta));
 text(-0.75,abs(delta),str,'fontsize',10);
 text(-0.75,-2*abs(delta),str,'fontsize',10);
 str = sprintf('%5.4f',-abs(delta));
 text(-0.78,-4*abs(delta),str,'fontsize',10);
 text(-0.22,1.6*abs(delta),'Error magnitude','fontsize',10);
 text(-0.25,-1.5*abs(delta),'Real rotated error','fontsize',10);
else
 str = sprintf('%5.4f',abs(delta));
 text(-0.14,abs(delta),str,'fontsize',10);
 text(-0.14,-2*abs(delta),str,'fontsize',10);
 str = sprintf('%5.4f',-abs(delta));
 text(-0.16,-4*abs(delta),str,'fontsize',10);
 text(0.155,1.6*abs(delta),'Error magnitude','fontsize',10);
 text(0.125,-1.5*abs(delta),'Real rotated error','fontsize',10);
end
xlabel('normalized frequency')
%%%%%%%%%% End crmz_plresult.m %%%%%%%%%%
```

```
%%%%%%%%%% crmz_plerror2.m %%%%%%%%%%
% plot error magnitude and real phase-rotated error.
%
EE1 = EE(iext(1));
EE_pl = [];
grd_pl = [];
i = 1; l = 0;
while (i < (length(indx_edges)-1))
  lo = l + 1;
  l = l + length([indx_edges(i):indx_edges(i+1)]);
  grd_pl = [grd_pl;grd(lo:l);nan];
  EE_pl = [EE_pl;EE(lo:l);nan];
  i = i + 2;
end;
lo = l + 1;
l = l + length([indx_edges(i):indx_edges(i+1)]);
grd_pl = [grd_pl;grd(lo:l)];
EE_pl = [EE_pl;EE(lo:l)];
EEE_pl = [real(EE_pl * conj(EE1/abs(EE1)) )-3*abs(delta),abs(EE_pl)];
EEE = [real(EE * conj(EE1/abs(EE1)) )-3*abs(delta), abs(EE)];
delr = real(delta);   deli = imag(delta);
axis('auto')
plot( grd_pl, EEE_pl, '-', [grd(1), 0.5], abs(delta)*[1;1], 'w--',...
 [grd(1), 0.5], abs(delta)*[1 -1;1 -1]-3*abs(delta), 'w:',...
  grd(jext), EEE(jext,:), 'o', grd(iext), EEE(iext,:), '+')
V = axis;
axis([-0.5*flag 0.5 V(3) V(4)])
%%%%%%%%%% End crmz_plerror2.m %%%%%%%%%%
```

D.2 Family Design Procedure (Chapter 5)

```
%%%%%% RUNNING INSTRUCTIONS:
%%%%%%     1) Run Matlab.
%%%%%%     2) Specify desired cutoff frequency ``wc'' in
%%%%%%        the file ``getfromSet.m''
%%%%%%     3) Copy the coefficients of family subfilter
%%%%%%        {tqr} (see Chapter 5 of Thesis) into t.mat.
%%%%%%        Alternatively, one can specify the impulse
```

```
%%%%%%        response ``f'' of the quadrant-symmetric
%%%%%%        subfilter in the first quadrant only (2-D matrix).
%%%%%%        In this latter case, comment out the commands
%%%%%%                 load t;
%%%%%%        and
%%%%%%                 f = tTof(t);
%%%%%%        from the file ``getfromSet.m'' listed below.
%%%%%%     3) Copy impulse response of designed prototype
%%%%%%        into a file ``p.mat''. For migration filters,
%%%%%%        the value of the parameter ``b'' (see below
%%%%%%        for explanation on what ``b'' is) need also
%%%%%%        to be saved in p.mat, or be set at the Matlab
%%%%%%        prompt before executing step 4.
%%%%%%     4) At matlab prompt, type
%%%%%%               getfromSet
%%%%%%


%%%%%%%%%% getfromSet.m %%%%%%%%%%
%%%% Computes the coefficients and frequency responses of
%%%% 1-D and 2-D migration or lowpass filters with cutoff
%%%% frequency "wc" by implementing the Chebyshev structures
%%%% described in Chapter 5 of Thesis.
%%
wc = 0.75*pi   % desired cutoff frequency in radian (0 < wc <pi)
Ng = 128;      % grid size for fft calculation
b = [];
load p;        % loads coefficents "p" of prototype filter
               % designed using crmz_adesc and approximating
               %      Pd(w) = e^{j*b*( sqrt((pi)^2 - 2*pi*w) )}
               % for complex migration filters, or a fan filter
               % with cutoff pi/2 for zero-phase lowpass filters.
               % The value "b" is loaded also for migration filters.
               %%%
load t;        % Matrix consisting of the coeffcients {tqr} of the
               % transformation subfilter, computed to best
               % approximate the ideal contours of the desired
               % family, for a given transformation order (Q by R).
               %%%
f = tTof(t);
%%% Compute coefficients of 1-D filter
h1 = oneDfromSet(wc,p,f);
```

