Efficiency in audio processing : filter banks and transcoding

Citation for published version (APA):
Lee, J. W. (2007). *Efficiency in audio processing : filter banks and transcoding*. [Phd Thesis 2 (Research NOT TU/e / Graduation TU/e), Electrical Engineering]. Technische Universiteit Eindhoven.
https://doi.org/10.6100/IR624553

DOI:
10.6100/IR624553

Document status and date:
Published: 01/01/2007

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Download date: 04. Oct. 2023

Efficiency in Audio Processing: Filter Banks and Transcoding

Efficiency in Audio Processing: Filter Banks and Transcoding

LEE JUN WEI

A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE
2007

Efficiency in Audio Processing: Filter Banks and Transcoding

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van de
Rector Magnificus, prof.dr.ir. C.J. van Duijn, voor een
commissie aangewezen door het College voor
Promoties in het openbaar te verdedigen
op maandag 19 maart 2007 om 16.00 uur

door

Jun Wei Lee

geboren te Singapore, Singapore

Dit proefschrift is goedgekeurd door de promotoren:

prof.dr.ir. J.W.M. Bergmans
en
prof.dr. C.C. Ko

Copromotor:
dr. S.H. Ong

© Copyright 2007 Lee Jun Wei
All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior written permission from the copyright owner.

CIP-DATA LIBRARY TECHNISCHE UNIVERSITEIT EINDHOVEN

Lee, Jun Wei

Efficiency in audio processing : filter banks and transcoding / by Jun Wei Lee. – Eindhoven : Technische Universiteit Eindhoven, 2007.
Proefschrift. – ISBN 978-90-386-1993-4
NUR 959
Trefw.: signaalverwerking / digitale filters / signaalcodering / digitale geluidstechniek.
Subject headings: signal processing / digital filters / audio coding / transcoding.

promotiecommissie:

dr.ir. M.J. Bastiaans

prof.dr.ir. J.W.M. Bergmans

prof.dr. H. Corporaal

dr.ir. R. Heusdens

prof.dr. C.C. Ko

prof.dr. Y.C. Lim

dr. S.H. Ong

Acknowledgements

This thesis is made possible only because of the great supervision and support that has been so generously given to me, which I am deeply grateful for.

I would like to first thank my main supervisor, Professor Lim Yong Ching, for his deep insights into problems, and his constant support in so many different ways. Many opportunities such as the joint PhD program with TU/e were made possible only because of his resource and support. It is a great pleasure to have such a patient and generous supervisor.

I am also extremely indebted to my co-supervisor Professor Jan Bergmans, for his time in guiding me and supervising this work. I am very glad to receive his critical, and yet patient, reviews on my work and this thesis. I have learnt much from him during this time, most especially, in looking at my own work in a critical manner.

I would like to thank my co-supervisor Professor Ong Sim Heng for his time and advice in the matters that arose during my candidature, and for being such a friendly and approachable supervisor.

I would like to highlight the fact that for a significant portion of this PhD candidature (3 years), the work was supported financially by the Singapore Millenium Foundation (SMF) scholarship. I am extremely grateful to SMF for their generous support towards research.

Much of the work in Chapter 10 was done in collaboration with Werner Oomen and Frans de Bont, from Philips PDSL Sound Coding Group, Eindhoven. It is my pleasure to have been friends with them and I am very grateful for their advice and help when I needed it. I would like to thank Dr See Chong Meng, from Defense Science Organization, Singapore. I am glad for the opportunity to have worked with him on the subject of filter banks.

I would like to thank Dr Ad van den Enden, who was instrumental in co-ordinating this Joint PhD project; Dr Aweke Lemma and Dr Michiel van der Veen, from Philips Research, for their insights and advice on some of the work in Chapter 8; and also Leon van de Kerkhof, of Philips PDSL for graciously presenting a paper at a conference that I was unable to attend.

I am very grateful to Sjoerd Ypma, from TU/e, for extending his extremely warm welcome during my attachment in Eindhoven; and to Yvonne Broers for helping out in some of the arrangements.

Lastly, and no doubt very very importantly, I want to thank my parents, Lee Ah Bah and Goh Tock Mui; my in-laws, Lloyd Lim and Yap Ah Moy; and my wife Lydia Lim, for their never-ending support.

Contents

Summary v

List of Tables vii

List of Figures ix

Glossary xiii

Chapter 1 Introduction 1

1.1 About this thesis 1
1.2 Compressed digital audio processing 1
1.3 About our work 3
1.4 Thesis overview 4
1.5 Publications 6

PART I: FILTER BANKS 7

Chapter 2 General Overview of Filter Banks 9

2.1 Filter bank overview 9
2.2 Definitions and conventions 11
2.3 Filter-array filter bank 14
2.4 Polyphase filter bank 15
2.5 Octave filter bank 18
2.6 Fast Filter Bank (FFB) 21
2.7 Filter bank comparisons 23
2.8 Summary 25

Chapter 3 Single-rate Systems I: The Fast Filter Bank 27

3.1 Frequency response masking technique 27
3.2 FFB structure and operation 30
3.3 FFB design considerations 34
3.4 Properties of the FFB 41
3.5 Design example: 32-channel FFB 43
3.6 Summary 45

Chapter 4 Single-rate Systems II: Simplifications of the Fast Filter Bank 47

4.1 Processing of real-valued input signals 47
4.2 Node-modulation method 50
4.3 Pruning method 53

4.4 Odd-stacked 16-channel FFB 57
4.5 A matrix formulation 58
4.6 Summary 63

Chapter 5 Multi-rate Systems I: Frequency Response Masking Filter Banks 65

5.1 Complexity considerations 65
5.2 Different types of FRM-class filter banks 70
5.3 Alternate multi-rate structures 76
5.4 Summary 78

Chapter 6 Multi-rate Systems II: Reconstruction Analysis and Multiplierless Designs 81

6.1 Reconstruction analysis for 4 channels 81
6.2 Reconstruction analysis for *M* channels 87
6.3 Design of filters using signed-powers-of-two terms 90
6.4 Variable bandwidth filter bank 92
6.5 Summary 98

References for Part I 101

PART II: AUDIO TRANSCODING 105

Chapter 7 Overview of Audio Transcoding 107

7.1 Introduction to audio coding 107
7.2 Introduction to audio transcoding 110
7.3 Cascaded quantization model 112
7.4 Definitions and conventions 115
7.5 Overview of quantization schemes 119
7.6 Outline of our work 123

Chapter 8 Tandem Quantization Error 125

8.1 Definition of tandem quantization error 125
8.2 Tandem quantization error regions 128
8.3 Mean tandem noise power 132
8.4 Effect of different cascade-pair combinations 133
8.5 Overview of proposed methods to reduce tandem noise power 134
8.6 Modified quantizer method 135
8.7 Quantizer selection method 144
8.8 Comments for other distributions 150
8.9 Summary 151

Chapter 9 Analysis of the Impact of Sample-synchronized and Non-sample-synchronized Audio Transcoding for MPEG 1 153

9.1 Overview 154
9.2 Analysis for MPEG 1 Layer 2 154

9.3 Analysis for MP3 160
9.4 Remarks 168
9.5 Summary 169

Chapter 10 An Ultra-fast Bit-rate Transcoder for MP3 171

10.1 Audio bit-rate transcoding 171
10.2 Complexity reduction I: Frame-synchronization 177
10.3 Complexity reduction II: Modified psychoacoustic model 177
10.4 Complexity reduction III: Direct mapping 178
10.5 Results 189
10.6 Discussions 191
10.7 Summary and future work 191

References for Part II 193

Chapter 11 Conclusion 197

Appendix A Length of a Filter 199

Appendix B Complexity of a Filter 201

Appendix C MP3 Description 205

Appendix D Input-to-output Relationship for a Transcoder 209

Appendix E Listening Test Results 215

Samenvatting 223

Curriculum Vitae 225

Summary

Efficiency in Audio Processing: Filter Banks and Transcoding

Audio transcoding is the conversion of digital audio from one compressed form A to another compressed form B, where A and B have different compression properties, such as a different bit-rate, sampling frequency or compression method. This is typically achieved by decoding A to an intermediate uncompressed form, and then encoding it to B. A significant portion of the involved computational effort pertains to operating the synthesis filter bank, which is an important processing block in the decoding stage, and the analysis filter bank, which is an important processing block in the encoding stage.

This thesis presents methods for efficient implementations of filter banks and audio transcoders, and is separated into two main parts. In the first part, a new class of Frequency Response Masking (FRM) filter banks is introduced. These filter banks are usually characterized by comprising a tree-structured cascade of subfilters, which have small individual filter lengths. Methods of complexity reduction are proposed for the scenarios when the filter banks are operated in single-rate mode, and when they are operated in multi-rate mode; and for the scenarios when the input signal is real-valued, and when it is complex-valued. An efficient variable bandwidth FRM filter bank is designed by using signed-powers-of-two reduction of its subfilter coefficients. Our design has a complexity an order lower than that of an octave filter bank with the same specifications.

In the second part, the audio transcoding process is analyzed. Audio transcoding is modeled as a cascaded quantization process, and the cascaded quantization of an input signal is analyzed under different conditions, for the MPEG 1 Layer 2 and MP3 compression methods. One condition is the input-to-output delay of the transcoder, which is known to have an impact on the audio quality of the transcoded material. Methods to reduce the error in a cascaded quantization process are also proposed. An ultra-fast MP3 transcoder that requires only integer operations is proposed and implemented in software. Our implementation shows an improvement by a factor of 5 to 16 over other best known transcoders in terms of execution speed.

List of Tables

Table 2-1 General complexity comparison. 23

Table 2-2 Complexity comparisons between the uniform polyphase filter bank and the FFB in single-rate mode (L=1). 24

Table 2-3 Complexity comparisons between the uniform polyphase filter bank, octave filter bank and the FFB in critically-decimated mode (L=M). 25

Table 2-4 Summary of the advantages and disadvantages associated with the various filter bank implementations. 25

Table 3-1 Effect of $\Delta_{0,0}$ on the filter bank complexity, for $K = 4$ and $\delta_s = -50$ dB. 40

Table 3-2 Complexity distribution of the FFB, for $M = 128$, $\Delta_{0,0} = 0.2\pi$ and $\delta_s = -50$ dB. 40

Table 3-3 Coefficient values for $H_{u,0}(z)$ for a 32-channel FFB. 43

Table 4-1 Complexity breakdown by stage for pruned node-modulated FFB. 58

Table 5-1 Lengths of subfilters with transfer functions $H_{u,0}(z)$ for various design conditions. 66

Table 5-2 Complexity comparisons of the various filter bank implementations. 74

Table 5-3 Summary of the characteristics of the various FRM-class filter banks. 75

Table 6-1 Example showing the values of $s(x,b)$ for the SPT-2 reduction of the number x= -0.19, for B=5. 91

Table 6-2 SPT coefficients for masking subfilters used in 5-channel variable bandwidth filter bank. 96

Table 6-3 Comparisons between our variable bandwidth filter bank and the narrow transition width octave filter bank. 97

Table 6-4 Comparisons between our variable bandwidth filter bank and the unequal transition width octave filter bank. 98

Table 7-1 Some examples of the number of subbands M for different compression methods. 113

Table 8-1 Comparison of quantization error power for different cascaded quantization methods. 140

Table 8-2 Mean quantization error powers, using the modified quantizer method when $q_{2}[m]$ is not known a-priori, for the example when a 7-step $q_{1}[m]$ is used. 143

Table 8-3 Regions of influence for a 7-step/5-step cascaded pair. 144

Table 8-4 Reference table for a 7-step/5-step cascaded pair. 148

Table 10-1 Allocation table for $(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})$, for a 'normal' degree of compression. 186

Table 10-2 Results of test on transcoding speed. 191

List of Figures

Figure 1-1 Simplified overview of an audio encoder/decoder. 2
Figure 2-1 Typical model of audio compression using a pair of analysis-synthesis filter banks. 9
Figure 2-2 Frequency response of a generic filter. 11
Figure 2-3 Frequency response for a (a) transfer function with purely real-valued coefficients; (b) transfer function with complex-valued coefficients. 13
Figure 2-4 Filter-array for an analysis-synthesis filter bank pair. 14
Figure 2-5 Frequency responses of (a) even-stacked, and (b) odd-stacked filter banks. 15
Figure 2-6 Polyphase structure for a *M*-channel analysis filter bank. 16
Figure 2-7 Equal bandwidth octave filter bank comprising two-channel filter banks.... 18
Figure 2-8 Octave filter bank in an unequal bandwidth configuration. 19
Figure 2-9 Frequency responses of the octave filter bank. 20
Figure 2-10 The tree-structured Fast Filter Bank. 22
Figure 3-1 Block diagram showing the structure of a (a) single-stage filter; (b) 2-stage FRM filter. 27
Figure 3-2 Plot of required filter length *N* for transition width Δ, when $\delta_p = 0.1$ dB and $\delta_s = -60$ dB. 28
Figure 3-3 Frequency response plots illustrating the FRM technique. 29
Figure 3-4 Representation of a subfilter block $H_{u,v}(z^a)$ 30
Figure 3-5 The tree-structured Fast Filter Bank. 31
Figure 3-6 Eight-channel sliding FFT with a decimation-in-time FFT structure. 32
Figure 3-7 Frequency responses of the subfilters in the FFB. 34
Figure 3-8 (a) Magnitude response of filters with different passband delays, (b) Group delay of filters with different passband delays. 38
Figure 3-9 Detailed FFB block diagram. 41
Figure 3-10 Frequency responses for a 32-channel FFB, and a 32-channel sliding FFT filter bank. 44
Figure 4-1 Even and odd stacked frequency responses for real-valued inputs. 50
Figure 4-2 Scheme for the proposed node-modulated even-stacked FFB. 51
Figure 4-3 Scheme for the proposed node-modulated odd-stacked FFB. 53

Figure 4-4 Bit-pattern used for a subfilter. 53
Figure 4-5 A 16-channel FFB with bit-pattern annotations. 54
Figure 4-6 Pruning for an even-stacked 16-channel FFB. 56
Figure 4-7 Pruning for an odd-stacked FFB with input modulation. 57
Figure 4-8 Using pruning and node-modulation to implement an efficient odd-stacked 16-channel FFB. 57
Figure 4-9 Data buffer shifting. 61
Figure 4-10 Overview of matrix method for stage u. 61
Figure 4-11 Re-ordering of data vectors. 62
Figure 5-1 A general representation of the FRM-class filter bank for multirate operation. 65
Figure 5-2 Plot of decimation factor ratio (L/M) vs $\Delta_{0,0}$, when $\Gamma_{FFB} = \Gamma_{PFB}$ 67
Figure 5-3 Illustration of subfilter complexity reduction when the outputs of the FFB are decimated. 68
Figure 5-4 (a) Comparison between the complexity of the FFB and the proposed simple complexity reduction method; (b) Plot of the values of the transition width $\Delta_{0,0}$, at which the FFB (with simple complexity reduction) has equal complexity to the polyphase filter bank, when the outputs are critically decimated. 69
Figure 5-5 M-channel odd-stacked Type *F-I* FFB. 70
Figure 5-6 Type *F-II* FRM structure, with a polyphase masking subfilter block. 71
Figure 5-7 Type *F-III* FRM structure, with polyphase masking subfilter blocks. 72
Figure 5-8 Type *F-IV* FFB structure. 73
Figure 5-9 Plots of complexity of different types of FRM filter banks, relative to the complexity of the polyphase filter bank, for M=4 and M=16. 74
Figure 5-10 Equivalent multirate structure for cascaded filters. 76
Figure 5-11 Equivalent multirate structure for Type *F-I* filter bank. 77
Figure 5-12 Equivalent multirate structure for Type *F-II* filter bank. 77
Figure 5-13 Applying the equivalent multirate structure efficiently in a general frequency-hopping scenario. 78
Figure 6-1 A pruned odd-stacked FFB for a real-valued input signal. 82
Figure 6-2 (a) Frequency response of $H_{0,0}(z)$, (b) Frequency responses of $H_{[m]}(z)$ for m=2 and 3. 89
Figure 6-3 (a) Spectrum of distortion error function of filter bank, (b) Spectrum of aliasing error function of filter bank for a=1. 90
Figure 6-4 Some variable bandwidth scenarios. 93
Figure 6-5 Proposed 5-stage architecture for variable bandwidth application. 94

Figure 6-6 Further simplification of the shaping stage. 94
Figure 6-7 Frequency responses for different variable bandwidth filter banks. 97
Figure 7-1 Absolute threshold of hearing in quiet. 108
Figure 7-2 Modified hearing threshold in the presence of certain tonal stimuli. 109
Figure 7-3 Transcoding from 256 kbps to 192 kbps for the MPEG 1 Layer 2 method. 111
Figure 7-4 Cascaded quantization model for audio transcoding. 112
Figure 7-5 Expanded view of cascaded quantization model. 114
Figure 7-6 Comparison of the effect of sample-synchronization on transcoding for MPEG 1 Layer 2 and MP3. 118
Figure 7-7 Illustration of frame-synchronization. 118
Figure 7-8 A block diagram showing the MPEG 1 Layer 2 quantization scheme. 120
Figure 7-9 Illustration of normalization scaling used in MPEG 1 Layer 2. 120
Figure 7-10 Plot of quantization characteristics. 121
Figure 7-11 A block diagram showing the MP3 quantization scheme. 121
Figure 7-12 Plot of $|\tilde{\mathbf{x}}_1|$ for a typical MP3 q-frame. 123
Figure 8-1 Cascaded quantization model for sample-synchronized transcoding. 126
Figure 8-2 Illustration of tandem quantization error effect. 127
Figure 8-3 Quantizer characteristics and tandem quantization error regions for the 5-step and 7-step quantizers. 129
Figure 8-4 Plot of tandem indicators for a 7-step/5-step quantizer pair. 130
Figure 8-5 Plot of tandem noise power for a 7-step/5-step quantizer pair. 131
Figure 8-6 Tandem noise power for a 5-step $q_{2[m]}$ and various $q_{1[m]}$. 134
Figure 8-7 Proposed method for reducing tandem noise power. 135
Figure 8-8 Illustration of a modified quantization method, where $q_{1[m]}$ is replaced with $\hat{q}_{1[m]}$. 136
Figure 8-9 Dividing each of the tandem quantization error regions into 2 half-parts. 137
Figure 8-10 Diagram illustrating method to divide tandem quantization error region. 137
Figure 8-11 Effect of dividing the tandem quantization error region on the error powers. 138
Figure 8-12 Plot of quantization error power vs. input sample value for different cascaded quantization methods. 140
Figure 8-13 Sample illustration of noise-to-signal ratio and mask-to-signal ratio for reduction of tandem noise power. 141
Figure 8-14 Block diagram for quantizer selection method. 144
Figure 8-15 Plot of the ratio of the number of unchanged q-frames to the total number of q-frames, against the tolerance σ. 149