```
%%% Compute coefficients of 2-D filter
fcs = [ 1 2 1; 2 -4 2; 1 2 1]./8; % McClellan transformation
                                  % for circularly-sym filters.
h2 = twoDfromSet(wc,p,f,fcs);
%%%% Generate normalized frequency grid for plotting %%%%
del_f = 1.0/Ng;
freq = [-0.5:del_f:0.5];
freq(Ng+1) = [];
%%%% Find passband in grid corresponding to [0,wc].
pindx = find( (freq < 0.98*(wc/(2*pi))) & (freq >=0));
fpas = freq(pindx);
pos = find(freq >= 0);
fpos = freq(pos);
%%%% Plot magnitude of 1-D frequency response %%%%
H = oneDfft(h1,Ng);
fc = wc/(2*pi);
dbmax = 20*log(max(abs(H(pos))));
plot(fpos, db(abs(H(pos)),45,dbmax),'-')
ph = gca; set(ph,'box','off'); set(ph,'Fontsize',15)
xlabel('normalized frequency'), ylabel('magnitude (log scale)')
ph = gca; set(ph,'box','off');
keyboard
if (length(b) > 0)
   J = sqrt(-1);
   D = exp(J*b*sqrt((wc).^2-(2*pi*fpas).^2)); D = D(:);
   phas_e = mod(angle(D),2*pi) - mod(angle(H(pindx)),2*pi);
   plot(fpas, angle(H(pindx)),'-')
   ph = gca; set(ph,'box','off');
   set(ph,'Fontsize',15)
   xlabel('normalized frequency'), ylabel('passband phase (radians)')
   hold on
   plot(fpas(:), angle(D),'--')
   V = axis; V(3) = min([min(angle(H(pindx))) min(angle(D))]);
   axis(V)
   v1 = (V(1)+V(2))/2+0.1*(V(2)-V(1))/2;
   v2 = max(angle(H(pindx)))+(V(4)-max(angle(H(pindx))))/2;
   str = sprintf('deviation %5.4f',max(abs(phas_e)));
   text(v1,v2,str,'fontsize',15)
   hold off
end
%%%% Plot magnitude of 2-D frequency response %%%%
H = twoDfft(h2,Ng);
```

```
[Fx,Fy] = meshgrid(freq);
clf
mesh(Fx,Fy,abs(H))
ph = gca; set(ph,'Fontsize',15)
zlabel('magnitude (linear scale)')
if (length(b) > 0)
 keyboard
 I = find((freq <= (wc/(2*pi))) & (freq >= - (wc/(2*pi))));
 mesh(Fx(I,I),Fy(I,I), angle(H(I,I)))
 ph = gca;
 set(ph,'Fontsize',15)
 zlabel('phase in passband (radians)')
end
%%%%%%%%%% End getfromSet.m %%%%%%%%%%


%%%%%%%%%% tTof.m %%%%%%%%%%
function f = tTof(t)
%%%                [LK, 01/23/95]
%% Converts the coefficients {tqr} associated with the
%% basis functions {cos(q w1)*cos(r w2)} into the
%% impulse response of the corresponding quadrant
%% symmetric subfilter.
%% f = matrix containing the impulse coefficients of
%%     the quadrant symmetric family transformation
%%     function (determined from {tqr} in Chapter 5 of
%%     Thesis). f contains only the first quadrant
%%     coeffcients and is of size (Q+1) by (R+1),
%%     where Q and R defines the order of the family
%%     transformation function.
%% t = basis coefficients corresponding to the basis
%%     {cos(q w1)*cos(r w2)} (correspond to {tqr} in
%%     Chapter 5 of thesis). "t" is of size (Q+1) by
%%     (R+1) where Q and R are the orders of the
%%     transformation function "f".
%%
[M,N] = size(t);
Q = size(t,1)-1;
R = size(t,2)-1;
f = zeros(Q+1,R+1);
f(1,1) = t(1,1);
f(1,2:R+1) = 0.5 * t(1,2:R+1);
```

```
f(2:Q+1,1) = 0.5 * t(2:Q+1,1);
f(2:Q+1,2:R+1) = 0.25 * t(2:Q+1,2:R+1);
%%%%%%%%%% End tTof.m %%%%%%%%%%


%%%%%%%%%% oneDfromSet.m %%%%%%%%%%
function h = oneDfromSet(wc,p,f)
%%%                              [LK, 01/23/95]
%% Retrieves the 1-D filter with cutoff "wc" from the
%% set (family) of variable-cutoff (complex or real) filters,
%% which is generated by transforming
%% the prototype filter with impulse "p" using the
%% quadrant symmetric transformation function with
%% impulse "f". Only the coefficients of "p" and "f"
%% are needed to retrieve and realize any 1-D filter
%% in the set.
%% h = impulse of 1-D filter with cutoff "wc" (noncausal and
%%     symmetric with respect to the center point).
%% wc = cutoff frequency of filter "h".
%% p = impulse response of prototype filter computed using
%%     crmz.m (Note that for migration filters "p" depends
%%     on the parameter "b").
%% f = matrix containing the impulse coefficients of
%%     the quadrant symmetric family transformation
%%     function (determined from {tqr} in Chapter 5 of
%%     Thesis - see the function tTof.m). f contains
%%     only the first quadrant coeffcients and is of size
%%     (Q+1) by (R+1), where Q and R defines the order of
%%     the family transformation function.
%%
m = size(f) - 1;
Q = m(1);
R = m(2);
%% Converts "f" to the coefficients "{tqr}" which correspond to
%% the coeffients of the basis functions "{cos(q w1)*cos(r w2)}".
%%
t = zeros(Q+1,R+1);
t(1,1) = f(1,1);
t(1,2:R+1) = 2*f(1,2:R+1);
t(2:Q+1,1) = 2*f(2:Q+1,1);
t(2:Q+1,2:R+1) = 4*f(2:Q+1,2:R+1);
%% Compute impulse "f1" of 1-D transformation subfilter using
```