Figure 8-16 Plot of the mean quantization error power $E\left[e_{C[m]}^2\right]$ against the tolerance σ. 149

Figure 8-17 Illustration of tandem quantization error for an input signal with non-uniform distribution - Case 1. 150

Figure 8-18 Illustration of tandem quantization error for an input signal with non-uniform distribution - Case 2. 151

Figure 9-1 Effect of non-sample-synchronization on the distribution of $x'_{1[m]}$, for a uniform distribution of $x_{0[m]}$........ 155

Figure 9-2 Effect of non-sample-synchronization on the distribution of $x'_{1[m]}$, for a non-uniform distribution of $x_{0[m]}$........ 156

Figure 9-3 Comparison of predicted and measured values of $E\left[e_{C[m]}^2\right]$ for different m, and for d=1 and d=6. 158

Figure 9-4 (Repeat of Figure 7-11.) 160

Figure 9-5 Plot of $|\tilde{\mathbf{x}}_2|$ for different values of $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$. 163

Figure 9-6 Illustration of informal test on fluctuating tandem gain. 164

Figure 9-7 Distributions of $x_{1[m]}$ and $x'_{1[m]}$ for different values of delay d. 165

Figure 9-8 Quantization of dispersed $x'_{1[m]}$. 166

Figure 9-9 Effect of different $\mathbb{Z}_{2\{s\}}$, with $\mathbb{Z}_{1\{s\}} = 0$, on $|\tilde{x}_{2[m]}|$ for non-sample-synchronized transcoding. 167

Figure 9-10 Estimated regional tandem gain for different values of $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$. 167

Figure 10-1 Data truncation using audio layered transcoding. 172

Figure 10-2 Using ancillary data for transcoding. 172

Figure 10-3 Block diagram for bit-rate scaling in coded data domain. 173

Figure 10-4 Conventional implementation of an MP3 transcoder. 176

Figure 10-5 Complexity reduction I: Frame-synchronization. 177

Figure 10-6 Complexity reduction II: Modified psychoacoustic model. 178

Figure 10-7 Complexity reduction III: Direct mapping. 178

Figure 10-8 Plot of $|\tilde{\mathbf{x}}_1|$ and $|\tilde{\mathbf{x}}_2|$ for $\delta_g = 5$ and $m_{cutoff} = 342$. 182

Figure 10-9 Block diagram for a fixed mapping transcoder. 183

Figure 10-10 Division of scalefactor bands into frequency regions for fixed mapping transcoder. 184

Figure 10-11 Block diagram for adaptive mapping transcoder. 185

Figure 10-12 Lookup table method for ultra-fast integer-to-integer mapping. 188

Figure 10-13 Results of Listening Test 1. 189

Figure 10-14 Results of Listening Test 2. 190

Glossary

Abbreviations used

AAC : Advanced Audio Coding
BLAS : Basic Linear Algebra Subprograms
CD : Compact Disc
DFT : Discrete Fourier Transform
FFB : Fast Filter Bank
FFT : Fast Fourier Transform
FIR : Finite Impulse Response
FRM : Frequency Response Masking
HAS : Human Auditory System
IDFT : Inverse Discrete Fourier Transform
IFFT : Inverse Fast Fourier Transform
ISO/IEC : International Standards Organization/International Electrotechnical Commission
MDCT : Modified Discrete Cosine Transform
MOS : Mean Opinion Score
MPEG : Moving Picture Experts Group
MP3 : MPEG 1 Layer 3
NMR : Noise-to-Mask Ratio
NPR : Near-perfect Reconstruction
PCM : Pulse Code Modulation
PR : Perfect Reconstruction
QMF : Quadrature Mirror Filter
SMR : Signal-to-Mask Ratio
SNR : Signal-to-Noise Ratio
SPT : Signed-Powers-of-Two

Cross-reference notations

[1]	a literature reference.
(2.1)	a reference to the 1st equation in Chapter 2.
3.1	a reference to Chapter 3 Section 1.

Often used symbols and notations in general

a, b, c, k, r, s : used as arbitrary variables.

d : used to denote a delay. Usage is local in context and will be explained where ambiguous. In some cases, when its reference is obvious, explanations will be omitted. e.g. in the equation $\bar{H}_{u,v}(z) = e^{-j\pi d} H_{u,v}(e^{-j\pi} z)$, d is the delay of $H_{u,v}(z)$.

e : used to denote an error between 2 values.

j : square root of -1, used to denote the imaginary part of a complex number.
m : subband index.
n : sampled discrete time index.
q : denotes a quantizer, or a quantization operation.
u, v, w : arbitrary indices, usually used as subscripts or superscripts.
z : a variable in the z-transform domain, z^{-1} referring to a delay of one.
L : decimation or interpolation factor.
M : total number of subbands.
ω : frequency, normalized to a sampling frequency of 2π .

$h(n)$: impulse response of an FIR filter, usually in relation to the analysis filter bank.
$g(n)$: impulse response of an FIR filter, usually in relation to the synthesis filter bank.
$H(z)$: transfer function of an FIR filter, the z-transform equivalent of $h(n)$.
$G(z)$: transfer function of an FIR filter, the z-transform equivalent of $g(n)$.
$H_{[m]}(z)$: transfer function of subband m of an analysis filter bank.
$G_{[m]}(z)$: transfer function of subband m of a synthesis filter bank.
H : the set of analysis transfer functions $H_{[m]}(z)$, for $0 \le m \le M-1$.
G : the set of synthesis transfer functions $G_{[m]}(z)$, for $0 \le m \le M-1$.

$f_{[m',m]}$: a component of the input-to-output transform function for an analysis-synthesis filter bank pair.
$\mathbf{F}_d$: the set of transform functions comprising $f_{[m',m]}$, that varies with delay d.

$p(x)$: probability density function of x.
$x_{[m]}$: subscripted $[m]$ indicates that $x_{[m]}$ is the m-th subband component of x.
$\hat{x}$: approximate of x.
$\mathbf{x}$: the set of $x_{[m]}$, for $0 \le m \le M-1$;
also used to denote a one-dimensional vector in general.
$\mathbf{X}$: used to denote a two-dimensional vector in general.

$*$: convolution operator.
* : complex-conjugate operator.
$|.|$: the absolute value (magnitude) of the bracketed term.
$\lfloor . \rfloor$: the bracketed term is rounded down to the nearest integer.
$[\![.]\!]$: the real part of the bracketed term is taken.
$E[.]$: the expectation of the bracketed term.
$\Re\{.\}$: the bracketed term is rounded to the nearest integer.

Often used symbols and notations specific to Part I

N : filter length.
K : number of cascade stages of a multi-stage filter bank.
B: number of bits used to represent a value in fixed point.

ω_p : passband edge of a filter.
ω_s : stopband edge of a filter.

Δ : transition width of a filter.
δ_p : peak passband ripple of a filter.
δ_s : peak stopband ripple of a filter.
Γ : the complexity of a system, usually approximated by the number of multiplications required per input sample.

$\lambda(n)$: a modulating signal.
$\Phi_T(z)$: distortion error function for a filter bank pair.
$\Phi_A(z)$: aliasing error function for a filter bank pair.
$\ell(u)$: an interpolation factor used in the context of the FFB.
$\alpha(m,u)$: a binary variable used in the context of the FFB.
$W(m,u)$: an exponential function used in the context of the FFB.

$\tilde{v}$: the bit-reversed version of v.
$\underset{\sim}{h}_v(n)$: the v-th polyphase component of $h(n)$.
$\bar{H}(z)$: the complementary transfer function of $H(z)$.
$H_{u,v}(z)$: transfer function of a subfilter in a multi-stage filter bank, indexed by u and v.

Often used symbols and notations specific to Part II

s : index of scalefactor band.
A: represents a compressed audio bitstream.
B: bit-rate.
N : number of quantization steps.
R : tandem region label.
N_T : number of tandem quantization error regions for a cascaded quantizer pair.
P_T : tandem noise power.
I_T: tandem indicator.
P_R : tandem gain.

ζ : a scaling factor used in MPEG 1 Layer 2 quantization.
Δ : quantization step-size.
Φ : the set of quantization boundaries of a quantizer.
Θ : the set of quantization steps of a quantizer.
$\mathbb{Z}$: lumped quantization parameter used to describe an MP3 quantizer.
δ : the difference between the 2 quantization parameters $\mathbb{Z}_1$ and $\mathbb{Z}_2$.

$T\{\,.\,\}$: a map which is used by a transcoder.
$\mathbf{R}$: set of tandem quantization error regions

$\mathbf{x}_{\{s\}}$: a vector comprising $x_{[m]}$ falling within the scalefactor band indexed by s, in the context of MP3.
$\tilde{x}$: integer component of the real-valued x, in the context of MP3.
$\bar{x}$: mean of x.
$\hat{q}$: a modified quantizer, which is derived from q.

Chapter 1

Introduction

1.1 About this thesis

This thesis is written in partial requirement of the PhD criteria for the Joint PhD program as a collaboration between the National University of Singapore (NUS) and the Technical University of Eindhoven (TU/e). The work done in this Joint PhD program comprises 2 parts. In Part I (conducted at NUS), we focus on efficient filter bank designs. These filter banks are directed at applications in audio, but are equally applicable to other generalized filter bank applications such as communications. In Part II (conducted at TU/e), we focus on the subject of audio transcoding.

The initial objective of our work was to look into efficient filter bank designs. With this in mind, we proposed several new efficient filter bank designs in Part I, which are based on the Frequency Response Masking technique. In the collaborative work in Part II, we originally intended to apply these filter banks to the implementation of efficient audio transcoders, with the expectation that more efficient filter banks would lead to more efficient transcoders. Typically, filter banks comprise a significant amount of processing with regards to the overall transcoder processing (in the region of 15%-35%, depending on the compression method used to encode the audio). As it turns out, we found that a very efficient audio transcoder can be realized, that actually eliminates filter banks altogether. As a result, we redirected the work in Part II to the design of a 'filter bank-less' audio transcoder.

1.2 Compressed digital audio processing

1.2.1 Background

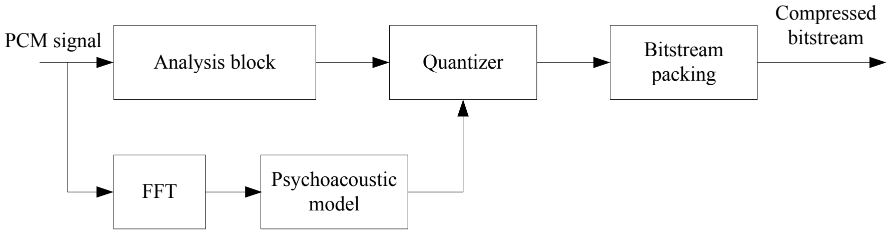
In recent decades, audio storage technology has shifted from analog audio in the form of magnetic tape storage to digital audio in the form of the Compact Disc (CD) [60]. CD-quality digital audio is sampled and quantized using the Pulse Code Modulated (PCM) format. PCM audio is typically associated with high bit-rates and large storage requirements. Conventional CD systems are usually sampled at 44.1 kHz using a 16-bit sample resolution. For a stereo audio representation, this translates to approximately 1.4 megabits per second (Mbps).

With the advances in internet technology, a demand for compressed digital audio was created. Reduced data rates and storage requirements were required for effective music sharing and distribution. Furthermore, mobile music devices using solid-state storage technology have high cost-to-storage ratios, and thus benefit from digital audio compression. Significant research and improvements have been made in the field of digital audio compression. Using current methods of compression, bit-rates can be reduced by a factor of more than 10 over the bit-rate of PCM audio, at no perceptible loss in audio quality.

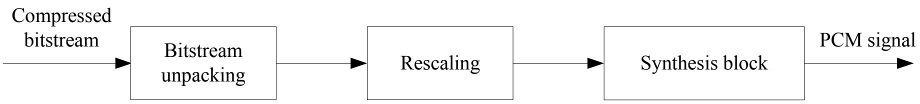
1.2.2 Audio encoding, decoding and transcoding

The main subject that is addressed in this thesis is the efficiency of audio compression systems. A typical audio encoder or decoder comprises several functional blocks, such as shown in Figure 1-1. During encoding, the PCM input is separated into its subband components by the analysis block. The subband components are then quantized and assembled by packing them into a compressed bitstream. The quantization process is controlled by a psychoacoustic model that analyzes the spectral content of the input, and determines the number of bits allocated to the quantization of each subband component. The quantized components are usually represented in the form of an integer factor and a scaling factor.

During decoding, the bitstream is first unpacked, then rescaled (by combining the integer and scaling factors), and the PCM output is finally reconstructed by the synthesis block. Filter banks are typically used in the implementation of the analysis and synthesis blocks. Different compression methods might use different types of filter banks. A more detailed explanation on the operation of the audio encoder and decoder, and the functions of their individual blocks can be found in Section 7.1.



(a) Typical audio encoder



(b) Typical audio decoder

Figure 1-1 Simplified overview of an audio encoder/decoder.

Audio transcoding refers, in a general sense, to the process whereby a compressed bitstream is re-encoded to another compressed bitstream. The re-encoded bitstream can be in a different compression method from the original, or it can have a different bit-rate, etc. The conventional method of transcoding is to first decode the compressed bitstream to a PCM signal, and then to re-encode this signal to the second compressed bitstream. i.e. the decoder is cascaded with the encoder. For example, a transcoding process is illustrated by the following: an MPEG 1 Layer 2 [71] song which is encoded at 192 kbps stereo, is decoded to the PCM format, by using an MPEG 1 Layer 2 decoder. This PCM signal is then encoded to an MP3 song at 128 kbps mono, using an MP3 encoder.

The conventional method of transcoding, as described above, presents several options for complexity reduction. For example, each individual block (in Figure 1-1) can be optimized separately: an efficient, modified psychoacoustic model can be used [106], an efficient FFT implementation can be used [11], optimized hardware programming language can be used to code the individual blocks, etc. A more detailed explanation of audio transcoding, and a discussion of the state-of-the-art pertaining to this subject, can be found in Section 7.2.

1.3 About our work

In this thesis, we consider the complexity of an audio transcoder, for the scenario of bit-rate alteration. We focus mainly on 2 areas of complexity reduction: (i) filter banks, and (ii) overall transcoder design. In the first part, we consider on a local scale, the use of efficient filter banks as a means of reducing the overall transcoder complexity. In the second part, we proceed to consider on a broader scale, the complexity reduction possibilities that are available, such as the elimination of the various processing blocks shown in Figure 1-1.

1.3.1 Filter banks

Part I of this thesis (Chapters 2-6) focuses on filter banks. A background on the state of the art, relevant to our work on filter banks, is provided in Chapter 3.

We propose several new filter bank structures for different operating conditions, such as for single-rate and multi-rate operation, and for scenarios where the input signal is real-valued or complex-valued. The proposed filter banks belong to a new class, which we call the Frequency Response Masking (FRM) [23] filter banks. The FRM filter banks are based on the Fast Filter Bank (FFB) [21].

Contributions

- The design considerations for the FFB are consolidated, and further expanded on. A general structure and working principle of the FFB was first proposed in [21]. We expand on this by performing an in-depth analysis on the design of the subfilters comprising the FFB (Chapter 3).
- A new structure, based on the FFB, is proposed by using the node-modulation method (Section 4.2).
- The pruning method is proposed for the scenario when the input to the FFB is real-valued (Section 4.3).
- A matrix formulation is proposed for the implementation of the FFB on systems optimized for vector and matrix processing (Section 4.5).

- The design of FRM filter banks for multi-rate processing is analyzed (Sections 6.1 and 6.2).
- A classification of FRM filter banks is proposed (Section 5.2).
- Efficient implementations of FRM filter banks for multi-rate processing are proposed (Sections 5.1 and 6.4). In Section 6.4, we design a very efficient multiplierless FRM filter bank, comprising subfilters having filter coefficients that are reduced to signed-powers-of-two terms.

Note that although efficient filter banks are useful in audio coding (for speeding up the audio encoding and decoding processes), we do not intend to restrict the application of these filter banks to only audio. Filter banks are widely applied in many areas of digital signal processing, and the proposed filter banks could be used in other areas, such as in communications.

1.3.2 Transcoding

The subject of transcoding is discussed in Part II: Chapters 7-10. A background on the state of the art, relevant to our work on transcoding, is provided in Chapter 7. We focus on the MPEG 1 Layer 2 and MP3 compression methods in this thesis.

The transcoding process is modeled as a cascaded quantization process (Section 7.4), which we use extensively in our work. The performance, in terms of audio quality and complexity, of a transcoder is then considered.

Contributions

- Tandem quantization error, which is a consequence of cascaded quantization, is analyzed. We propose several new methods of reducing this error (Chapter 8), such as the modified quantizer method and the quantizer selection method.
- We analyze the effect of different delays (measured from the input to the output of the transcoder) on the audio quality of transcoded materials. We develop a measure for estimating the impact of different delays when transcoding MPEG 1 Layer 2 and MP3 audio (Chapter 9).
- A very efficient MP3 transcoder is proposed (Chapter 10).

1.4 Thesis overview

The flow of the thesis is as follows.

Part I: Filter Banks - Chapters 2-6

In Chapter 2, we provide a general overview of frequently used filter banks, such as the polyphase filter bank and the octave filter bank. The Fast Filter Bank (FFB) [21], which belongs to a new class of FRM filter banks, is also introduced. A preliminary comparison of the characteristics of each type of filter bank is also provided.

In Chapter 3, we explain the operation of the FFB. Its design and properties are then considered. The FFB usually comprises a tree-structured cascade of subfilters, which have complex-valued coefficients and small filter lengths. The FFB is useful in designs requiring low complexity at very small transition widths. As an example, a 32-channel FFB design is given.