```
%% "tqr" (matrix t) and the cutoff frequency "wc".
%% "f1" corresponds to the transformation coefficients {f_q}
%% in Chapter 5 of Thesis.
%%
vc = cos([0:R]*wc);
vc = vc(:);
tmp = t * vc;
f1 = [0.5*tmp(Q+1:-1:2); tmp(1); 0.5*tmp(2:Q+1)];
%% Convert "p" to the basis coefficients vector "cp" which is
%% associated with the Chebyshev polynomials and which is used
%% in chebpolyTrans.m.
%%
p = p(:);
p = p.';
L = floor((length(p)-1)/2);
cp = [p(L+1) 2*p(L+2:2*L+1)];
%% Use the Chebyhshev polynomials to compute the coefficients
%% "h" of the 1-D filter with cutoff frequency "wc".
%%
h = chebpolyTrans(cp,f1);
%%%%%%%%%% End of oneDfromSet.m %%%%%%%%%%


%%%%%%%%%% twoDfromSet.m %%%%%%%%%%
function h = twoDfromSet(wc,p,f,f2)
%%%                              [LK, 01/23/95]
%% Retrieves the 2-D filter with cutoff "wc" from the
%% set (family) of variable-cutoff (complex or real) filters.
%% The family of filters is generated by transforming
%% the prototype filter with impulse "p" using the
%% quadrant symmetric transformation function with
%% impulse "f". Only the coefficients of "p", "f", and
%% "f2" are needed to retrieve and realize any 2-D filter
%% in the set.
%% h = impulse of 1-D filter with cutoff "wc" (noncausal and
%%     symmetric with respect to the center point).
%% wc = cutoff frequency of filter "h".
%% p = impulse response of prototype filter computed using
%%     crmz.m (Note that for migration filters "p" depends
%%     on the parameter "b").
%% f = matrix containing the impulse coefficients of
%%     the quadrant symmetric family transformation
```

```
%%     function (determined from {tqr} in Chapter 5 of
%%     Thesis - see the function tTof.m). f contains
%%     only the first quadrant coeffcients and is of size
%%     (Q+1) by (R+1), where Q and R defines the order of
%%     the family transformation function.
%% f2 = matrix containing the impulse coefficients of
%%      a transformation function (corresponds to {Fcs}
%%      in Chapter 5 of Thesis) for generating
%%      circularly symmetric (or other) filters.
%%
m = size(f) - 1;
Q = m(1);
R = m(2);
%% Converts "f" to the coefficients "{tqr}" which correspond to
%% the coeffients of the basis functions "{cos(q w1)*cos(r w2)}".
%%
t = zeros(Q+1,R+1);
t(1,1) = f(1,1);
t(1,2:R+1) = 2*f(1,2:R+1);
t(2:Q+1,1) = 2*f(2:Q+1,1);
t(2:Q+1,2:R+1) = 4*f(2:Q+1,2:R+1);
%% Compute impulse "f1" of 1-D transformation subfilter using
%% "tqr" (matrix t) and the cutoff frequency "wc".
%%
vc = cos([0:R]*wc);
vc = vc(:);
tmp = t * vc;
f1 = [0.5*tmp(Q+1:-1:2); tmp(1); 0.5*tmp(2:Q+1)];
%% Use the Chebyshev structure (with "f1" as prototype and "f2" as
%% the transformation subfilter) to compute the coefficients
%% of the 2-D transformation subfilter "fp" which transforms "p"
%% into the desired 2-D filter with cutoff "wc"
%%
f1 = f1(:);
f1 = f1.';
cf1 = [f1(Q+1) 2*f1(Q+2:2*Q+1)];
fp = chebpolyTrans(cf1,f2);
%% Convert "p" to the basis coefficients vector "cp" which is
%% associated with the Chebyshev polynomials and which is used
%% in chebpolyTrans.m.
%%
p = p(:);
```

```
p = p.';
L = floor((length(p)-1)/2);
cp = [p(L+1) 2*p(L+2:2*L+1)];
%% Use the Chebyhshev structure to compute the coefficients
%% "h" of the 2-D filter with cutoff frequency "wc".
%%
h = chebpolyTrans(cp,fp);
%%%%%%%%%% End of twoDfromSet.m %%%%%%%%%%


%%%%%%%%%% chebpolyTrans.m %%%%%%%%%%
function h = chebpolyTrans2(a,f)
%%%%                          [LK, 01/23/95]
%% Computes the impulse response "h" using the Chebyshev
%% polynomials with the basis coefficients "c" and the
%% impulse response "f" of a transformation function.
%% h = impulse response of transformed filter.
%% a = basis coefficients associated with the Chebyshev polynomials;
%%     i.e., \sum(a_{n}T_{n}(f)) where T_n is the nth degree
%%     Chebyshev polynomial.
%% f = impulse of transformation subfilter F and is of size
%%     (2Q+1) by (2R+1).
%%
% McClellan transformation (Default if no argument)
if nargin<2, f = [1 2 1;2 -4 2;1 2 1]/8; end
L = length(a);
if (~L)
  error('The basis coefficients vector "c" is empty')
end
inset = floor((size(f)-1)/2);
Q = inset(1);
R = inset(2);
% Use Chebyshev polynomials to compute "h"
T0 = 1; T1 = f;
if (L > 1)
  h = a(2)*T1;
  rows = Q+1; cols = R+1;
  h(rows,cols) = h(rows,cols)+a(1)*T0;
else
  h = a(1);
end
for i=3:L,
```

```
  T2 = 2*conv2(f,T1);
  rows = rows + Q; cols = cols + R;
  T2(rows,cols) = T2(rows,cols) - T0;
  rows = Q + [1:size(T1,1)];
  cols = R + [1:size(T1,2)];
  hh = h;
  h = a(i)*T2; h(rows,cols) = h(rows,cols) + hh;
  T0 = T1;
  T1 = T2;
end
%%%%%%%%%% End of chebpolyTrans.m %%%%%%%%%%


%%%%%%%%%% oneDfft.m %%%%%%%%%%
function [H] = oneDfft(h, Ng)
%%%                          [LK, 01/23/95]
%% Computes the frequency response of the 1-D filter with
%% impulse "h". With "m" being the size of "h",
%% assumes starting index of impulse is equal to -(m-1)/2
%% if "m" odd, and to -m/2 if "m" even.
%%
m = max(size(h)); is_odd = mod(m,2);
if (is_odd)
  rf = (m-1)/2;
else
  rf = m/2;
end;
hc = zeropad(h,Ng-m);
hc = crmz_rotate(hc,-rf);
H = fftshift(fft(hc));
%%%%%%%%%% End of oneDfft.m %%%%%%%%%%


%%%%%%%%%% indxoneDfft.m %%%%%%%%%%
function [H] = indxoneDfft(h, N1, Ng)
%%%                          [LK, 01/23/95]
%% Computes the frequency response of the 1-D filter with
%% impulse "h".
%% N1 = starting index of h
%% Generalized version of oneDfft.m
%%
m = max(size(h));
```

```
hc = zeropad(h,Ng-m);
hc = crmz_rotate(hc,N1);
H = fftshift(fft(hc));
%%%%%%%%%% End of indxoneDfft.m %%%%%%%%%%


%%%%%%%%%% twoDfft.m %%%%%%%%%%
function [H] = twoDfft(h, Ng)
%%%                          [LK, 01/23/95]
%% Computes the frequency response of the 2-D filter
%% with (noncausal) impulse h with odd-length sides.
%%
m2 = max(size(h));
rf = (m2-1)/2;
hc = zeropad(h,Ng-m2);
hc = crmz_rotate(hc,-rf);
hc = hc.';
hc = zeropad(hc,Ng-m2);
hc = crmz_rotate(hc,-rf);
hc = hc.';
H = fftshift(fft2(hc));
%%%%%%%%%% End of twoDfft.m %%%%%%%%%%