In Chapter 4, we propose several methods to reduce the complexity of the FFB. We consider both the scenarios when the input signal is real-valued, and when it is complex-valued. The node-modulation method is proposed as a method of reducing FFB complexity, by reducing its subfilters to having real-valued coefficients. The pruning method is proposed for the scenario of a real-valued input signal. As an example, we then design a 16-channel FFB using the pruning and node-modulation methods, and compare its complexity to the original FFB. A matrix formulation of the FFB is then proposed for software implementation on machines which have vector- or matrix-optimized capability (such as the matrix processing package BLAS [38]).

In Chapter 5, we consider the multi-rate operation of the FRM class of filter banks (when the outputs are decimated). A method of decimation is proposed that reduces the complexity of the filter bank in multi-rate mode. We categorize the FRM filter banks into 4 different types, based on the implementation of the subfilters. The different types of FRM filter banks are then compared in terms of complexity.

In Chapter 6, we analyze the reconstruction characteristics of the FRM filter banks in the multi-rate mode, when the analysis filter bank is cascaded with the synthesis filter bank. The design of filter banks with small aliasing and distortion error is considered. A filter bank with variable bandwidths is then designed. The coefficients of its subfilters are reduced to signed-powers-of-two (SPT) [45] terms. The resultant complexity is shown to be lower than that of the octave filter bank.

Part II*:* Audio Transcoding - Chapters 7-10

In Chapter 7, we introduce the concept of audio coding and audio transcoding, which we model as a cascaded quantization process. The terms and notational conventions which are frequently used for Part II are defined.

In Chapter 8, we analyze transcoding as a cascaded quantization process. When the input-to-output delay of the transcoder is a multiple of the decimation factor, cascaded quantization leads to an error component which we call 'tandem quantization error'. This is analyzed for the MPEG 1 Layer 2 method, and several methods of reducing the tandem quantization error are proposed.

In Chapter 9, we study the effect of different input-to-output delays on the quality of transcoded audio material. Both the MPEG 1 Layer 2 and MP3 methods are discussed.

In Chapter 10, a very efficient transcoder is proposed. Our method uses an integer-to-integer mapping. Since no floating-point computations are required, the transcoder is suitable for fast implementations that require a small software memory footprint (or hardware chip area). A prototype transcoder is implemented using the C programming language, and results of listening tests (in terms of audio quality) and execution speed tests are included. A patent application [104] was filed on the proposed method of transcoding.

1.5 Publications

- J.W. Lee and Y.C. Lim, "Efficient Implementation of Real Filter Banks using Frequency Response Masking Techniques", *IEEE Asia Pac Conf. on Circuits and Systems*, vol. 1, pp 69-72, Oct. 2002.
- J.W. Lee and Y.C. Lim, "Designing the Fast Filter Bank with a Minimum Complexity Criterion", *IEEE Int. Symp. on Signal Processing Applications*, vol. 2, pp 279-282, July 2003.
- Y.C. Lim and J.W. Lee, "Matrix Formulation: Fast Filter Bank", *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, vol. 5, pp 133-136, May 2004.
- J.W. Lee and Y.C. Lim, "A multiplierless filter bank with deep stopband suppression and narrow transition width", *IEEE Int. Symp. on Circuits and Systems*, pp. 4305-4308, May 2005.
- J.W. Lee, A. Lemma, M. van der Veen, "An analysis of tandem error during audio transcoding", *AES 117th Convention*, Preprint Number 6198, October 2004.
- J.W. Lee, Y.C. Lim, S.H. Ong, "A flexible and efficient sharp filter bank architecture for variable bandwidth systems", *IEEE Int. Symp. on Circuits and Systems*, pp. 2029-2032, May 2006.
- J.W. Lee and Y.C. Lim, "Efficient implementation of the Fast Filter Bank for critically decimated systems", *IEEE Asia Pac. Conf. on Circuits and Systems*, pp. 551-554, Dec 2006.
- [Patent application] J.W. Lee, W. Oomen, F. de Bonts, "Method and device for transcoding", Philips internal reference number PHNL040946EPP, Application date: 31 Aug 2004.

PART I:
FILTER BANKS

Chapter 2

General Overview of Filter Banks

In this chapter, we provide a brief review of the various filter bank implementations that are typically used in the field of audio compression. Due to its high efficiency in multirate systems, the polyphase filter bank is the most widely applied filter bank in audio compression, communications systems and general signal processing applications. A multirate system is defined as a system where different sampling rates exist in different parts of the system. A single-rate system is defined as a system where the sampling rate remains constant throughout. The octave-type filter bank is also included in our discussion, and is often applied in the areas of speech and image processing. A brief overview of the Fast Filter Bank (on which we will be focusing our work) is also provided. The polyphase and octave-type filter banks provide the basis of our subsequent complexity comparisons with the Fast Filter Bank.

2.1 Filter bank overview

The perceptual model used in audio compression techniques is based on the human ear's varying sensitivity to noise at different frequencies. Most audio coders utilize a filter bank to separate an audio signal into different subbands, which are then separately quantized. The perceptual model, the audio compression techniques and the quantization processes are described in greater detail in Part II. In Part I, we focus on filter banks.

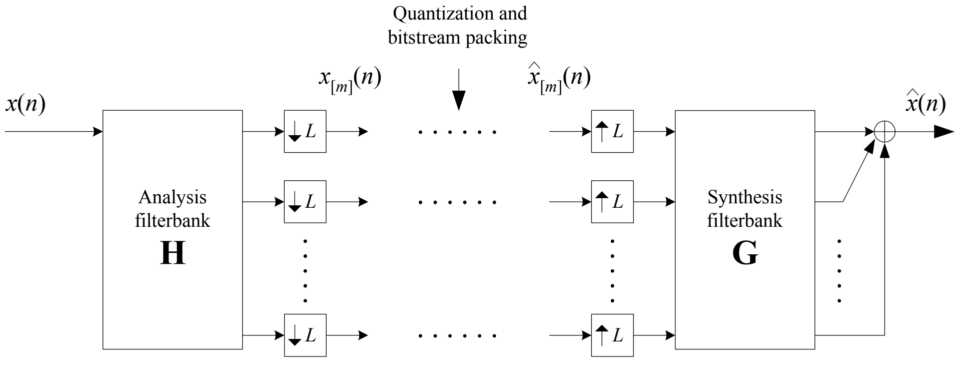


Figure 2-1 Typical model of audio compression using a pair of analysis-synthesis filter banks.

A typical application of filter banks in audio compression is shown in Figure 2-1. We use **H** to represent the set of analysis filter transfer functions $H_{[m]}(z)$ for the subband *m*, where $0 \le m \le M-1$, *M* denotes the total number of subbands, and **G** represents the set of synthesis filter transfer functions $G_{[m]}(z)$. In our work, we only consider Finite Impulse Response (FIR) filters and filter banks. The input signal to the analysis filter bank is denoted as $x(n)$.

The *z*-transform of $x(n)$ is notated by $X(z)$, where:

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n} . \tag{2.1}$$

Similarly, $H_{[m]}(z)$ and $G_{[m]}(z)$ are implicitly understood to be the *z*-transforms of $h_{[m]}(n)$ and $g_{[m]}(n)$ respectively. For a FIR filter, the coefficients of the transfer function $H_{[m]}(z)$ are equal to the values of the filter impulse response $h_{[m]}(n)$.

The outputs of the analysis filter bank are assumed to be decimated by a factor of *L*, and are subsequently interpolated by a factor of *L* before reconstruction [3]. If $L = 1$, then the system is said to be single-rate. If $L < M$, the system is oversampled, and if $L = M$, then the system is critically-decimated.

The filtered version of the input signal for the *m*-th subband, after passing through the analysis filter bank, is given by:

$$x_{[m]}(n) = h_{[m]}(n) * x(n) , \tag{2.2}$$

where $*$ represents the convolution operator. The signal $x_{[m]}(n)$ is then quantized to:

$$\hat{x}_{[m]}(n) = q_{[m]}\left(x_{[m]}(n)\right) , \tag{2.3}$$

where $q_{[m]}(.)$ represents the quantization used in the subband *m*.

The quantization error in each subband is denoted by:

$$e_{[m]}(n) = \hat{x}_{[m]}(n) - x_{[m]}(n) , \tag{2.4}$$

and the reconstructed signal is:

$$\hat{x}(n) = \sum_{m=0}^{M-1} g_{[m]}(n) * \hat{x}_{[m]}(n) . \tag{2.5}$$

In the scenario when there is no quantization error, $\hat{x}(n)$ is related to $x(n)$ by the filter bank pair **G** and **H** only. If $\hat{x}(n)$ is a delayed version of $x(n)$, i.e. $\hat{x}(n) = x(n-d)$, where *d* is the total delay of the filter bank pair, then the filter bank pair is said to be perfectly reconstructing (PR). If $\hat{x}(n) \approx x(n-d)$ to a very good approximation, then the filter bank pair is said to be near-perfectly reconstructing (NPR).

Our objective in Part I of this thesis is to study the complexity-related issues of current filter banks used in audio compression techniques. Also, we examine the possibility of using the Fast Filter Bank (FFB) as an efficient alternative. The FFB is a new category of filter banks that is designed using the Frequency Response Masking (FRM) Technique. We provide an in-depth coverage of the design and implementation issues of the FFB, and propose several architectural improvements and complexity reductions. We first review the filter banks that are commonly used.

2.2 Definitions and conventions

2.2.1 Indices used

Due to the frequent references to multi-stage filter banks, which consist of many subfilters arranged in a 2-dimensional structure, we use the symbols u and v mainly as horizontal and vertical indices of these subfilters respectively. The symbol m is used as the subband index. When m is used in transfer functions and impulse responses, it is typically enclosed in square brackets to avoid confusion with other indices, e.g. $H_{[m]}(z)$.

2.2.2 Filter specifications

In this thesis, we will be concerned with low-pass, high-pass and band-pass filters. By way of illustration, we consider the archetypal low-pass characteristics in Figure 2-2.

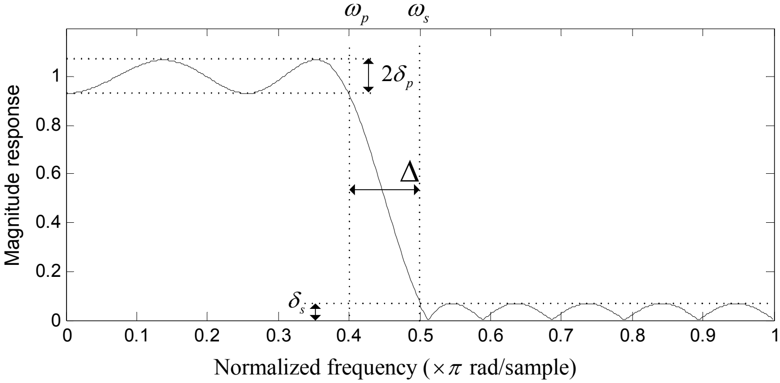


Figure 2-2 Frequency response of a generic filter.

Sampling frequency: In our work, we assume that the sampling frequency is normalized to 2π radians/sample.

Passband edge/stopband edge: These are denoted by the symbols ω_p and ω_s respectively and are shown in Figure 2-2.

Peak passband/stopband ripple: These are denoted by the symbols δ_p and δ_s respectively and are shown in Figure 2-2.

Note that the peak passband ripple is taken about a gain of unity, and the peak stopband ripple is taken about a gain of zero. Hence, equal values of δ_p and δ_s does not necessarily equate to equal values in terms of dB. For example, $\delta_p = 0.01 = 0.086\,\text{dB}$ and $\delta_s = 0.01 = -40\,\text{dB}$.

Stopband attenuation: The value of the average passband gain divided by the peak stopband ripple, in dB.

Transition width: The difference in frequency between the passband edge and the stopband edge and is denoted by Δ (Figure 2-2). These are expressed in the form $b\pi$ radians/sample, as was explained in the paragraph on sampling frequency.

Bandpass/highpass filters: The same set of specifications apply for highpass filters, except that the passband is centered at the frequency π, instead of zero. The passband edge and stopband edge are thus reversed laterally. For a bandpass filter with bandwidth ω_B, centred at frequency ω_C, its passband edges are at $(\omega_C + \omega_B / 2)$ and $(\omega_C - \omega_B / 2)$. Its stopband edges are at $(\omega_C + \Delta + \omega_B / 2)$ and $(\omega_C - \Delta - \omega_B / 2)$. The peak passband/stopband ripple are defined as for the lowpass filter.

2.2.3 Complex/real scenarios for filter bank

First, we make the distinction between the transfer functions of the filter bank and the nature of the inputs and outputs of the filter bank.

The input signal to the filter bank can be real-valued or complex-valued. We begin our discussion by assuming the general case that the input signal is complex-valued (Chapters 2 and 3). Since our work is applied to audio, the case of a real-valued input signal is relevant. In Chapter 4, we make the transition from the complex-valued input signal case to the real-valued input signal case, and propose methods to reduce filter bank complexity for the real-valued input signal case. In Chapters 5 and 6, we consider the case when the input signal is real-valued.

The transfer functions of the filter bank can have coefficients that are real-valued or complex-valued. Transfer functions with purely real-valued coefficients have frequency responses that are symmetrical about the frequency origin (e.g. (a)). Transfer functions that have complex-valued coefficients have non-symmetrical frequency responses about the origin (e.g. (b)). In Chapters 2 and 3, we assume the general case when the filter bank transfer functions have complex-valued coefficients.

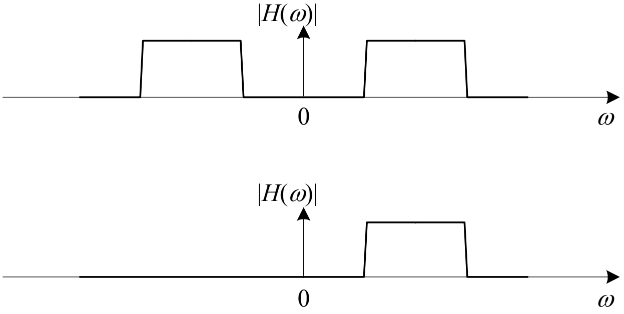


Figure 2-3 Frequency response for a (a) transfer function with purely real-valued coefficients; (b) transfer function with complex-valued coefficients.

When the input and output signals are both real-valued, a transfer function with purely real-valued coefficients is sufficient. We would like to clarify that even when the transfer function from the input to the output has purely real-valued coefficients, the transfer functions of the subfilters that make up the filter bank may represent either case (real-valued, or complex-valued coefficients). In Chapters 5 and 6, we use filter banks with transfer functions that have real-valued coefficients. These filter banks comprise subfilters with transfer functions that have complex-valued coefficients. In Chapter 4, both cases are considered.

2.2.4 Glossary and Appendix information

Frequently used notational conventions are provided in the Glossary.

In Appendix A, we provide information on methods of filter length estimation. Estimated filter length is often used when we compare different filter bank implementations in Part I of this thesis. In Appendix B, we define our methods for calculating the complexity of a filter for different scenarios. For example, one such scenario is a complex-valued input signal, and a linear phase FIR filter with real-valued coefficients. The information in Appendix B is used when we calculate complexity in Part I of this thesis.

2.2.5 Causality

Causal form: When it is not stated, we assume the causal form of the filters in the diagrams and equations given in this thesis. The causal form of a filter with transfer function $H(z)$ is defined to have coefficients $h(n)$, where $0 \le n \le N-1$ and N is the filter length of $H(z)$.

Non-causal form: For the sake of representation simplicity, we occasionally assume the non-causal form of the filters where stated. The reason for using the non-causal form is so that the delay of an odd-length linear-phase FIR filter is equal to zero. Odd-length filters with length N and impulse response $h(n)$ are defined for $-(N-1)/2 \le n \le (N-1)/2$, where the center coefficient is located at n=0. Even-length filters with length N and impulse response $h(n)$ are defined for $-(N/2-1) \le n \le N/2$.

2.3 Filter-array filter bank

Before discussing the various filter bank implementation strategies, let us first examine the basic filter-array. The filter-array analysis and synthesis filter banks (Figure 2-4) consist of an array of bandpass filters $H_{[m]}(z)$ and $G_{[m]}(z)$, which have passbands in different parts of the frequency spectrum.

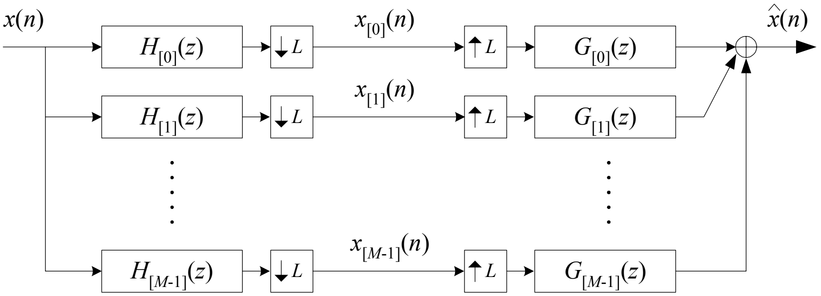


Figure 2-4 Filter-array for an analysis-synthesis filter bank pair.

In much of our work, we omit the discussion of the synthesis filter bank. Instead, we focus on the design of the analysis filter bank. In places where the synthesis filter bank is not mentioned, we can simply assume that the synthesis filter bank is the mirror image of the analysis filter bank, i.e. $H_{[m]}(z)=G_{[m]}(z)$.

2.3.1 *Uniform filter bank*

A uniform filter bank is defined as one where the frequency responses of the subbands have equal bandwidths, transition widths, passband and stopband ripples. One such example is when the frequency response of the *m*-th subband filter is a modulated version of the lowpass filter $H_{[0]}(z)$, i.e.:

$$H_{[m]}(z) = H_{[0]}(e^{-j2\pi m/M} z), \text{ for } 1 \le m \le M-1. \tag{2.6}$$

2.3.2 *Even-stacked and odd-stacked filter bank*

If the passband of the subband *m* has center frequency at $2m\pi/M$ (Figure 2-5(a)), then the subbands of the filter bank are said to be even-stacked. An odd-stacked filter bank on the other hand, has center frequencies located at $(2m+1)\pi/M$ (Figure 2-5(b)).

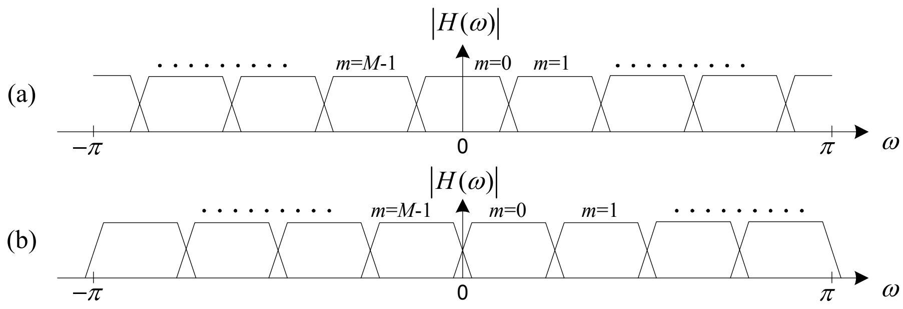


Figure 2-5 Frequency responses of (a) even-stacked, and (b) odd-stacked filter banks.

2.3.3 Complexity

We estimate complexity as the approximate number of operations required per input sample that is processed into the filter bank. For a decimation factor of *L*, *L* input samples are processed into the filter bank per output sample. The approximate number of operations required for one output sample (*L* input samples) is *MN*, and the approximate number of operations required for one input sample is therefore *MN*/*L*. Accordingly, the complexity of the filter-array filter bank is:

$$\Gamma_A = MN/L, \tag{2.7}$$

where we assumed a real-valued input signal and real-valued filter coefficients. A general discussion on our approach to calculating complexity for a complex-valued input signal or complex-valued coefficients is provided in Appendix B.

The filter-array is very inefficient in a practical situation, especially when the number of subbands is large. As can be seen from (2.7), the number of multiplications required scales proportionally upwards with the number of subbands.

An advantage of the filter-array is that each filter can be designed to have a frequency response that is independent of the others. Thus, it is suitable for the implementation of non-uniform filter banks with highly customized specifications, such as the case when each subband has a different bandwidth, stopband attenuation, etc.

2.4 Polyphase filter bank

An efficient method for filter bank implementation is the polyphase filter bank [4]-[8]. For an analysis filter bank with uniform subbands, the transfer function for subband *m* can be expressed as a modulated version of the lowpass prototype filter with transfer function $H_{[0]}(z)$ as shown in (2.6).

2.4.1 Polyphase decomposition

We first decompose the transfer function of the lowpass prototype filter in terms of its v-th polyphase component $\underset{\sim}{H}_v(z)$, which is annotated with a tilde underline, as follows:

$$H_{[0]}(z) = \sum_{v=0}^{M-1} z^{-v} \underset{\sim}{H}_v(z^M)\,, \tag{2.8}$$

where the coefficients of the v-th polyphase component can be found from the time-domain impulse response of the prototype filter:

$$\underset{\sim}{h}_v(n) = h_{[0]}(nM+v)\,, \text{ for } 0 \le v \le M-1\,. \tag{2.9}$$

Note: We chose v as the index here, instead of the usual convention as in [5], to conform to our convention of assigning v as a vertical index for filter banks.

The transfer function for subband m can be expressed in terms of the polyphase components $\underset{\sim}{H}_v(z)$:

$$H_{[m]}(z) = \sum_{v=0}^{M-1} e^{-j2\pi vm/M} z^{-v} \underset{\sim}{H}_v(z^M)\,. \tag{2.10}$$

In matrix form, this is expressed as:

$$\begin{bmatrix} H_{[0]}(z) \\ H_{[1]}(z) \\ \vdots \\ H_{[M-1]}(z) \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{-j2\pi/M} & & e^{-j2\pi(M-1)/M} \\ \vdots & & e^{-j2\pi vm/M} & \vdots \\ 1 & e^{-j2\pi(M-1)/M} & \cdots & e^{-j2\pi(M-1)^2/M} \end{bmatrix} \begin{bmatrix} z^{-0}\underset{\sim}{H}_0(z^M) \\ z^{-1}\underset{\sim}{H}_1(z^M) \\ \vdots \\ z^{-(M-1)}\underset{\sim}{H}_{M-1}(z^M) \end{bmatrix}. \tag{2.11}$$

From (2.11), since the M filters share a common set of polyphase components which are modulated by an M-point inverse discrete fourier transform (IDFT), they can be conveniently implemented using the structure illustrated in Figure 2-6. In the figure, we assumed an arbitrary decimation factor of L. If $L = M$, then the decimation block can be brought to the front of the polyphase subfilters.

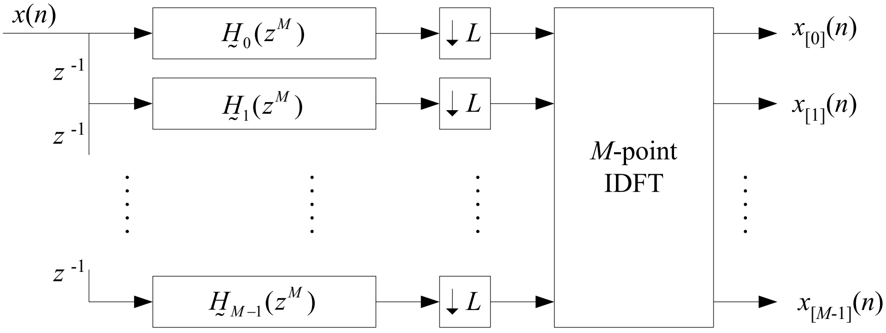


Figure 2-6 Polyphase structure for a M-channel analysis filter bank.

A variation of the above, that is often used for real-valued outputs, is the cosine-modulated filter bank (CMFB). An efficient polyphase structure can be obtained by decomposing the prototype filter into 2M polyphase components, followed by a M by 2M cosine modulation matrix **T**, comprising the elements t_{mv} given by:

$$t_{mv} = 2\cos\left(\frac{\pi}{M}(m+0.5)(v-\frac{N-1}{2})+(-1)^m\frac{\pi}{4}\right). \qquad (2.12)$$

The prototype filter has the impulse response $h(n)$, transfer function $H(z)$, and a filter length of N. The coefficients of its v-th polyphase component are given by:

$$\underset{\sim}{h}_v(n) = h(2nM+v)\,, \text{ for } 0 \le v \le 2M-1\,. \qquad (2.13)$$

The transfer function of the m-th subband of the CMFB takes the form:

$$H_{[m]}(z) = \sum_{v=0}^{2M-1} t_{mv} z^{-v} \underset{\sim}{H}_v(-z^{2M})\,. \qquad (2.14)$$

It is also interesting to note that the modified discrete cosine transform (MDCT) is a special case of the CMFB, when the prototype filter has a filter length of 2M and the outputs are critically decimated. The MDCT is used in many audio compression methods such as MP3, AAC and WMA.

2.4.2 Complexity

The complexity of the polyphase structure is determined by that of a single prototype filter, with transfer function $H_{[0]}(z)$, and that of an M-point Fast Fourier Transform (FFT)/MDCT. For a polyphase structure, the complexity scales inverse-proportionally with the decimation factor:

$$\Gamma_P\left(M,L\right) = \frac{\Gamma_{H_{[0]}} + \Gamma_{IDFT/MDCT}\left(M\right)}{L}, \qquad (2.15)$$

where L is the decimation factor, $\Gamma_{H_{[0]}}$ is the complexity of the prototype lowpass filter and $\Gamma_{IDFT/MDCT}\left(M\right) = \mathrm{O}(M\log_2 M)$ is the complexity of the M-point IDFT/MDCT.

The filter length of the prototype filter can be estimated (refer to Appendix A). Due further to many well-known efficient methods of computing the IDFT/MDCT using the FFT or IFFT, the polyphase filter bank can be very efficiently implemented. Assuming that M is a power of 2, the complexity of an M-point IDFT is:

$$\Gamma_{IDFT} = 2M\log_2 M\,. \qquad (2.16)$$

Various methods have also been proposed to compute the M-point MDCT very efficiently ([10]-[12]). In [11], an efficient method was proposed with the following number of multiplications for an M-point MDCT:

$$\Gamma_{MDCT}\left(M\right) = 0.5M\log_2 M \,. \tag{2.17}$$

The filter length of $H_{[0]}(z)$, given by *N*, is usually much greater than *M*. Therefore, $\log_2 M << (N-1)$ and $\Gamma_P << \Gamma_A$. Thus the polyphase filter bank has a much lower complexity than the filter-array implementation.

2.5 Octave filter bank

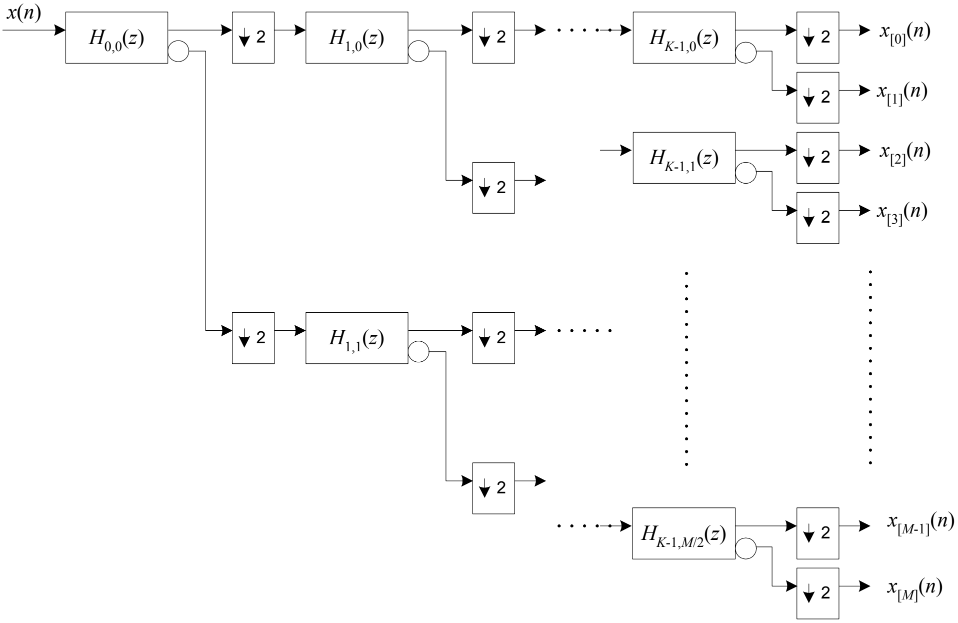


Figure 2-7 Equal bandwidth octave filter bank comprising two-channel filter banks.

A two-channel filter bank can be cascaded into a tree-structured format to obtain an octave filter bank [13]. Figure 2-7 shows the structure of an equal bandwidth octave filter bank with *K* stages. At each stage, the input signal is filtered into its highpass and lowpass components and then decimated by a factor of 2. Therefore, the outputs of the filter bank are critically decimated (*L* = *M*). A key advantage of this structure is the ease of a PR design. As long as the PR property is satisfied for each 2-channel subfilter, the analysis-synthesis filter bank pair is PR. Literature on the design of 2-channel filter banks satisfying the PR property is relatively abundant [14]-[18].

Each subfilter has 2 outputs: the upper output path has a transfer function $H_{u,v}(z)$, and the lower output path (marked with a circle symbol) has a transfer function $\overline{H}_{u,v}(z)$. The transfer function $\overline{H}_{u,v}(z)$ is complementary to $H_{u,v}(z)$ in its frequency response: if $H_{u,v}(z)$

is lowpass, then $\bar{H}_{u,v}(z)$ is highpass. The 2-channel analysis filter bank (with transfer functions $H_{u,v}(z)$ and $\bar{H}_{u,v}(z)$) and the corresponding 2-channel synthesis filter bank (with transfer functions $G_{u,v}(z)$ and $\bar{G}_{u,v}(z)$) form a 2-channel analysis-synthesis pair.

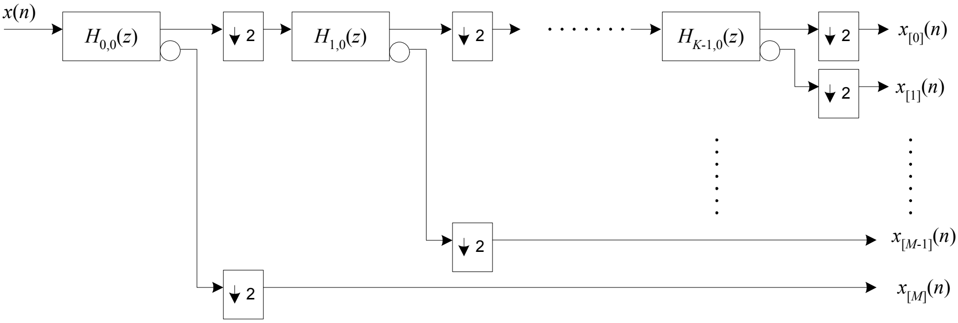
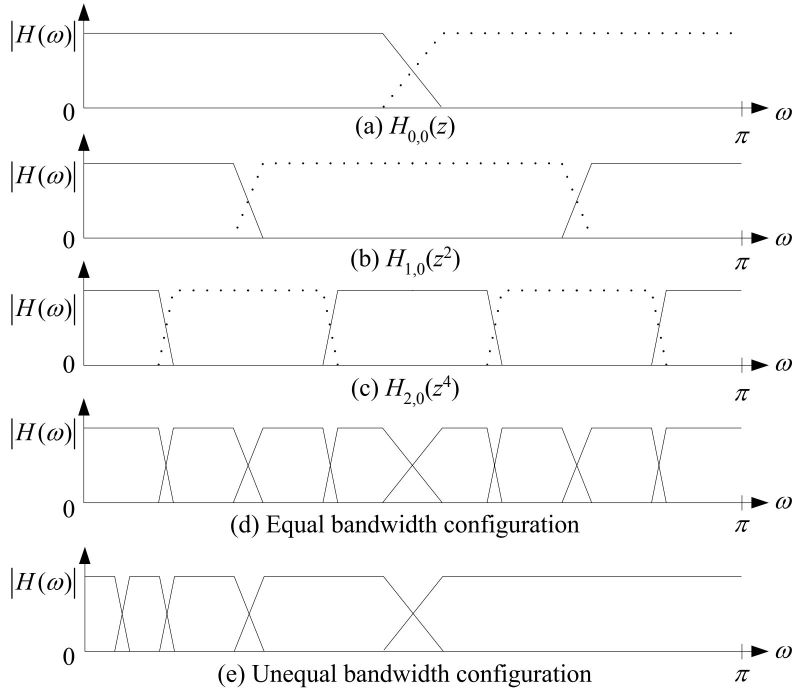


Figure 2-8 Octave filter bank in an unequal bandwidth configuration.

Figure 2-8 shows a modified implementation of the octave filter bank which is used in applications such as speech coding [19]-[20]. The outputs of this filter bank have unequal bandwidths, i.e. some subbands have larger bandwidths than others. The octave filter bank shown in Figure 2-8 has a larger bandwidth at higher frequencies, and a smaller bandwidth at lower frequencies. Its frequency response is shown in Figure 2-9(e). The subbands with small bandwidths correspond to the frequencies where the human ear is more sensitive to noise, and the subbands with large bandwidths correspond to the frequencies where the human ear is less sensitive to noise. In such a scenario, the number of subbands for an unequal bandwidth filter bank is less than the number of subbands for an equal bandwidth filter bank, assuming that the subband with the smallest bandwidth has the same bandwidth as each subband of the equal bandwidth filter bank (e.g. Figure 2-9(d) and Figure 2-9(e)). Therefore, less operations are required for processing the decreased number of subbands, e.g. compression, quantization, etc.

Let us represent the transfer function from the input $x(n)$ to the output $x_{[m]}(n)$ of the octave filter bank as $H_{[m]}(z)$. These transfer functions are related to the transfer functions of the subfilters depending on the filter path taken. As an illustration, for the 8-channel equal bandwidth octave filter bank using the structure shown in Figure 2-7, $H_{[0]}(z) = H_{0,0}(z)H_{1,0}(z^2)H_{2,0}(z^4)$ and $H_{[2]}(z) = H_{0,0}(z)\bar{H}_{1,0}(z^2)H_{2,0}(z^4)$.



(a) $H_{0,0}(z)$

(b) $H_{1,0}(z^2)$

(c) $H_{2,0}(z^4)$

(d) Equal bandwidth configuration

(e) Unequal bandwidth configuration

Figure 2-9 Frequency responses of the octave filter bank.

The solid lines in Figure 2-9(a)-(c) represent the frequency responses of $H_{0,0}(z)$, $H_{1,0}(z^2)$ and $H_{2,0}(z^4)$ respectively, if we assume that the same base filter is used for each subfilter, i.e. $H_{0,0}(z) = H_{1,0}(z) = H_{2,0}(z)$. The dotted lines represent the complementary frequency responses of $\overline{H}_{0,0}(z)$, $\overline{H}_{1,0}(z^2)$ and $\overline{H}_{2,0}(z^4)$ respectively. Figure 2-9(d) shows the frequency responses of $H_{[m]}(z)$ for the 8-channel equal bandwidth octave filter bank using the structure shown in Figure 2-7. We note that although the bandwidths of the different subbands are equal, their transition widths are unequal. Since $H_{2,0}(z^8)$ is interpolated from $H_{2,0}(z)$, the transition edges in Figure 2-9(d) corresponding to the transition edges of $H_{2,0}(z^8)$ have smaller widths, and the transition edges corresponding to the transition edges of $H_{0,0}(z)$ have larger widths. Figure 2-9(e) shows the frequency responses of an unequal bandwidth octave filter bank corresponding to the structure in Figure 2-8.

The equal bandwidth octave filter bank is considered only for interest, and is typically not used in the implementation of uniform filter banks. For the case shown in Figure 2-9(d), in order that the transition widths of the different subbands are equal, the filter lengths of $H_{0,0}(z)$ and $H_{1,0}(z)$ must be increased. For a filter bank with equal transition widths, the filter lengths of $H_{u,v}(z)$ increases as u decreases, i.e. $N_{0,v} > N_{1,v} > N_{2,v}$.... (where $N_{u,v}$ is the filter length of $H_{u,v}(z)$). From Figure A-2, we observe that filter length increase drastically as the transition width becomes smaller.