%%%%%%%%%% indxtwoDfft.m %%%%%%%%%%
function [H] = indxtwoDfft(h, N1, N2, Ng)
%%%                          [LK, 01/23/95]
%% Computes the frequency response of the 2-D filter
%% with impulse h.
%% N1 = row starting index
%% N2 = column starting index
%% Generalized version of twoDfft.m
%%
m2 = size(h,2);
hc = zeropad(h,Ng-m2);
hc = myrotate(hc,N2);
hc = hc.';
m2 = size(h,1);
hc = zeropad(hc,Ng-m2);
hc = crmz_rotate(hc,N1);
hc = hc.';
H = fftshift(fft2(hc));
%%%%%%%%%% End of indxtwoDfft.m %%%%%%%%%%
```

APPENDIX E

Filter Coefficients Listing

Due to the large size of the designed two-dimensional filters, only the coefficients of the one-dimensional filters designed in Chapter 3 are listed here.

E.1 Nearly-linear Phase

Example 1 ($N = 32$)

h_1	–	h_6 :	−0.0006	−0.0073	−0.0119	−0.0178	−0.0213	−0.0194
h_7	–	h_{12} :	−0.0093	0.0106	0.0398	0.0754	0.1121	0.1436
h_{13}	–	h_{18} :	0.1639	0.1685	0.1560	0.1286	0.0911	0.0504
h_{19}	–	h_{24} :	0.0135	−0.0143	−0.0300	−0.0336	−0.0279	−0.0167
h_{25}	–	h_{30} :	−0.0045	0.0055	0.0114	0.0130	0.0112	0.0077
h_{31}	–	h_{32} :	0.0050	0.0001				

Example 2 ($N = 25$)

h_1	–	h_6 :	0.0109	$0.0227i$	−0.0423	$0.0328i$	−0.0859	$-0.0528i$
h_7	–	h_{12} :	−0.1312	$-0.6241i$	0.6357	$0.8047i$	−0.0587	$-0.0628i$
h_{13}	–	h_{18} :	−0.3345	$0.0666i$	−0.0423	$-0.0972i$	−0.0067	$-0.0882i$
h_{19}	–	h_{24} :	0.0005	$-0.0069i$	0.0419	$-0.0044i$	0.0123	$0.0133i$
		h_{25} :	0.0003					

Example 3 ($N = 31$)

h_1	–	h_6 :	0.0099	0.0180	−0.0242	−0.0459	0.0084	0.0375
h_7	–	h_{12} :	0.0018	0.0445	0.0746	−0.1186	−0.2376	0.0684
h_{13}	–	h_{18} :	0.3380	0.0718	−0.2590	−0.1345	0.0891	0.0535
h_{19}	–	h_{24} :	0.0011	0.0497	0.0128	−0.0608	−0.0273	0.0146
h_{25}	–	h_{30} :	−0.0041	0.0051	0.0307	0.0053	−0.0242	−0.0096
		h_{31} :	0.0045					

Example 4 ($N = 40$)

h_1	–	h_6 :	0.0490	0.0577	0.0215	−0.0276	−0.0213	0.0253
h_7	–	h_{12} :	0.0289	−0.0242	−0.0404	0.0230	0.0589	−0.0234
h_{13}	–	h_{18} :	−0.1024	0.0237	0.3159	0.4756	0.3189	0.0247
h_{19}	–	h_{24} :	−0.1047	−0.0248	0.0602	0.0245	−0.0398	−0.0239
h_{25}	–	h_{30} :	0.0277	0.0231	−0.0191	−0.0216	0.0133	0.0198
h_{31}	–	h_{36} :	−0.0093	−0.0182	0.0078	0.0179	−0.0104	−0.0284
h_{37}	–	h_{40} :	0.0052	0.0561	0.0671	0.0355		

E.2 Exactly-linear Phase

Example 1 ($N = 32$)

h_1	–	h_6 :	0.0044	0.0041	0.0035	0.0002	−0.0061	−0.0146
h_7	–	h_{12} :	−0.0230	−0.0279	−0.0256	−0.0133	0.0103	0.0435
h_{13}	–	h_{18} :	0.0821	0.1200	0.1503	0.1672	0.1672	0.1503
h_{19}	–	h_{24} :	0.1200	0.0821	0.0435	0.0103	−0.0133	−0.0256
h_{25}	–	h_{30} :	−0.0279	−0.0230	−0.0146	−0.0061	0.0002	0.0035
h_{31}	–	h_{32} :	0.0041	0.0044				

Example 2 ($N = 25$)

h_1	–	h_6 :	0.0166	$0.0069i$	0.0110	$0.0220i$	−0.0303	$0.0697i$
h_7	–	h_{12} :	−0.0453	$0.0253i$	−0.0918	$-0.1888i$	−0.3445	$-0.4095i$
h_{13}	–	h_{18} :	0.9702	$0.4095i$	−0.3445	$0.1888i$	−0.0918	$-0.0253i$
h_{19}	–	h_{24} :	−0.0453	$-0.0697i$	−0.0303	$-0.0220i$	0.0110	$-0.0069i$
		h_{25} :	0.0166					

Example 3 ($N = 31$)

h_1	–	h_6 :	−0.0001	0.0051	−0.0012	0.0137	0.0210	−0.0288
h_7	–	h_{12} :	−0.0564	0.0116	0.0444	0.0013	0.0483	0.0817
h_{13}	–	h_{18} :	−0.1261	−0.2482	0.0701	0.3381	0.0701	−0.2482
h_{19}	–	h_{24} :	−0.1261	0.0817	0.0483	0.0013	0.0444	0.0116
h_{25}	–	h_{30} :	−0.0564	−0.0288	0.0210	0.0137	−0.0012	0.0051
		h_{30} :	−0.0001					

Example 4 ($N = 40$)

h_1	–	h_6 :	0.0404	0.0631	0.0385	−0.0105	−0.0274	0.0065
h_7	–	h_{12} :	0.0248	−0.0026	−0.0287	0.0004	0.0328	0.0056
h_{13}	–	h_{18} :	−0.0414	−0.0134	0.0535	0.0301	−0.0791	−0.0708
h_{19}	–	h_{24} :	0.1663	0.4323	0.4323	0.1663	−0.0708	−0.0791
h_{25}	–	h_{30} :	0.0301	0.0535	−0.0134	−0.0414	0.0056	0.0328
h_{31}	–	h_{36} :	0.0004	−0.0287	−0.0026	0.0248	0.0065	−0.0274
h_{37}	–	h_{40} :	−0.0105	0.0385	0.0631	0.0404		

E.3 Seismic Migration

Example 1 ($N = 19$)

h_1	–	h_4 :	$0.0147 - 0.0293i$	$-0.0194 - 0.0055i$	$0.0034 + 0.0380i$	$0.0305 - 0.0529i$
h_5	–	h_8 :	$-0.0567 + 0.0168i$	$0.0348 + 0.0568i$	$0.0546 - 0.0957i$	$-0.1690 + 0.0043i$
h_9	–	h_{12} :	$0.1176 + 0.2814i$	$0.5214 + 0.3678i$	$0.1176 + 0.2814i$	$-0.1690 + 0.0043i$
h_{13}	–	h_{16} :	$0.0546 - 0.0957i$	$0.0348 + 0.0568i$	$-0.0567 + 0.0168i$	$0.0305 - 0.0529i$
h_{17}	–	h_{19} :	$0.0034 + 0.0380i$	$-0.0194 - 0.0055i$	$0.0147 - 0.0293i$	

Example 2 ($N = 35$)

h_1	–	h_4 :	$0.0014 - 0.0072i$	$-0.0387 + 0.0072i$	$0.0879 + 0.0050i$	$-0.1299 - 0.0280i$
h_5	–	h_8 :	$0.1263 + 0.0500i$	$-0.0655 - 0.0515i$	$-0.0185 + 0.0243i$	$0.0663 + 0.0155i$
h_9	–	h_{12} :	$-0.0433 - 0.0358i$	$-0.0231 + 0.0169i$	$0.0616 + 0.0248i$	$-0.0252 - 0.0431i$
h_{13}	–	h_{16} :	$-0.0519 + 0.0040i$	$0.0717 + 0.0659i$	$0.0279 - 0.0700i$	$-0.1427 - 0.0965i$
h_{17}	–	h_{20} :	$-0.1110 + 0.2857i$	$-0.0176 + 0.5709i$	$-0.1110 + 0.2857i$	$-0.1427 - 0.0965i$
h_{21}	–	h_{24} :	$0.0279 - 0.0700i$	$0.0717 + 0.0659i$	$-0.0519 + 0.0040i$	$-0.0252 - 0.0431i$