Typically, for the design of equal bandwidth octave filter banks, $H_{u,v}$ is selected to be equal to $H_{u,0}$. Hence, the overall complexity of the octave filter bank can be estimated from:

$$\Gamma_O = \sum_{u=0}^{K-1} N_{u,0} \, . \tag{2.18}$$

Example: In an illustration of the application of the octave filter bank to the design of uniform filter banks, let us consider a 128-channel octave filter bank with equal transition widths of $\pi/320$. Assume that the passband and stopband ripples for all subfilters are equal to 0.086 dB and -80 dB respectively.

The transition widths for the subfilters are:

$$\Delta_{0,v}=\pi/320,\ \Delta_{1,v}=\pi/160,\ \Delta_{2,v}=\pi/80,\ \Delta_{3,v}=\pi/40,$$

$$\Delta_{4,v}=\pi/20,\ \Delta_{5,v}=\pi/10,\ \Delta_{6,v}=\pi/5.$$

The lengths of the subfilters are estimated from (A.1):

$$N_{0,v}\text{=}2061,\ N_{1,v}\text{=}1031,\ N_{2,v}\text{=}516,\ N_{3,v}\text{=}258,\ N_{4,v}\text{=}129,\ N_{5,v}\text{=}65,\ N_{6,v}\text{=}33.$$

It is observed from the example that the lengths of the subfilters can be very large for a uniform filter bank implementation with many subbands.

In this and the next few chapters, our comparisons mainly concern equal bandwidth filter banks. Section 6.4 will be dedicated to the efficient design of unequal bandwidth filter banks.

2.6 Fast Filter Bank (FFB)

The Fast Filter Bank (FFB) [21] is a tree-structured filter bank and its structure is shown in Figure 2-10. In this section, we provide a very simplified introduction to the structure of the FFB and its notational conventions, as well as a brief discussion of its computational complexity. An in-depth coverage of the working principle and design of the Fast Filter Bank will be provided in Chapter 3. Improvements to the efficiency of the FFB will be proposed and analyzed in subsequent chapters.

Note: The original motivation behind the naming of the Fast Filter Bank (FFB) was due to it being a generalized form of the sliding FFT filter bank [31]. When the subfilters of the FFB are reduced to a length of 2, and having the transfer function of $H_{u,v}(z) = 1 + e^{-j2\pi\tilde{v}/M} z^{-1}$, the FFB becomes a sliding FFT filter bank.

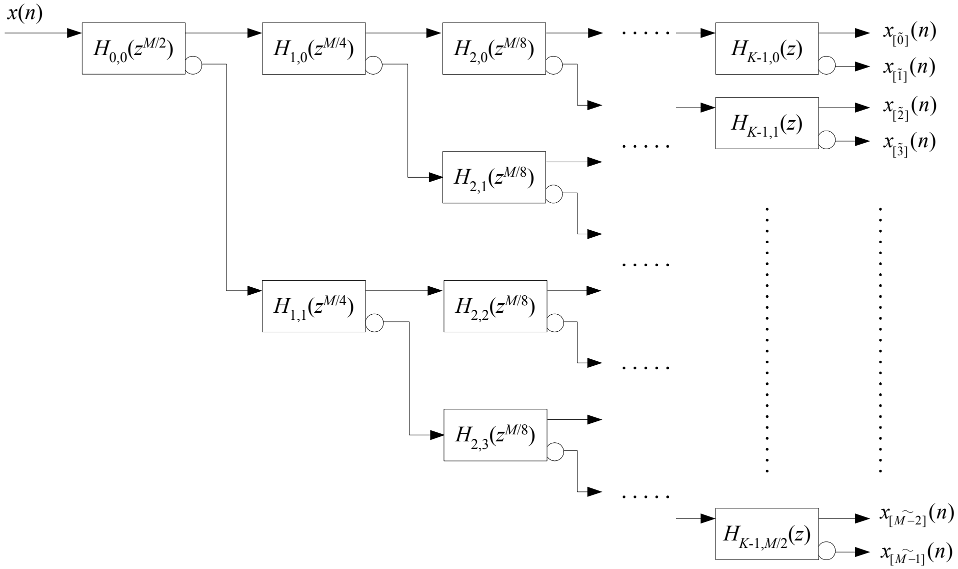


Figure 2-10 The tree-structured Fast Filter Bank.

In Figure 2-10, the FFB is shown, comprising a K-stage cascade of subfilters with transfer functions $H_{u,v}(z)$. The number of subbands is given by $M = 2^K$.

Although an initial glance at the FFB seems to suggest a great similarity to the octave filter bank discussed in the previous section, their methods of operation are in fact fundamentally different. For the octave filter bank, each subfilter is a 2-channel filter bank that separates the input signal into its lowpass and highpass components. For the FFB on the other hand, the subfilter with transfer function $H_{0,0}(z)$ is interpolated to obtain a frequency response with multiple passbands. The subfilters with transfer functions $H_{u,0}(z)$ for $u>0$, then act to mask the unwanted passbands such that the outputs of the FFB have a single passband for each subband. The frequency response masking aspect of the FFB is further explained in Chapter 3.

Example: In an illustration of the lengths of the subfilters normally used in the FFB, let us consider a 128-channel FFB with equal transition widths of $\pi/320$. Assume that the passband and stopband ripples for all subfilters equal to 0.086 dB and -80 dB respectively.

The transition widths for the subfilters are:

$$\Delta_{0,v}=0.2\pi,\ \Delta_{1,v}=0.4\pi,\ \Delta_{2,v}=0.7\pi,\ \Delta_{3,v}=0.85\pi,$$

$$\Delta_{4,v}=0.925\pi,\ \Delta_{5,v}=0.9625\pi,\ \Delta_{6,v}=0.9812\pi.$$

The minimum filter lengths required for the subfilters are:

$$N_{0,v}=31,\ N_{1,v}=14,\ N_{2,v}=6,\ N_{3,v}=4,\ N_{4,v}=3,\ N_{5,v}=2,\ N_{6,v}=2.$$

From the example, it can be seen that the FFB typically comprises subfilters with very short filter lengths. This is due to the decreased transition width requirements on the subfilters by using the FRM technique. As a result, the FFB has a very low complexity when operated in single-rate mode (this is presented in the next section). The complexity of the FFB can be generally described by:

$$\Gamma_F = N_{0,0} + 4\sum_{u=1}^{K-1} 2^u N_{u,0} \,. \tag{2.19}$$

When the outputs of the FFB are decimated by a factor of L, its complexity does not scale downward proportionally to L (such as when compared to the polyphase filter bank (2.16)). The reason for this is that the subfilters in the last stage, given by $H_{K-1,v}(z)$, are not interpolated and thus we are unable to bring the decimation factor towards the front of the filter bank (this is explained in Appendix B). In Chapters 5 and 6, we shall propose methods to improve the efficiency of the FFB in multirate operation.

2.7 Filter bank comparisons

It is useful at this point to have an approximate idea of the relative complexities of the various filter bank implementations. In this comparison, we consider uniform filter banks with $M = 2^K$ uniform subbands and transition widths Δ. Table 2-1 shows an approximate comparison of the computational complexities for the various filter bank implementations.

Filter bank	Approximate complexity
Filter-array	$\Gamma_A = MN/L$
Polyphase	$\Gamma_P = N + 2M\log_2 M/L$
Octave	$\Gamma_O = \sum_{u=0}^{K-1} N_{u,0}$, when $L=M$
FFB	$\Gamma_F = N_{0,0} + 4\sum_{u=1}^{K-1} 2^u N_{u,0}$

Table 2-1 General complexity comparison.

Example - Comparison for L=1, $\delta_p = \delta_s$ *=0.001:*

Let us first consider the single-rate case, i.e. L=1. For an example of $\delta_p = \delta_s$=0.001 (0.0087 dB and -60 dB respectively), we tabulate the estimated complexity in Table 2-2 for various transition widths and numbers of subbands, for the polyphase filter bank and the FFB.

		Polyphase filter bank		Fast Filter Bank	
M	Δ	N	Γ_P	$N_{0,0}$	Γ_F
16	$\pi/32$	202	330	25	352
	$\pi/64$	402	530	50	339
	$\pi/128$	804	932	101	372
	$\pi/256$	1607	1735	201	465
32	$\pi/64$	402	722	25	497
	$\pi/128$	804	1124	50	477
	$\pi/256$	1607	1927	101	507
	$\pi/512$	3213	3533	201	598
128	$\pi/256$	1607	3399	25	1105
	$\pi/512$	3213	5005	50	1073
	$\pi/1024$	6426	8218	101	1096

Table 2-2 Complexity comparisons between the uniform polyphase filter bank and the FFB in single-rate mode (L=1).

From Table 2-2, we can observe that the filter length N and complexity of the polyphase filter bank increase significantly when the transition width is decreased. For the FFB, on the other hand, the filter length of $H_{0,0}(z)$ is much smaller. Furthermore, the complexity of the FFB does not increase much for very small transition widths. For single-rate filter banks with large M and very small transition widths, the FFB is clearly more efficient than the polyphase filter bank.

Example - Comparison for L=M, $\delta_p = \delta_s$ *=0.001:*

Consider the critically decimated case, i.e. L=M. For the same example of $\delta_p = \delta_s$=0.001 (0.0087 dB and -60 dB respectively), we tabulate the estimated complexity for various transition widths and numbers of subbands for the polyphase filter bank, octave filter bank and the Fast Filter Bank in Table 2-3.

As mentioned earlier, the octave filter bank is unsuitable for the implementation of uniform filter banks, because the lengths of some subfilters become prohibitively large. The example clearly illustrates this. The FFB has a high complexity compared to the polyphase filter bank in the critically-decimated case. However, its complexity is still significantly lower than that of the uniform octave filter bank.

Note: The comparisons given above are based on a general estimation of the complexity of the filter banks. In reality, the complexity may be influenced by other factors: e.g., fixed or floating-point implementation, complexity reduction strategies specific to the type of filter bank, etc.

The polyphase filter bank is very efficient, and popularly used for the implementation of uniform filter banks (such as in communications and audio compression). For the implementation of non-uniform filter banks, the octave filter bank is efficient and popularly used (such as in speech and image processing). In our work, we compare the FFB to the polyphase filter bank for the case of uniform filter banks, and to the octave filter bank for the case of non-uniform filter banks.

M	Δ	Polyphase, Γ_P	Octave, Γ_O	FFB, Γ_F
16	$\pi/32$	21	378	352
	$\pi/64$	33	756	339
	$\pi/128$	58	1509	372
	$\pi/256$	108	3015	465
32	$\pi/64$	23	780	497
	$\pi/128$	35	1560	477
	$\pi/256$	60	3116	507
	$\pi/512$	110	6229	598
128	$\pi/256$	27	3191	1105
	$\pi/512$	39	6380	1073
	$\pi/1024$	64	12756	1096

Table 2-3 Complexity comparisons between the uniform polyphase filter bank, octave filter bank and the FFB in critically-decimated mode (*L*=*M*).

2.8 Summary

In this chapter, we have introduced the 4 main types of filter banks that are relevant to our work: the 1) filter-array filter bank, 2) polyphase filter bank, 3) octave filter bank and 4) Fast Filter Bank. The advantages and disadvantages associated with the various types of filter banks are summarized in Table 2-4.

Filter bank	Advantages	Disadvantages
Filter-array	- Can have highly customized specifications.	- Very high complexity.
Polyphase	- Very efficient when operated in multi-rate mode.	- The filter lengths may be very large, and increase with the number of channels. - High cost of decreasing transition width.
Octave	- Useful for non-uniform filter bank implementation.	- Very long filters required for uniform filter bank implementation. - Very high cost of decreasing transition width.
FFB	- Very efficient in single-rate mode. - Low coefficient count. - Low cost of decreasing transition width.	- Not well adapted for operating in multirate mode.

Table 2-4 Summary of the advantages and disadvantages associated with the various filter bank implementations.

The FFB has a very low coefficient count, and is very efficient in single-rate mode. In an example, the FFB was shown to have a significantly lower complexity than the other implementations, including the polyphase filter bank, when operated in single-rate mode.

However, the complexity of the FFB does not scale downward when its outputs are decimated. In an example for which the outputs of the filter banks were critically decimated, the complexity of the FFB significantly exceeded that of the polyphase filter bank.

Chapter 3

Single-rate Systems I: The Fast Filter Bank

Until recently, the theory on the Fast Filter Bank has remained as a study of efficient single-rate filter banks, with limited application to practical situations. The literature base relating to the FFB is very small, with the introduction of its concept in [21] and a short analysis on the selection of the subfilter transition widths in [22].

The FFB is normally characterized by having subbands with narrow transition widths, and consists of subfilters with very small filter lengths. It has been mentioned in the previous chapter that the FFB is very efficient in single-rate mode. However, when used as a multi-rate filter bank, its complexity becomes significantly higher than that of the polyphase filter bank. In this chapter, we focus on the design of the FFB for single-rate operation.

In this chapter, we assume the non-causal form for the representation of all filter transfer functions, i.e. they are represented as zero-delay filters.

3.1 Frequency response masking technique

Before delving into the topic of the FFB, let us first give an overview of the concept of Frequency Response Masking (FRM). The FRM technique was first introduced in [23]-[24], and it is used in the design of FIR filters with narrow transition widths by means of a multi-stage structure. The complexity of the multi-stage FRM filter is greatly reduced when compared to the conventional single-stage FIR filter.

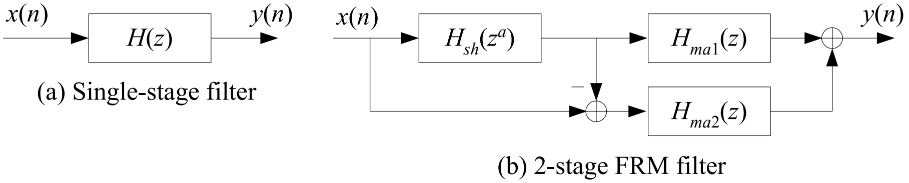


Figure 3-1 Block diagram showing the structure of a (a) single-stage filter; (b) 2-stage FRM filter.

Consider a single-stage FIR filter with transfer function $H(z)$ (Figure 3-1(a)). Suppose that the following set of specifications is desired: passband edge = ω_p, stopband edge = ω_s, peak passband ripple = δ_p and peak stopband ripple = δ_s.

The length of the optimal equiripple filter fulfilling this set of specifications can be estimated (see Appendix A). The transition width of the filter ($\Delta = \omega_s - \omega_p$) is an important determining factor in the resultant filter length N. Figure 3-2 shows the estimated minimum filter lengths for various transition widths and fixed values of $\delta_p = 0.1$ dB, $\delta_s = -60$ dB. It can be observed that for narrow transition widths, the required filter lengths increase drastically.

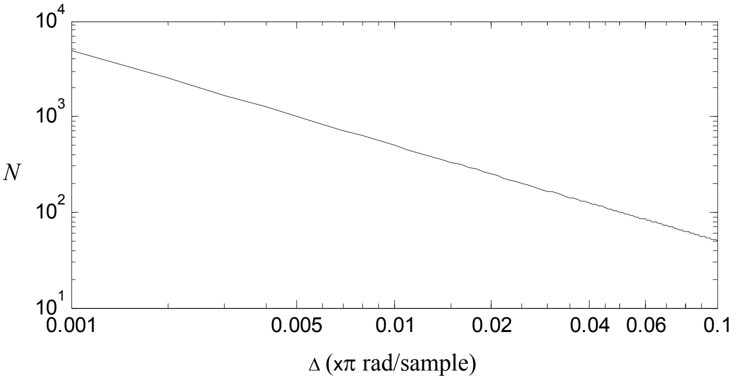


Figure 3-2 Plot of required filter length N for transition width Δ, when $\delta_p = 0.1$ dB and $\delta_s = -60$ dB.

In the Frequency Response Masking (FRM) approach to filter design, a multi-stage filter structure (such as shown in Figure 3-1(b)) is adopted. By appropriately designing the filters with transfer functions $H_{sh}(z^a)$, $H_{ma1}(z)$ and $H_{ma2}(z)$, the overall complexity of the filter system can be greatly reduced when compared to $H(z)$. The subscripts *sh* and *ma* are used to denote 'shaping' and 'masking', respectively.

To understand how the complexity of the filter system can be reduced by using this method, let us first consider the frequency response shown in Figure 3-3(a). Due to the narrow transition width of this response, a large filter length is expected for the single-stage structure of Figure 3-1(a).

In the FRM approach, a lowpass filter with transfer function $H_{sh}(z)$ is first designed. Its frequency response is shown in Figure 3-3(b). As can be observed from the plot, the transition width of this filter is much larger than that of $H(z)$, hence we expect $H_{sh}(z)$ to have a smaller filter length than $H(z)$. Interpolating $H_{sh}(z)$ by an integer factor a, the resultant frequency response of $H_{sh}(z^a)$ is given by the dotted line in Figure 3-3(c). The transfer function $H_{sh}(z^a)$ describes a sparse filter, because many of its coefficients are zero-valued. The number of non-zero coefficients in $H_{sh}(z^a)$ is equal to the filter length of $H_{sh}(z)$, and hence the effective complexity of $H_{sh}(z^a)$ is equal to the complexity of $H_{sh}(z)$.

The masking filter with transfer function $H_{ma1}(z)$ is designed with a frequency response shown by the bold line in Figure 3-3(c) and its purpose is to mask the unwanted passbands of $H_{sh}(z^a)$. The resultant frequency response of the cascade $H_{sh}(z^a)H_{ma1}(z)$ is shown in Figure 3-3(d). Again, the transition width of $H_{ma1}(z)$ is larger than that of $H(z)$, so we expect $H_{ma1}(z)$ to be a shorter filter than $H(z)$.