h_{25}	–	h_{28} :	$0.0616 + 0.0248i$	$-0.0231 + 0.0169i$	$-0.0433 - 0.0358i$	$0.0663 + 0.0155i$
h_{29}	–	h_{32} :	$-0.0185 + 0.0243i$	$-0.0655 - 0.0515i$	$0.1263 + 0.0500i$	$-0.1299 - 0.0280i$
h_{33}	–	h_{35} :	$0.0879 + 0.0050i$	$-0.0387 + 0.0072i$	$0.0014 - 0.0072i$	

Example 2 ($N = 36$)

h_1	–	h_4 :	$0.0350 + 0.0083i$	$-0.0543 - 0.0275i$	$0.0581 + 0.0438i$	$-0.0267 - 0.0479i$
h_5	–	h_8 :	$-0.0236 + 0.0314i$	$0.0543 - 0.0031i$	$-0.0379 - 0.0158i$	$-0.0119 + 0.0099i$
h_9	–	h_{12} :	$0.0478 + 0.0135i$	$-0.0316 - 0.0274i$	$-0.0221 + 0.0107i$	$0.0542 + 0.0258i$
h_{13}	–	h_{16} :	$-0.0204 - 0.0406i$	$-0.0510 + 0.0016i$	$0.0685 + 0.0664i$	$0.0290 - 0.0696i$
h_{17}	–	h_{20} :	$-0.1410 - 0.0966i$	$-0.1129 + 0.2851i$	$-0.0178 + 0.5710i$	$-0.1098 + 0.2869i$
h_{21}	–	h_{24} :	$-0.1424 - 0.0979i$	$0.0259 - 0.0706i$	$0.0722 + 0.0694i$	$-0.0481 - 0.0010i$
h_{25}	–	h_{28} :	$-0.0310 - 0.0403i$	$0.0620 + 0.0272i$	$-0.0119 + 0.0095i$	$-0.0638 - 0.0267i$
h_{29}	–	h_{32} :	$0.0839 + 0.0090i$	$-0.0178 + 0.0255i$	$-0.0932 - 0.0479i$	$0.1763 + 0.0445i$
h_{33}	–	h_{36} :	$-0.1875 - 0.0233i$	$0.1367 + 0.0022i$	$-0.0680 + 0.0079i$	$0.0175 - 0.0069i$

Example 3 ($N = 31$)

h_1	–	h_4 :	$-0.0065 - 0.0005i$	$0.0208 + 0.0108i$	$-0.0369 - 0.0374i$	$0.0380 + 0.0798i$
h_5	–	h_8 :	$-0.0059 - 0.1194i$	$-0.0609 + 0.1246i$	$0.1342 - 0.0724i$	$-0.1665 - 0.0252i$
h_9	–	h_{12} :	$0.1246 + 0.1155i$	$-0.0257 - 0.1339i$	$-0.0583 + 0.0589i$	$0.0446 + 0.0500i$
h_{13}	–	h_{16} :	$0.0829 - 0.0668i$	$-0.1896 - 0.1193i$	$-0.0954 + 0.3400i$	$-0.0162 + 0.5018i$
h_{17}	–	h_{20} :	$-0.0954 + 0.3400i$	$-0.1896 - 0.1193i$	$0.0829 - 0.0668i$	$0.0446 + 0.0500i$
h_{21}	–	h_{24} :	$-0.0583 + 0.0589i$	$-0.0257 - 0.1339i$	$0.1246 + 0.1155i$	$-0.1665 - 0.0252i$
h_{25}	–	h_{28} :	$0.1342 - 0.0724i$	$-0.0609 + 0.1246i$	$-0.0059 - 0.1194i$	$0.0380 + 0.0798i$
h_{29}	–	h_{31} :	$-0.0369 - 0.0374i$	$0.0208 + 0.0108i$	$-0.0065 - 0.0005i$	

Bibliography

[1] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing.* New Jersey: Prentice Hall, 1989.

[2] T. W. Parks and C. S. Burrus, *Digital Filter Design.* New York: Wiley, 1987.

[3] J. H. McClellan, *On the Design of One-dimensional and Two-dimensional FIR Digital Filters.* PhD thesis, Rice University, April 1973.

[4] D. E. Dudgeon and R. M. Mersereau, *Multidimensional Digital Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1984.

[5] X. Chen and T. W. Parks, "Design of FIR filters in the complex domain," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 144–153, Feb. 1987.

[6] K. Preuss, "On the design of FIR filters by complex Chebyshev approximation," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 702–712, May 1989.

[7] R. L. Streit and A. H. Nuttall, "A general Chebyshev complex function approximation procedure and an application to beamforming," *J. Acoust. Soc. Amer.*, vol. 72, pp. 181–190, July 1982.

[8] L. G. Cuthbert, "Optimizing non-recursive digital filters to non-linear phase characteristics," *The Radio and Electronic Engineer*, vol. 44, pp. 645–651, Dec. 1974.

[9] A. G. J. Holt, J. Attikiouzel, and R. Bennett, "Iterative technique for designing non-recursive digital filters to non-linear phase characteristics," *The Radio and Electronic Engineer*, vol. 46, pp. 589–592, Dec. 1976.

[10] E. Y. Remes, "General computational methods of Tchebycheff approximation," tech. rep., Kiev, 1957. (Atomic Energy Commission Translation 4491, pp. 1-85).

[11] J. H. McClellan and T. W. Parks, "A unified approach to the design of optimal FIR linear-phase digital filters," *IEEE Trans. Circuit Theory*, vol. CT-20, pp. 697–701, Nov. 1973.

[12] I. Barrodale, L. M. Delves, and J. C. Mason, "Linear Chebyshev approximation of complex-valued functions," *Math Comp.*, vol. 32, pp. 853–863, 1978.

[13] K. Steiglitz, "Design of digital phase networks," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 171–176, April 1981.

[14] K. Glashoff and K. Roleff, "A new method for Chebyshev approximation of complex valued functions," *Math. Comp.*, vol. 36, pp. 233–239, 1981.

[15] D. G. Luenberger, *Linear and Nonlinear Programming*. Reading, MA: Addison-Wesley, 1984.

[16] P. P. Tang, "A fast algorithm for linear complex Chebyshev approximations," *Math. Comp.*, vol. 52, pp. 721–739, Oct. 1988.

[17] A. S. Alkhairy, K. G. Christian, and J. S. Lim, "Design and characterization of optimal FIR filters with arbitrary phase," *IEEE Trans. Signal Processing*, vol. 41, pp. 559–572, Feb. 1993.

[18] M. Schulist, "Complex approximation with additional constraints," in *Proc. IEEE ICASSP 1992*, pp. 101–104, 1992.

[19] M. Schulist, "Improvements of a complex FIR filter design algorithm," *Signal Processing*, vol. 20, pp. 81–90, May 1990.