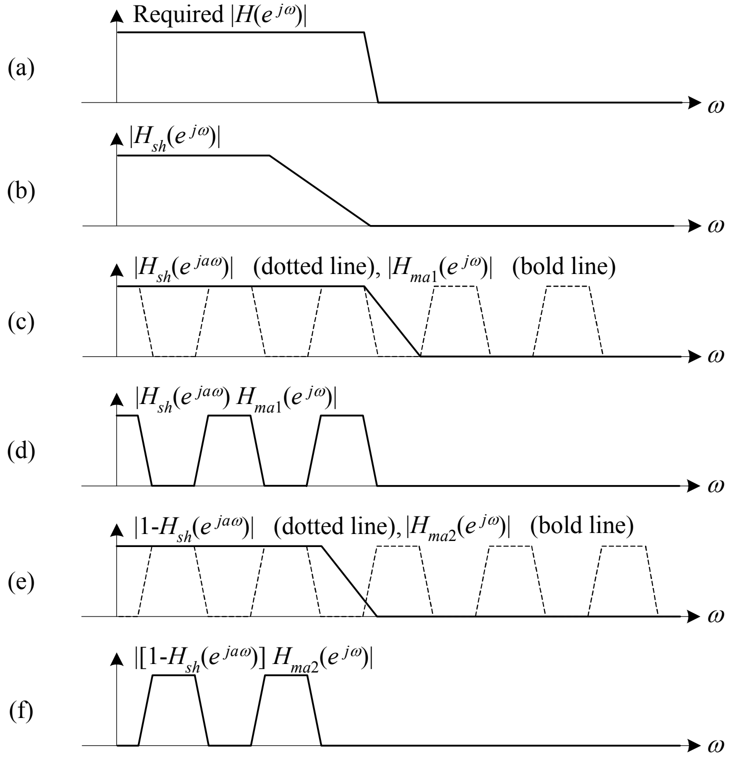


Figure 3-3 Frequency response plots illustrating the FRM technique.

The dotted line in Figure 3-3(e) shows the frequency response of $\left(1-H_{sh}(z^a)\right)$ and the bold line shows the frequency response of $H_{ma2}(z)$. The frequency response of the cascaded filters $\left(1-H_{sh}(z^a)\right)H_{ma2}(z)$ is shown in Figure 3-3(f). The two outputs of the FRM filter system are then summed together to give the resultant transfer function:

$$H_{FRM}(z)=\left(1-H_{sh}(z^a)\right)H_{ma2}(z)+H_{sh}(z^a)H_{ma1}(z)\,. \tag{3.1}$$

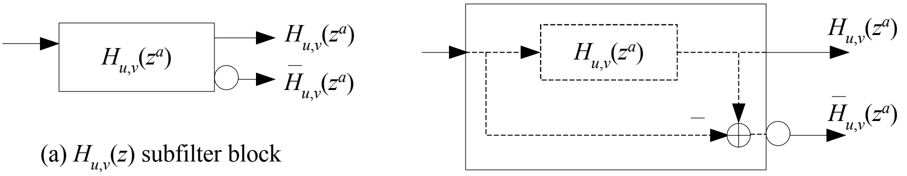
The resultant FRM filter system has a frequency response approximating the desired narrow transition width, lowpass FIR filter $H(z)$. However, it comprises component FIR filters that have lower complexity than $H(z)$, and the overall complexity is reduced.

In a FRM filter design example in [23], a linear-phase FIR lowpass filter with specifications $\omega_p = 0.6\pi$, $\omega_s = 0.61\pi$, $\delta_p = 0.1$ dB and $\delta_s = -40$ dB was designed. An estimated filter length of 383 was required for the $H(z)$ design. Using the FRM technique, the lengths of the filters $H_{sh}(z)$, $H_{ma1}(z)$ and $H_{ma2}(z)$ were reported to be 45, 38 and 30 respectively. The interpolation factor a was chosen to be 9. In this particular design example, the complexity of the FRM system was only 29.5% of the non-FRM system. A drawback of the FRM technique however, is a slightly increased group delay. In this example the group delay of the FRM system was 216.5, compared to 191 in the non-FRM system, an increase of 13.4%.

3.2 FFB structure and operation

In this section, we describe the operation of the FFB, and formalize the notations used. Note that we generally assume that the inputs and outputs of the FFB are complex-valued. Real-valued inputs and outputs will be considered in Section 4.1.

3.2.1 Representation of a subfilter block



(a) $H_{u,v}(z)$ subfilter block

(b) $\bar{H}_{u,v}(z^a) = 1 - H_{u,v}(z^a)$

Figure 3-4 Representation of a subfilter block $H_{u,v}(z^a)$.

In this section, we extend the FRM theory to the design of the FFB. The non-causal forms for all filters and subfilters are assumed in this discussion. We begin by first defining a subfilter block $H_{u,v}(z^a)$ as shown in Figure 3-4(a). The subfilter block $H_{u,v}(z^a)$ consists of 2 filters with a transfer function $H_{u,v}(z^a)$ defined by the upper output path, and its complementary transfer function $\bar{H}_{u,v}(z^a)$ defined by the lower output path which is demarcated by a circle. For example, if $H_{u,v}(z^a)$ is a lowpass function, then $\bar{H}_{u,v}(z^a)$ is a highpass function, and the sum of the 2 functions is equal to unity:

$$\bar{H}_{u,v}(z^a) + H_{u,v}(z^a) = 1. \tag{3.2}$$

Figure 3-4(b) is an implementation of the subfilter block $H_{u,v}(z^a)$, where $\bar{H}_{u,v}(z^a) = 1 - H_{u,v}(z^a)$. Another possible definition of the complementary transfer function is $\bar{H}_{u,v}(z^a) = H_{u,v}(-z^a)$.

3.2.2 *Description of FFB structure*

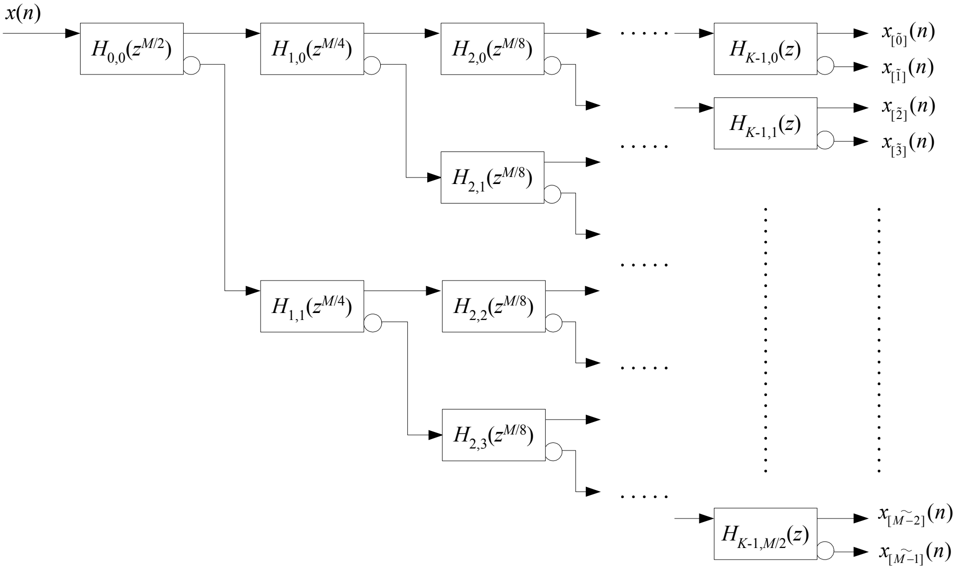


Figure 3-5 The tree-structured Fast Filter Bank.

Figure 2-10 (which we repeat in Figure 3-5 for convenience) shows the block diagram of the M-channel Fast Filter Bank (FFB), which consists of a multi-stage, tree-structured arrangement of subfilters. The number of cascade stages is K, and the number of subbands is $M = 2^K$. The subbands of the FFB shown in Figure 3-5 are even-stacked. For an input $x(n)$, the M subband outputs of the FFB are represented using $x_{[m]}(n)$, for $0 \le m \le M-1$. It is convenient to use m to index the subbands in order of increasing frequency, such that the center frequency of the subband m is at $2\pi m / M$. In so doing, the outputs of the filter bank (Figure 3-5) become ordered in bit-reversed fashion, where $\tilde{m}$ is the bit-reversed version of m in K bits:

$$\tilde{m} = \sum_{k=0}^{K-1} m_2(k) 2^{K-1-k} , \tag{3.3}$$

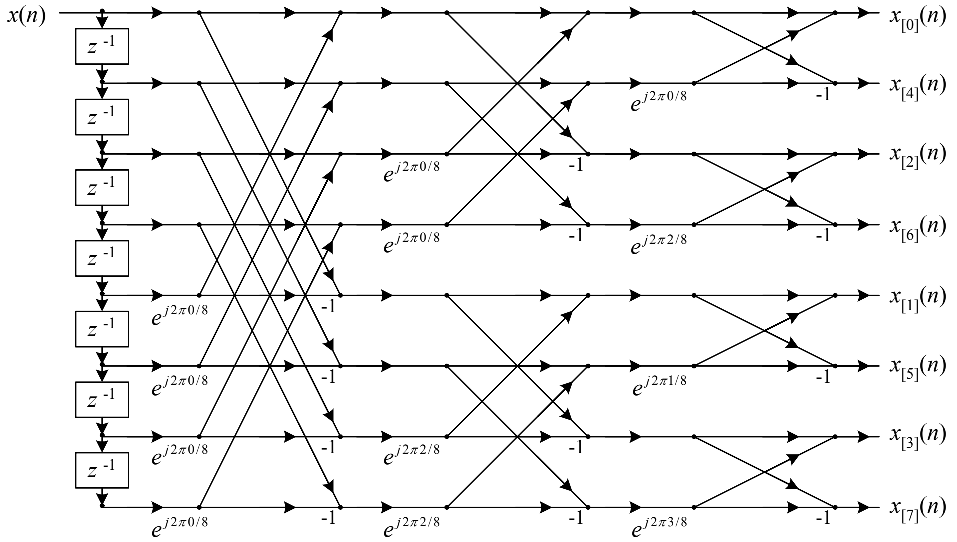
where $m_2(k)$ is the k-th least significant bit of m, such that:

$$m = \sum_{k=0}^{K-1} m_2(k) 2^{k} . \tag{3.4}$$

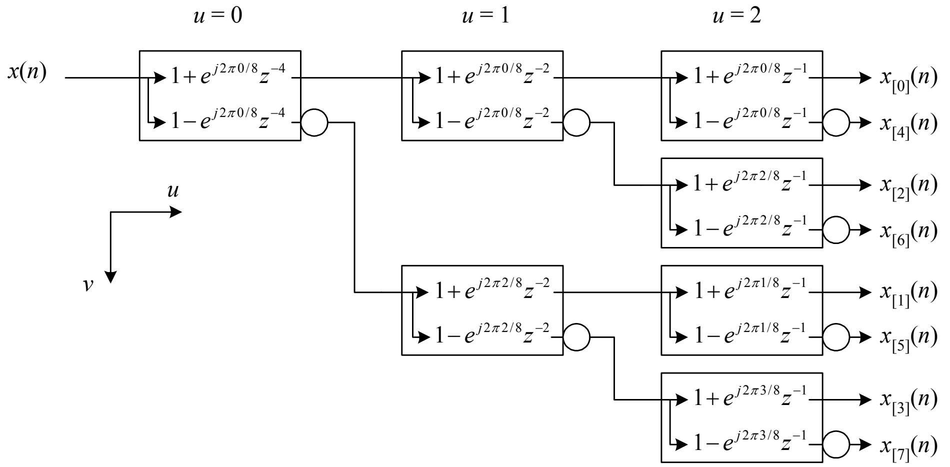
The subscript '2' indicates a base-2 representation, and $m_2(0)$ is the value of the least significant bit of m.

The reason for the bit-reversed ordering of the subband index at the FFB outputs lies in the structure of the FFB, and is left to be explained in Section 3.2.3. The FFB transfer functions $H_{[m]}(z)$ generally have complex-valued coefficients. We consider the FFB for the complex-valued input signal, complex-valued output signal scenario in this chapter.

3.2.3 FFB and sliding FFT



(a) Represented using butterfly structure.



(b) Represented using subfilter block diagram.

Figure 3-6 Eight-channel sliding FFT with a decimation-in-time FFT structure.

The sliding FFT ([31]-[32]) originally inspired the FFB, and hence the structure of the FFB follows closely that of the sliding FFT. The sliding FFT takes the structure of the decimation-in-time FFT, as shown in Figure 3-6 for the 8-channel case. In Figure 3-6(a), the sliding FFT is shown using the well-known FFT butterfly structure. In Figure 3-6(b), we represent the sliding FFT using the equivalent subfilter block diagram, which is used throughout this thesis for representing the FFB.

For the sliding FFT, the input $x(n)$ is in natural order, and the outputs are in bit-reversed order. The transfer functions of the subfilters at each stage have only 2 non-zero coefficients, given by:

$$H_{u,v}(z) = 1 + e^{j2\pi\tilde{v}/M} z^{-2^{K-1-u}}, \tag{3.5}$$

and their complementary transfer functions:

$$\bar{H}_{u,v}(z) = 1 - e^{j2\pi\tilde{v}/M} z^{-2^{K-1-u}}, \tag{3.6}$$

where K is the number of stages, M is the number of subbands, and $\tilde{v}$ is the bit-reversed version of v in K-1 bits. Bit-reversal is further explained below.

The sliding FFT has fixed transition widths and stopband attenuation. The FFB is a generalized case of the sliding FFT (see Section 3.4), in the sense that the filter lengths of its subfilters are arbitrary, and the frequency responses of its outputs are arbitrary.

3.2.4 *Bit-reversal*

The integer $\tilde{v}$ is the bit-reversed version of the integer v expressed in u bits. If v is expressed in binary form:

$$v = \sum_{u'=0}^{u-1} v_2(u') 2^{u'}, \tag{3.7}$$

where $v_2(u')$ is the binary value of the u'-th bit of v, then $\tilde{v}$ is defined as:

$$\tilde{v} = \sum_{u'=0}^{u-1} v_2(u') 2^{u-1-u'}. \tag{3.8}$$

For example, an integer v = 10 can be expressed in binary form as '...001010'. The bit-reversed version of v in 4 bits is given by '0101', and $\tilde{v} = 5$. The bit-reversed version of v in 5 bits is given by '01010' and $\tilde{v} = 10$.

3.2.5 *Description of FFB operation*

To understand how the M-channel FFB shown in Figure 3-5 operates, let us first consider the subfilter with transfer function $H_{0,0}(z)$. Its frequency response is shown in Figure 3-7(a). Interpolating the filter by a factor of M/2, the resultant frequency response is shown by the dotted line in Figure 3-7(b). The interpolated transfer function $H_{0,0}(z^{M/2})$ consists of a set of *M*/2 passbands and the complementary transfer function $\bar{H}_{0,0}(z^{M/2})$ consists of a complementary set of *M*/2 passbands. Together, they constitute the *M* subbands of the FFB.

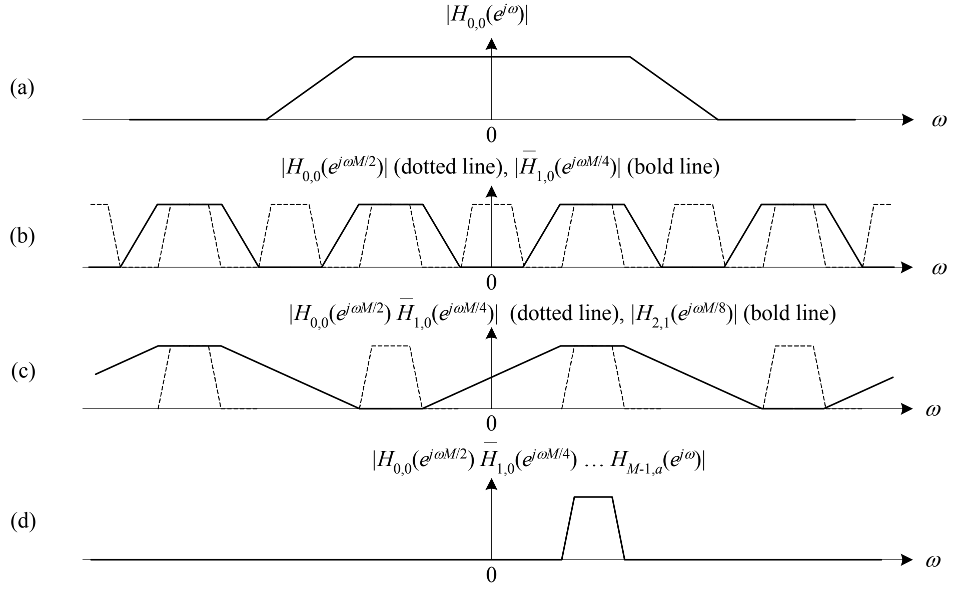


Figure 3-7 Frequency responses of the subfilters in the FFB.

Applying a masking subfilter with transfer function $\bar{H}_{1,0}(z^{M/4})$ and frequency response given by the bold line in Figure 3-7(b), the cascaded transfer function $H_{0,0}(z^{M/2})\bar{H}_{1,0}(z^{M/4})$ retains only the passbands shown by the dotted line in Figure 3-7(c). By masking half of the number of passbands at each cascade stage in the FFB, only one passband per output channel is retained after *K* stages. Figure 3-7(d) shows the cumulative frequency response of the *K* cascaded transfer functions $H_{0,0}(z^{M/2})\bar{H}_{1,0}(z^{M/4})\ldots H_{K-1,a}(z)$.

Similarly, other cascade paths in the filter bank lead to different masking combinations and the outputs of the FFB take on the *M*-channel filter bank characteristic. Since the transition edges of the FFB are primarily determined by the first subfilter with transfer function $H_{0,0}(z)$, this subfilter is termed the shaping subfilter. The remaining subfilters perform masking functions to remove unwanted passbands of the interpolated shaping subfilter, and are therefore termed masking subfilters.

3.3 FFB design considerations

In the previous section, we have discussed the operating principle of the FFB. In this section, we provide design considerations for the FFB and its subfilters. Consider the design of a filter bank with M subbands. We are generally restricted to $M = 2^K$, where K is the number of cascade stages. Assume that the desired passband and stopband ripple are δ_{pp} and δ_{ss} respectively, and the desired transition width of each subband is Δ. Let us denote the passband and stopband edges of the subfilter with transfer function $H_{u,v}(z)$ as $\omega_{p(u,v)}$ and $\omega_{s(u,v)}$, its transition width as $\Delta_{u,v}$, and its passband and stopband ripples as $\delta_{p(u,v)}$ and $\delta_{s(u,v)}$.

3.3.1 Subfilter design

We are required to design the shaping subfilter with transfer function $H_{0,0}(z)$ and the corresponding set of masking subfilters with transfer functions $H_{u,v}(z)$, such that the transfer functions $H_{[m]}(z)$ (from the input to the m-th output of the FFB), for $0 \le m \le M-1$, are bandpass and have center frequencies $2\pi m/M$. In the following discussion, we establish a set of basic conditions governing the design of the FFB subfilters. *Conditions 1* to 3 are necessary conditions that are required for the outputs of the FFB to attain the desired bandpass characteristic with center frequency at $2\pi m/M$. *Conditions 4* and *5* are optional conditions that determine properties of the output frequency responses which are possibly desired but not necessarily required.

Condition 1: The shaping subfilter has a transfer function $H_{0,0}(z)$. The frequency response of its interpolated form $H_{0,0}(z^{M/2})$ must be approximately equal to the frequency response of the desired subband $H_{[m]}(z)$ within the passband and transition band. Since the subbands are uniform, this can be equivalently represented as $H_{0,0}(e^{j\omega M/2}) \approx H_{[0]}(e^{j\omega})$, for $\left(0 \le \omega \le \frac{\Delta}{2} + \frac{\pi}{M}\right)$.

Since $H_{0,0}(z)$ is interpolated by a factor of $M/2$, the transition width requirement of the interpolated $H_{0,0}(z)$ is relaxed by a factor of $M/2$:

$$\Delta_{0,0} = \omega_{s(0,0)} - \omega_{p(0,0)} = \frac{M}{2}\Delta \,. \tag{3.9}$$

Its passband and stopband edges are:

$$\omega_{p(0,0)} = \frac{M}{2}\omega_{pp} \text{, and}$$

$$\omega_{s(0,0)} = \frac{M}{2}\omega_{ss} \,. \tag{3.10}$$

Furthermore, the passband edge of $H_{u,0}(z^{2^{K-u-1}})$ must overlap the passband and transition band of $H_{[0]}(z)$:

$$\omega_{p(u,0)} \geq 2^{-u}\omega_{s(0,0)}, \text{ for } 1 \leq u \leq K-1. \tag{3.11}$$

<u>Condition 2</u>: The masking subfilters have transfer functions given by $H_{u,0}(z^{2^{K-u-1}})$, for $1 \leq u \leq K-1$. In order that the frequency response at the *m*-th output of the FFB has only one passband, all unwanted passbands and transition bands of $\prod_{u'=0}^{u-1} H_{u',0}(z^{2^{K-u'-1}})$ must be in the stopband of $\prod_{u'=u}^{K-1} H_{u',0}(z^{2^{K-u'-1}})$. *Condition 2* can be satisfied by defining the stopband edges of $H_{u,0}(z)$ as follows:

$$\omega_{s(u,0)} \leq \pi - 2^{-u}\omega_{s(0,0)}, \text{ for } 1 \leq u \leq K-1. \tag{3.12}$$

We note that the transition widths of the subfilters become increasingly larger as *u* increases, and are given by:

$$\Delta_{u,0} = \pi - 2^{-u+1}\omega_{s(0,0)}, \text{ for } 1 \leq u \leq K-1. \tag{3.13}$$

We can thus infer that the filter length of $H_{u,v}(z)$ generally becomes smaller as *u* increases.

<u>Condition 3</u>: In order that the subband frequency responses $H_{[m]}(z)$ for $0 \leq m \leq M-1$ have mutually exclusive passbands, the transfer functions $H_{u,v}(z^{2^{K-u-1}})$ (for a given *u* and $0 \leq v \leq 2^u - 1$) must be designed such that their passbands cover mutually exclusive regions of the frequency spectrum, in addition to satisfying *Conditions 1 and 2*. For our work, we can choose (as is used for the case of the sliding FFT):

$$H_{u,v}(z) = H_{u,0}(e^{-j\frac{2\pi\tilde{v}}{M}}z), \text{ for } 0 \leq v \leq 2^u - 1, \tag{3.14}$$

where $\tilde{v}$ is the bit-reversed version of *v* in *K*-1 bits.

<u>Optional condition 4</u>: Let the transfer functions of the subbands of the filter bank sum to unity, i.e. $\sum_{m=0}^{M-1} H_{[m]}(z) = 1$. In this case, the sum of the outputs of the FFB $\hat{x}(n)$ is equal to the input $x(n)$. For a single-rate system, this obviates the necessity of a synthesis filter bank. The condition can be satisfied if we define:

$$\bar{H}_{u,v}(z) = 1 - H_{u,v}(z), \text{ for all } u \text{ and } v. \tag{3.15}$$

<u>Optional condition 5</u>: Let the outputs of the FFB have exactly uniform subbands, such that $H_{[m]}(z) = H_{[0]}(e^{-j\frac{2\pi m}{M}}z)$. This is achieved when *Condition 3* is satisfied and moreover:

$$\bar{H}_{u,v}(z) = H_{u,v}(-z), \text{ for all } u \text{ and } v. \tag{3.16}$$

In order to satisfy both *Conditions 4* and *5* for linear phase FIR filters, we require (i) the coefficients of the subfilters to be zero for all even values of n, and (ii) the coefficient $h_{u,v}(0)$ to be 0.5. For both requirements to be satisfied, a half-band filter is necessary for $H_{u,0}(z)$. If a non-half-band filter is used for $H_{u,0}(z)$, then either the sum of the impulse responses of the FFB subbands is not unity (*Condition 4* is not satisfied), or the frequency responses of the subbands are not exactly uniform (*Condition 5* is not satisfied).

Note: A zero-phase half-band filter ([25]-[26]) is defined by the transfer function $H(z) = 0.5 + z^{-1}H'(z^2)$, where the transfer function $H'(z)$ contains the odd-numbered coefficients of the half-band filter. The half-band filter has the property that $H(z) + H(-z) = 1$.

An additional advantage of a half-band filter is its decreased complexity when compared to a non-half-band filter with similar specifications. We note that the coefficients of the half-band filter $h(n)$ are equal to zero for even non-zero n. However, the passband and stopband ripples of a half-band filter are constrained to be equal, and this might result in a slightly longer filter than is necessary, leading to an increased delay.

To illustrate the above points, let us consider an example. Suppose that a filter with specifications of $\delta_p = 0.01 = 0.086$ dB, $\delta_s = 0.001 = -60$ dB and $\Delta = 0.2\pi$ is desired. A linear-phase equiripple filter requires a filter length of 27. A half-band equiripple filter, on the other hand, requires a filter length of 31, because δ_p must be equal to δ_s. In this example, the half-band filter is longer and hence a greater delay is incurred. However, the half-band filter has only 17 non-zero coefficients when compared to the non-half-band filter which has 27 non-zero coefficients.

3.3.2 *Filter bank delay*

In most designs, the subfilter with transfer function $H_{0,0}(z)$ has the smallest transition width of all the subfilters. Hence, $H_{0,0}(z)$ is usually the longest subfilter. Furthermore, it is interpolated by $M/2$, which is the largest interpolation factor for all the subfilters. For a linear-phase FIR design, the delay incurred in stage u by $H_{u,0}(z^{M/2^{u+1}})$ is:

$$d_u = M(N_{u,0} - 1)/2^{u+2}, \tag{3.17}$$

where d_u is the delay of $H_{u,0}(z^{M/2^{u+1}})$.

In most FFB designs, the filter lengths of the subfilters decrease as u increases, i.e. $N_{u+1,0} \le N_{u,0}$. Therefore, we can assume that $d_{u+1} \le d_u / 2$. Then, the delay d_0 incurred by the shaping subfilter is at least half of the overall filter bank delay, i.e. $d_0 > 2\sum_{u=0}^{K-1} d_u$.

Furthermore, the contribution to the overall complexity of the filter bank due to $H_{0,0}(z)$ is usually small.

Thus, a significant reduction in the overall filter bank delay without a significant change to the overall complexity can be achieved, if we design $H_{0,0}(z)$ to have a small delay. Methods of designing suitable low-delay FIR filters can be found in [27]-[30]. A minimum-phase FIR filter [9] can be designed, by taking an FIR filter and replacing those zeros which are outside the unit circle by their inverses. The resultant filter has a non-linear phase response.

A more flexible approach, where the phase response is approximately linear in the passband can be adapted from [30]. We approach the design of $H_{0,0}(z)$ by optimizing its coefficients to have minimum energy in the stopband. The problem is phrased as an eigenproblem, with the solution being the eigenvector corresponding to its minimum eigenvalue. The desired passband delay $d_{0,0}$ can be selected to take on an odd value ($2k$+1, where k is an integer), by constraining the coefficients $h_{0,0}(n = 2k'+1)$ to be non-zero at $k' = k$, and zero at all other k'. The reader is referred to [30] for a more detailed description.

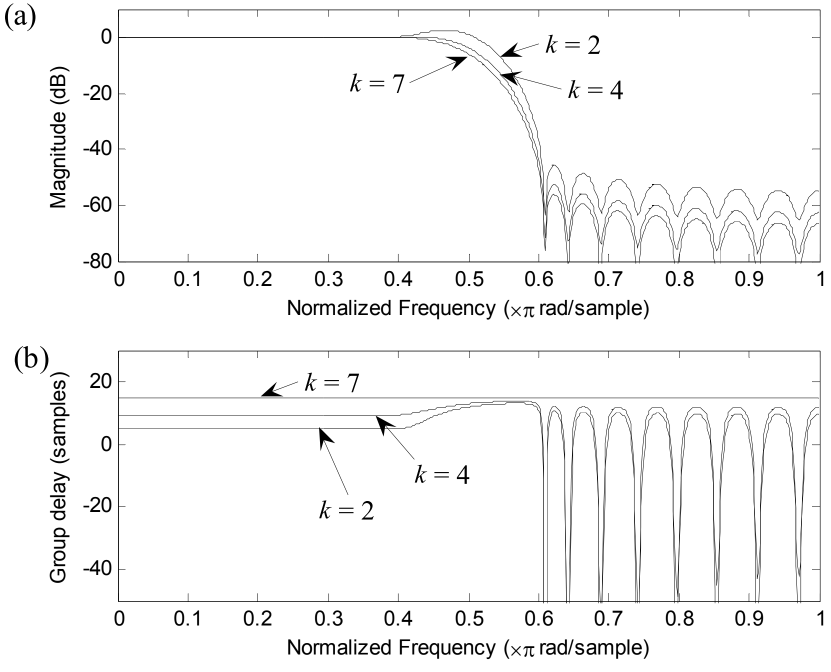


Figure 3-8 (a) Magnitude response of filters with different passband delays, (b) Group delay of filters with different passband delays.

For $H_{0,0}(z)$ with a filter length of 31, we used this method to design filters with different values of k. For k=7, $H_{0,0}(z)$ is linear-phase. Figure 3-8(a) shows a comparison of their magnitude responses, and Figure 3-8(b) shows a comparison of their group delays. Low delay FIR filters designed using this method have a non-linear phase in the transition band and stopband. Its magnitude response in the transition band and stopband is also shifted upwards. In the passband, however, the magnitude and phase responses remain almost unchanged. The phase in the passband is also approximately linear.

3.3.3 Selection of transition width

We have denoted the length of $H_{u,0}(z)$ as $N_{u,0}$. We have also made the general assumption that the complexity of the Fast Filter Bank is approximated by:

$$\Gamma_{FFB} \approx N_{0,0} + 4\sum_{u=1}^{K-1} 2^u N_{u,0} \ . \tag{3.18}$$

Note that if we use linear-phase half-band subfilters, the complexity is reduced to less than half of the above value (refer to Appendix B). Furthermore, we noted that since the transition widths of the subfilters $\Delta_{u,0}$ generally increases as u increases, the lengths of the subfilters $N_{u,0}$ decreases as u increases. From (3.13), we know that $\Delta_{u,0}$ can be calculated from $\omega_{s(0,0)}$. If the width of the passband is approximately equal to the width of the stopband for $H_{0,0}(z)$, i.e. $\omega_{p(0,0)} \approx \pi - \omega_{s(0,0)}$, then $\Delta_{0,0} \approx 2\omega_{s(0,0)} - \pi$.

Consider that there are 2^u subfilters in stage u. The contribution to the overall complexity of the filter bank from stage (u-1) and u is given by $4.2^{u-1}\left(N_{u-1,0} + 2N_{u,0}\right)$. For the case when $\Delta_{u-1,0} = \Delta_{u,0} = \pi/3$, then $N_{u-1,0} = N_{u,0}$ and their contribution to the overall complexity is $4.2^{u-1}\left(3N_{u,0}\right)$. For the case when $\Delta'_{u-1,0} > \pi/3$, then $\Delta'_{u,0} < \pi/3$. Therefore, $N'_{u,0} > N_{u,0}$ and $N'_{u-1,0} < N_{u,0}$, and their contribution to the overall complexity is increased to $4.2^{u-1}\left(N'_{u-1,0} + 2N'_{u,0}\right)$.

From this reasoning, if $\Delta_{0,0} > \pi/3$ then $\Delta_{1,0} < \pi/3$. The contribution to the overall complexity in this case is greater than if $\Delta_{0,0} = \Delta_{1,0} = \pi/3$. Since the overall transition width of the filter bank is determined by the transition width of the shaping subfilter, it is desirable to keep $\Delta_{0,0}$ small. Increasing $\Delta_{0,0}$ beyond $\pi/3$ not only increases the overall complexity of the filter bank, but also provides no advantage with regards to the filter bank transition width Δ.

Table 3-1 shows values of $\Delta_{u,0}$, $N_{u,0}$ and overall complexity for various $\Delta_{0,0}$, for a FFB with K=4. The stopband attenuation is 50 dB in this example. We can see that the optimal complexity occurs for $\Delta_{0,0}$ falling in the region of 0.28π to 0.3π. Decreasing the transition width below this amount causes $N_{0,0}$ to increase, and increasing the transition width causes $N_{1,0}$ to be increased.

$\Delta_{0,0}$	$\Delta_{1,0}$	$\Delta_{2,0}$	$\Delta_{3,0}$	$N_{0,0}$	$N_{1,0}$	$N_{2,0}$	$N_{3,0}$	Γ_{FFB}
0.10π	0.450π	0.725π	0.8625π	27	7	5	3	259
0.20π	0.400π	0.700π	0.8500π	13	7	5	3	245
0.24π	0.380π	0.690π	0.8450π	11	7	5	3	243
0.28π	0.360π	0.680π	0.8400π	9	7	5	3	241
0.30π	0.350π	0.675π	0.8375π	9	7	5	3	241
0.32π	0.340π	0.670π	0.8350π	9	9	5	3	257
0.35π	0.325π	0.663π	0.8313π	7	9	5	3	255
0.50π	0.250π	0.625π	0.8125π	7	11	5	3	271
0.70π	0.150π	0.575π	0.7875π	5	19	5	5	397
0.80π	0.100π	0.550π	0.7750π	5	27	7	5	493

Table 3-1 Effect of $\Delta_{0,0}$ on the filter bank complexity, for $K = 4$ and $\delta_s = -50$ dB.

We find that in most cases, the general rule applies: the minimum complexity design occurs when $0.28 < \Delta_{0,0} / \pi < 0.3$. Thus, for design specifications where $\Delta > 0.6\pi / M$, we recommend setting $\Delta_{0,0} = 0.3\pi$ which leads to $\Delta' = 0.6\pi / M$. The result of using Δ' instead of Δ is not only a smaller transition width, but also a reduced overall filter bank complexity. For design specifications with a very small transition width, such as $\Delta < 0.1\pi / M$, we recommend using the FRM technique as described in Section 3.1 to further decompose the subfilter with transfer function $H_{0,0}(z)$.

3.3.4 Distribution of complexity by stage

From (3.18), we note that the complexity contributed by the subfilters in stage u is equal to the complexity of $H_{u,0}(z)$, multiplied by the number of subfilters 2^u. Therefore, the complexity distribution of the FFB is heavily weighted towards the latter stages. Table 3-2 lists the complexity distribution of the FFB for M = 128, $\Delta_{0,0} = 0.2\pi$ and stopband attenuation of 50 dB.

M	Γ_{FFB}	Total complexity of all subfilters in the stage u						
		0	1	2	3	4	5	6
128	1663	23	88	112	96	192	384	768

Table 3-2 Complexity distribution of the FFB,
for M = 128, $\Delta_{0,0} = 0.2\pi$ and δ_s = -50 dB.

We can draw several conclusions here:

(i) Significant savings can be obtained if we focus on reducing the complexity of the subfilters in the latter stages. This is addressed in Section 5.1.2, where we consider a method of decimation for multi-rate filter banks that reduces subfilter complexity in the latter stages; and in Section 6.4, where we consider filter banks with signed-powers-of-two coefficients.

(ii) The shaping subfilter with transfer function $H_{0,0}(z)$ has a small contribution to the overall complexity. Therefore, a great flexibility is allowed in the design of $H_{0,0}(z)$ without a significant impact on the overall complexity.

(iii) From (ii), we can conclude that the overall complexity is also insensitive to changes in the transition width of the filter bank. Thus, we can design efficient filter banks with very small transition widths. This was clearly demonstrated in the examples in Section 2.7, where we showed that at very small transition widths, the FFB is more efficient than the polyphase filter bank when in single-rate operation.

3.4 Properties of the FFB

In this section, we formalize the properties and definitions of the Fast Filter Bank. Figure 3-9 shows a detailed block diagram of the FFB.

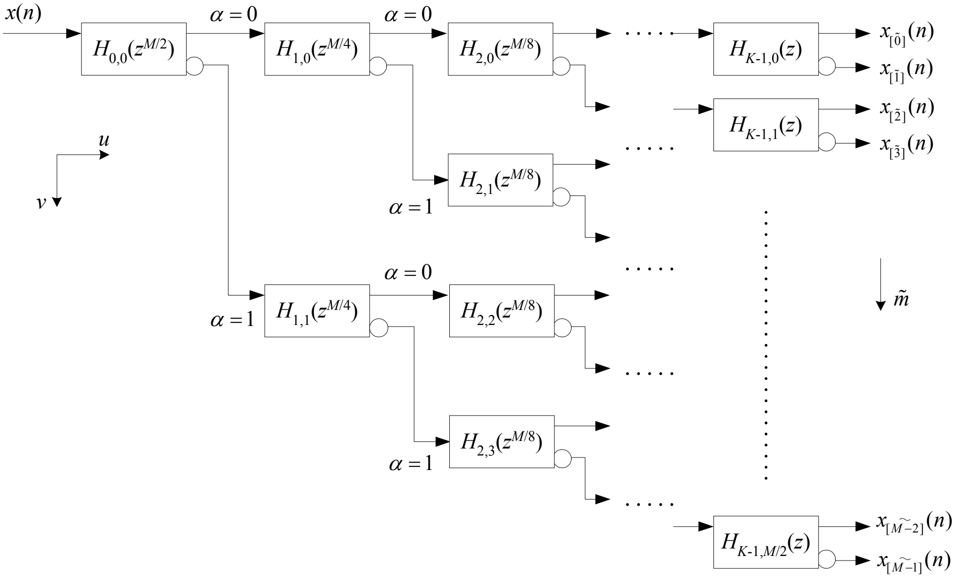


Figure 3-9 Detailed FFB block diagram.