[20] G. A. Watson, "The calculation of best restricted approximations," *SIAM J. Num. Anal.*, vol. 11, pp. 693–699, Sept. 1974.

[21] G. A. Watson, "A multiple exchange algorithm for multivariate chebyshev approximation," *SIAM J. Num. Anal.*, vol. 12, pp. 46–52, March 1975.

[22] D. Burnside and T. W. Parks, "Accelerated design of FIR filters in the complex domain," in *Proc. IEEE ICASSP 1993*, pp. 81–84, 1993.

[23] J. D. Fisher, *Design of finite impulse response digital filters.* PhD thesis, Rice University, Houston, TX, May 1973.

[24] S. Ellacott and J. Williams, "Linear Chebyshev approximation in the complex plane using Lawson's algorithm," *Math. Comp.*, vol. 30, pp. 35–44, Jan. 1976.

[25] C. S. Burrus, J. A. Barreto, and I. W. Selesnick, "Reweighted least squares design of FIR filters," in *The 1992 DSP Workshop*, Sept. 1992.

[26] C.-Y. Tseng, "Further results on complex Chebyshev FIR filter design using a multiple exchange algorithm," in *Proc. IEEE ISCAS*, pp. 343–346, May 1993.

[27] A. W. Potchinkov and R. M. Reemtsen, "FIR filter design in the complex domain by a semi-infinite programming technique. i. The method," *International Journal of Electronics and Communications*, vol. 48, no. 3, pp. 135–144, 1994.

[28] A. W. Potchinkov and R. M. Reemtsen, "FIR filter design in the complex domain by a semi-infinite programming technique. ii. Examples," *International Journal of Electronics and Communications*, vol. 48, no. 4, pp. 200–209, 1994.

[29] T. W. Parks and J. H. McClellan, "Chebyshev approximation for nonrecursive digital filters with linear phase," *IEEE Trans. Circuit Theory*, vol. CT-19, pp. 189–194, March 1972.

[30] E. W. Cheney, *Introduction to Approximation Theory.* New York: McGraw-Hill, 1966.

[31] G. Meinardus, *Approximation of Functions: Theory and Numerical Methods.* New York: Springer-Verlag, 1967. (Translated by L. L. Schumaker).

[32] G. G. Lorentz, *Approximation of Functions.* New York: Holt, Rinehart and Winston, 1966.

[33] T. J. Rivlin and H. S. Shapiro, "A unified approach to certain problems of approximation and minimization," *J. Soc. Indust. Appl. Math.*, vol. 9, pp. 670–699, Dec. 1961.

[34] J. R. Rice, *The Approximation of Functions.* Reading, Massachusetts: Addison-Wesley Publishing Corp., 1969.

[35] D. G. Luenberger, *Optimization by Vector Space Methods.* New York: Wiley, 1968.

[36] J. Williams, "Numerical Chebyshev approximation in the complex plane," *SIAM J. Numerical Analysis*, vol. 9, pp. 638–649, Dec. 1972.

[37] M. T. McCallig, "Design of digital FIR filters with complex conjugate pulse responses," *IEEE Trans. Circuit and Systems*, vol. CAS-25, pp. 1103–1105, Dec. 1978.

[38] V. F. Demjanov and V. N. Malozemov, *Introduction To Minimax.* New York: Wiley, 1974.

[39] L. J. Karam and J. H. McClellan, "Complex Chebyshev approximation for FIR filter design." To appear in the IEEE Trans. on Circuits and Systems II - March 1995.

[40] D. B. Harris and R. M. Mersereau, "A comparison of algorithms for minimax design of two-dimensional linear phase fir digital filters," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-25, pp. 492–500, Dec. 1977.

[41] V. F. Demjanov, "Algorithms for some minimax problems," *Journal of Computer and System Sciences*, vol. 2, pp. 342–380, 1968.

[42] P. Wolfe, "Finding the nearest point in a polytope," *Mathematical Programming*, vol. 11, pp. 128–149, 1976.

[43] P. Wolfe, "A method of conjugate subgradients for minimizing nondifferentiable functions," *Mathematical Programming Study*, vol. 3, pp. 145–173, 1975.

[44] A. Feuer, "Minimizing well-behaved functions," in *12th Annual Allerton Conference on circuit and system theory*, Oct. 1974.

[45] L. J. Karam and J. H. McClellan, "A multiple exchange Remez algorithm for complex FIR filter design in the Chebyshev sense," in *Proc. IEEE ISCAS*, June 1994.

[46] D. B. Harris, *Design and Implementaion of Rational 2-D Digital Filters.* PhD thesis, Massachusetts Institute of Technology, Nov. 1979. See Appendix.

[47] J. Claerbout, *Fundamentals of Geophysical Data Processing.* New York: McGraw-Hill, 1976.

[48] D. Hale, "3-D depth migration via McClellan transformations," *Geophysics*, vol. 56, pp. 1778–1785, Nov. 1991.

[49] J. P. Hedley, "3-D migration via McClellan transformations on hexagonal grids," *Geophysics*, vol. 57, pp. 1048–1053, Aug. 1992.

[50] J. H. McClellan, "The design of two-dimensional digital filters by transformations," in *Proc. 7th Annual Princeton Conf. on Inform. Sci. and Syst.*, pp. 247–251, 1973.

[51] J. V. Hu and L. R. Rabiner, "Design techniques for two-dimensional digital filters," *IEEE Trans. Audio Electroacoust.*, vol. AU-20, pp. 249–257, Oct. 1972.

[52] Y. Kamp and J. P. Thiran, "Chebushev approximation for two-dimesnional nonrecursive digital filters," *IEEE Trans. on Circuits and Systems*, vol. CAS-22, pp. 208–218, March 1975.

[53] C. Y. Chi and S. L. Chiou, "A new self-initiated WLS approximation method for the design of two-dimensional equiripple FIR digital filters," in *IEEE ISCAS*, pp. 1436–1439, May 1992.

[54] Y. C. Lim, J. H. Lee, C. K. Chen, and R. H. Yang, "A weighted least squares algorithm for quasi-equiripple FIR and IIR digital filter design," *IEEE Trans. on Signal Processing*, vol. ASSP-40, pp. 551–558, March 1992.

[55] C. Y. Chi and S. L. Chiou, "A new iterative WLS Chebyshev approximation method for the design of two-dimensional FIR digital filters," in *IEEE ISCAS*, pp. 335–338, May 1993.

[56] R. M. Mersereau and D. E. Dudgeon, "The representation of two-dimensional sequences as one-dimensional sequences," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-22, pp. 320–325, Oct. 1974.

[57] R. M. Mersereau, W. G. Mecklenbräuker, and T. F. Quatieri, "McClellan transformations for two-dimensional digital filtering: I - design," *IEEE Trans. Circuits Syst.*, vol. CAS-23, pp. 405–414, 1976.

[58] D. T. Nguyen and M. N. S. Swamy, "Approximation design of 2-D digital filters with elliptical magnitude response of arbitrary orientation," *IEEE Trans. Circuits and Systems*, vol. CAS-33, pp. 597–603, June 1986.

[59] J. H. McClellan and D. S. Chan, "A 2-d filter structure derived from the chebyshev recursion," *IEEE Trans. Circuits Syst.*, vol. CAS-24, pp. 372–378, 1977.

[60] T. F. Quatieri, "The design of two-dimensional digital filters by generalized mcclellan transformations," Master's thesis, Massachusetts Institute of Technology, June 1975.

[61] T. Sekiguchi and S. Takahashi, "McClellan transformations for multidimensional FIR digital filters with complex coefficients," *Electronics and Communications in Japan, Part 3*, vol. 72, no. 4, pp. 79–93, 1989.

[62] Y. L. Tai and T. P. Lin, "Design of 2-D FIR filter possessing purely imaginary frequency response by the transformation method," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 2, pp. 89–91, March 1992.

[63] S. C. Pei and J. J. Shyu, "Design of 2-D FIR digital filters by McClellan transformation and least squares eigencontour mapping," *IEEE Trans. Circuits and Systems – II*, vol. 40, pp. 119–129, Sept. 1993.

[64] S. Tazawa, H. Honma, and M. Sagawa, "A method of designing FIR fan filters based on McClellan transformations," *Electronics and Communications in Japan, Part 3*, vol. 73, no. 12, pp. 79–87, 1990.

[65] J. H. McClellan and T. W. Parks, "Equiripple approximation of fan filters," *Geophysics*, vol. 37, pp. 573–583, 1972.

[66] P. K. Rajan and M. N. S. Swamy, "Design of circularly symmetric two-dimensional fir filters employing transformations with variable parameters," *IEEE Trans. Acoustic, Speech and Signal Processing*, vol. ASSP-31, pp. 637–642, June 1983.

[67] A. V. Oppenheim, W. F. G. Mecklenbräuker, and R. M. Mersereau, "Variable cutoff linear phase digital filters," *IEEE Trans. on circuits and systems*, vol. CAS-23, pp. 199–203, April 1976.

[68] K. Nayebi, *A Time Domain Framework for the Analysis and Design of FIR Multirate Filter Bank Systems.* PhD thesis, Georgia Institute of Technology, Dec. 1990.

[69] L. Veidinger, "On the numerical determination of the best approximations in the chebyshev sense," *J. Math. Anal. Appl.*, vol. 25, pp. 692–700, 1969.

Vita

Lina J. Karam was born on the 23rd of January 1968 in Beirut, Lebanon. After obtaining the French Baccalaureate (with concentration in Math and Physics), she joined the School of Engineering at the American University of Beirut in the Fall of 1985. There she received the B.E. degree in Electrical Engineering (Computer and Communication major) with distinction (Highest Honor) in 1989. In 1992, she received the M.S. degree in Electrical Engineering from the Georgia Institute of Technology. During her undergraduate studies (Summer 1988), she worked at Imperial College in London (UK) on problems in image processing and computer graphics. From 1991 to 1995, she was a graduate research assistant in the Graphics, Visualization, and Usability Center and then the School of Electrical and Computer Engineering at Georgia Institute of Technology. During the summer of 1992, she worked at Schlumberger Well Services in Austin (Texas) on problems related to data modeling and visualization. In Summer 1994, she worked in the Signal Processing Department of AT&T Bell Labs in Murray Hill (New Jersey) on problems in video coding.

Ms. Lina Karam is the recipient of the 1994 Society of Women Engineers Outstanding Graduate Student Award. She is also a member of the *Signal Processing* and *Circuits and Systems* societies of the IEEE.