Property 1: The FFB has $M = 2^K$ uniform subbands, where K is the number of cascade stages. The passband of the subband m has a center frequency $2\pi m / M$. The transfer function of each subband $H_{[m]}(z)$ is generally complex, i.e. its frequency response is not symmetrical about zero.

Property 2: The subfilters are defined as $H_{u,v}(z)$ for $0 \le u \le K-1$ and $0 \le v \le 2^u - 1$. At stage u, there are 2^u subfilters. We define the prototype lowpass subfilter for stage u as the subfilter $H_{u,0}(z)$.

Property 3: The subfilters $H_{u,v}(z)$ can be derived from the prototype lowpass subfilter $H_{u,0}(z)$ by complex-modulating its coefficients according to:

$$H_{u,v}(z) = H_{u,0}(e^{-j\frac{2\pi\tilde{v}}{M}} z), \tag{3.19}$$

where $\tilde{v}$ is the bit-reversed version of v in K-1 bits. As a result, the coefficients of $H_{u,v}(z)$ are generally complex-valued.

Property 4: The subfilters at stage u are interpolated by a factor of 2^{K-u-1}.

Property 5: The outputs of the FFB are arranged from top-to-bottom order according to $\tilde{m}$, where $\tilde{m}$ is the bit-reversed version of m in K bits.

Property 6: The transfer function of subband m is given in a general form by:

$$\begin{aligned} H_{[m]}(z) &= \prod_{u=0}^{K-1}\Big[\big(1-\alpha(m,u)\big)\big(H_{u,v(m,u)}(z^{\ell(K,u)})\big) \\ &\quad +\alpha(m,u)\big(\bar{H}_{u,v(m,u)}(z^{\ell(K,u)})\big)\Big] \\ &= \prod_{u=0}^{K-1}\Big[\big(1-\alpha(m,u)\big)\big(H_{u,0}(W(m,u)z^{\ell(K,u)})\big) \\ &\quad +\alpha(m,u)\big(\bar{H}_{u,0}(W(m,u)z^{\ell(K,u)})\big)\Big], \end{aligned} \tag{3.20}$$

where:

- $v(m,u)$ is a function representing the value of v in the cascade path for the subband m and stage u. The variable $v(m,u)$ can be found by taking the bit-reversed version of $\tilde{v}(m,u)$ in u bits, where:

$$\tilde{v}(m,u) = m - 2^u \left\lfloor m/2^u \right\rfloor. \tag{3.21}$$

  The symbol $\lfloor . \rfloor$ represents the operator for rounding down to the nearest integer. The above equation can be simply interpreted as taking the remainder when m is divided by 2^u.

- $\alpha(m,u)$ is a binary value, and is equal to 0 when the cascade path of the subband m and stage u is the direct output of the subfilter $H_{u,v}(z)$. It is equal to 1 when the cascade path of the channel m and stage u is the complementary output of the subfilter $\bar{H}_{u,v}(z)$. The variable $\alpha(m,u)$ satisfies the condition:

$$m = \sum_{u=0}^{K-1} 2^u \alpha(m,u). \tag{3.22}$$

  The variable $\alpha(m,u)$ can be found by taking the value of the u-th bit of m, where bit 0 is the least significant bit.

- $W(m,u)$ is an exponential function used to modulate the coefficients to shift the passband to the correct center frequency, and:

$$W(m,u) = e^{-j\frac{2\pi\tilde{v}}{M}} . \tag{3.23}$$

- $\ell(K,u)$ is the interpolation factor of the subfilter in stage u and is given by:

$$\ell(K,u) = 2^{K-u-1} . \tag{3.24}$$

Since K is constant for an instance of the filter bank, we can also express this as $\ell(u)$.

3.5 Design example: 32-channel FFB

3.5.1 Design example

In this section, we design a 32-channel FFB with $\Delta = \pi / 80$, $\delta_{pp} = \delta_{ss} = 0.001$ (0.0087 dB and -60 dB respectively). Half-band subfilters are used, and the coefficients of the prototype filters $H_{u,0}(z)$ are tabulated in Table 3-3. We assume the non-causal form, where $h(n)$ for $n = 0$, is the center coefficient.

n	$h_{0,0}(n)$	$h_{1,0}(n)$	$h_{2,0}(n)$	$h_{3,0}(n)$	$h_{4,0}(n)$
0	0.5000	0.5000	0.5000	0.5000	0.5000
1	0.3157	0.3079	0.2864	0.2825	0.2506
2	0.0000	0.0000	0.0000	0.0000	
3	-0.0984	-0.0782	-0.0370	-0.0326	
4	0.0000	0.0000			
5	0.0515	0.0259			
6	0.0000	0.0000			
7	-0.0298	-0.0063			
8	0.0000				
9	0.0172				
10	0.0000				
11	-0.0094				
12	0.0000				
13	0.0046				
14	0.0000				
15	-0.0021				

Table 3-3 Coefficient values for $H_{u,0}(z)$ for a 32-channel FFB.

The frequency responses for adjacent subbands m=0 and m=1 are shown in Figure 3-10. In the same figure, we also show the frequency response of a sliding FFT filter bank, for m=2. Recall that the sliding FFT filter bank is a specific case of the FFB, when the filter length of each subfilter is equal to 2. As a result, we expect the FFB to have a smaller transition width and a larger stopband attenuation than the sliding FFT filter bank. This can be clearly observed in Figure 3-10 for the given example.

We also expect the FFB to have a larger complexity and delay than the sliding FFT filter bank. For the FFB, the complexity and delay are calculated to be 200 and 314 respectively. For the sliding FFT filter bank, the complexity and delay are calculated to be 121 and 15.5

respectively. Note that in this example, the FFB was designed to a much more stringent specification than the sliding FFT filter bank, hence the greatly increased delay.

A polyphase filter bank with the same specifications as the designed FFB is calculated to have a complexity of 800, and a delay of 240. In single-rate mode, the 32-channel FFB has about 1/4 of the complexity, and 1.3 times the delay of the polyphase filter bank.

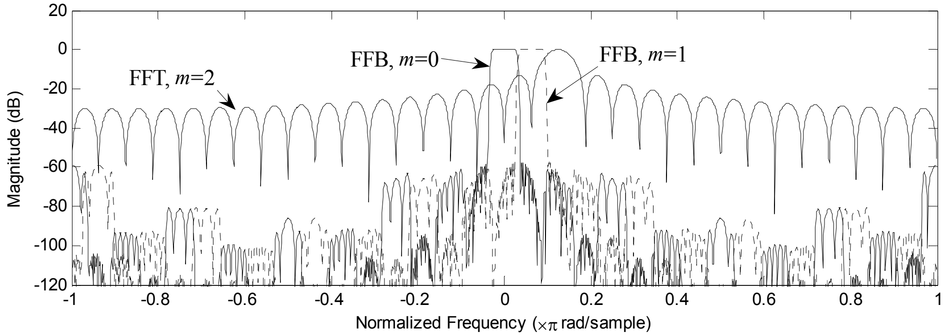


Figure 3-10 Frequency responses for a 32-channel FFB, and a 32-channel sliding FFT filter bank.

3.5.2 Discussions

We considered the design of a 32-channel FFB. In single-rate operation, the FFB is very efficient. In multi-rate operation however, the FFB becomes inefficient compared to the polyphase filter bank. This aspect will be further discussed in Section 5.1.

In modern audio compression methods (and also in other areas of application such as communications), the filter banks that are used typically have a much larger number of channels, e.g. 576 channels for MP3 and 1024 channels for AAC. The design of the FFB for a large number of channels is not difficult. The filter lengths of the subfilters remain small, and it is the interpolation factor of the transfer functions of these subfilters that are increased. The design methodology is repetitive and can be easily performed recursively. A basic design methodology is as follows:

1. Set stopband attenuation. Set stage $u = 0$.
2. Design shaping subfilter. Calculate transition width for next stage.
3. $u = u + 1$.
4. Design prototype masking subfilter. Complex-modulate its coefficients to obtain the remaining 2^u-1 masking subfilters in stage u.
5. Calculate transition width for next stage. Go to step 3.

Implementing the FFB in hardware is also not difficult, even when the number of channels is large. In fact, we can take advantage of the low sensitivity of the FFB's frequency response to variations in the masking subfilter coefficients. Very efficient hardware implementations can be obtained by using the signed-powers-of-two design method. This is further discussed in Chapter 6.

Implementing the FFB in software, on the other hand, presents more obstacles. As the number of stages increases, it can be seen that the number of subfilters increases drastically. Generally, the number of subfilters is one less than the number of channels. At each subfilter, the signal must be filtered and branched into 2 connecting subfilters at its output. This requires a significant amount of memory and control logic to store the intermediate signals, as well as to direct the flow of these signals. A matrix formulation method is discussed in Section 4.5 to overcome this, but it is still considerably more complicated to implement than the polyphase filter bank.

3.6 Summary

In this chapter, we briefly introduced the concept of Frequency Response Masking (FRM), and a class of filter banks that operate based on the FRM principle. We focused on the single-rate operation of the Fast Filter Bank (FFB), and formalized the representation and notations of the M-channel FFB. We considered the design aspects of the FFB, and imposed a set of conditions on the subfilters such that the masking subfilters effectively mask the unwanted passbands in the shaping stage. A relationship was established between the FFB and the FFT, when the number of non-zero coefficients in each subfilter is equal to 2.

We found that when designing the shaping subfilter $H_{0,0}(z)$, its transition width should not be greater than 0.3π for desirable overall complexity. Finally, we compared the single-rate design of a 32-channel FFB, 32-channel sliding FFT, and the 32-channel polyphase filter bank. The FFB is very efficient in single-rate applications and is highly suitable for designs which require very small transition widths.

Chapter 4

Single-rate Systems II: Simplifications of the Fast Filter Bank

In this chapter, we further look into the options that are available for the single-rate processing scenario. First, we consider the scenario when the input signal to the FFB is real-valued, and find that the FFB method of Chapter 3 presents an amount of redundant processing. Using our proposed node-modulation and pruning methods, we are able to reduce the amount of processing required. We then illustrate this with an example of a 16-channel odd-stacked FFB.

In the last part of the chapter, we propose a matrix method for processing the filtering operations. By formulating the data and filtering operations in terms of matrices and vectors, we are able to improve processing time on a computer platform using specialized software packages. We have published work on the pruning and node-modulation methods in [33]-[34], and the matrix method in [35].

4.1 Processing of real-valued input signals

In Chapter 3, we assumed the general scenario that the input to the FFB is complex-valued. In this section, we consider the scenario when the input is real-valued.

4.1.1 Even-stacked FFB

An even-stacked filter bank was defined to have frequency responses with passband center frequencies at $2\pi m/M$, where m is the subband index and M is the total number of subbands. For an even-stacked FFB, the frequency responses of the filter bank for subbands m=1 to m=(M/2-1) form a set of M/2-1 complex-conjugate pairs with the frequency responses of the filter bank for subbands m=(M/2+1) to m=M-1. The following property holds:

$$H_{[m]}(z) = H^{*}_{[M-m]}(z)\,,\ \text{for } 0 \le m \le M\,. \tag{4.1}$$

This is evident, since for even-stacked uniform filter banks:

$$\begin{aligned}H_{[m]}(z) &= H_{[0]}(e^{-j2\pi m/M}z)\\ &= H_{[0]}(e^{-j2\pi m/M}.e^{j2\pi M/M}z)\\ &= H_{[0]}(e^{-j2\pi(m-M)/M}z)\\ &= H^{*}_{[0]}(e^{-j2\pi(M-m)/M}z)\\ &= H^{*}_{[M-m]}(z).\end{aligned} \tag{4.2}$$

Note that $H_{[0]}(z)$ has real-valued coefficients, and $H_{[0]}(z) = H_{[M]}(z)$. Similarly, $H_{[M/2]}(z)$ has real-valued coefficients. On the other hand, $H_{[m]}(z)$ for $0 < m < M/2$ and $M/2 < m < M$ have complex-valued coefficients. Also, note that $H_{u,0}(z)$ has real-valued coefficients, and $H_{u,v}(z)$ for $v \neq 0$ have complex-valued coefficients.

When the input signal to the FFB is real-valued, the outputs $X_{[m]}(z) = H_{[m]}(z)X(z)$ form complex-conjugate sets such that $X_{[m]}(z) = X^{*}_{[M-m]}(z)$. Therefore, only $M/2+1$ unique outputs exist for $0 \leq m \leq M/2$. The outputs can be made to be real-valued by taking the real parts of the complex-valued outputs. The real-valued part of the outputs is given by:

$$\begin{aligned}X_{R[m]}(z) &= X_{[m]}(z) + X^{*}_{[m]}(z)\\ &= 2[\![X_{[m]}(z)]\!], \qquad \text{for } 1 \leq m \leq M/2-1,\end{aligned} \tag{4.3}$$

where $[\![.]\!]$ denotes taking the real part of the bracketed term.

The input-to-output transfer function from $X(z)$ to $X_{R[m]}(z)$ is then given by the real-valued part of the filter bank transfer function:

$$H_{R[m]}(z) = H_{[m]}(z), \text{ for } m = 0, M/2; \text{ and}$$

$$H_{R[m]}(z) = 2[\![H_{[m]}(z)]\!], \text{ for } 1 \leq m \leq M/2-1. \tag{4.4}$$

Using the FFB for processing real-valued input signals results in $M_R = 2^{K-1}+1$ subbands, which are even-stacked with center frequencies at $2\pi m/2^K$, for $0 \leq m \leq M_R - 1$. We note that the subbands m=0 and m=M_R-1 have only half of the bandwidth of the subbands from m=1 to m=M_R-2. This is illustrated in Figure 4-1(a).

4.1.2 Odd-stacked FFB

An odd-stacked filter bank was defined to have frequency responses with passband center frequencies at $\pi(2m+1)/M$, where m is the subband index and M is the total number of subbands. A modified form of the FFB can be obtained such that its frequency responses are odd-stacked, by defining the transfer function $H'_{[m]}(z)$ of the odd-stacked FFB to be:

$$H'_{[m]}(z) = H_{[m]}(e^{-j\pi/M}z), \text{ for } 0 \leq m \leq M, \tag{4.5}$$

where $H_{[m]}(z)$ represents the transfer function of the even-stacked FFB.

Statement: One method to satisfy (4.5) is by modulating $H_{u,v}(z^{2^{K-u-1}})$ by a frequency of π / M :

$$H'_{u,v}(z^{2^{K-u-1}}) = H_{u,v}\left((e^{-j\pi/M}z)^{2^{K-u-1}}\right), \tag{4.6}$$

which is equivalently represented by:

$$H'_{u,v}(z) = H_{u,v}(e^{-j\pi 2^{-(u+1)}}z). \tag{4.7}$$

Proof: Modifying the transfer function given in (3.20) by (4.5), we get:

$$\begin{aligned} H'_{[m]}(z) = \prod_{u=0}^{K-1}\Big[&\left(1-\alpha(m,u)\right)\left(H_{u,v(m,u)}\left((e^{-j\pi/M}z)^{\ell(K,u)}\right)\right) \\ &+\alpha(m,u)\left(\bar{H}_{u,v(m,u)}\left((e^{-j\pi/M}z)^{\ell(K,u)}\right)\right)\Big], \end{aligned} \tag{4.8}$$

and this is equivalent to (4.6).

The frequency responses of the odd-stacked FFB for *m*=0 to *m*=(*M*/2-1) form a set of *M*/2 complex-conjugate pairs with the frequency responses for *m*=*M*/2 to *m*=*M*-1. The following property holds:

$$H'_{[m]}(z) = H'^{*}_{[M-m-1]}(z), \text{ for } 0 \le m \le M-1. \tag{4.9}$$

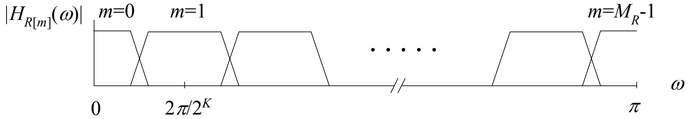
We note that $H'_{u,v}(z)$, for all *u* and *v*, have complex-valued coefficients. When the input signal to the FFB is real-valued, the outputs $X_{[m]}(z) = H'_{[m]}(z)X(z)$ form complex-conjugate sets given by $X_{[m]}(z) = X^{*}_{[M-m-1]}(z)$, and only *M*/2 unique outputs exist for $0 \le m \le M/2-1$. The outputs can be made to be real-valued by taking the real parts of the complex-valued outputs:

$$X_{R[m]}(z) = 2\llbracket X_{[m]}(z)\rrbracket, \text{ for } 0 \le m \le M/2-1. \tag{4.10}$$

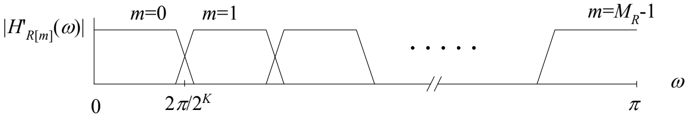
The transfer functions of the filter bank are then given by:

$$H'_{R[m]}(z) = 2\llbracket H'_{[m]}(z)\rrbracket, \text{ for } 0 \le m \le M/2-1. \tag{4.11}$$

Using the odd-stacked FFB for processing real-valued input signals results in $M_R = 2^{K-1}$ subbands, which have center frequencies at $2\pi(m+0.5)/2^K$, for $0 \le m \le M_R - 1$. The subbands have equal bandwidths. This is illustrated in Figure 4-1(b). We note that the odd-stacked form of the FFB is interesting in our work because audio filter banks (such as those used in MPEG 1 and 2) are also odd-stacked.



(a) Even-stacked



(b) Odd-stacked

Figure 4-1 Even and odd stacked frequency responses for real-valued inputs.

4.2 Node-modulation method

The FFB comprises subfilters with transfer functions $H_{u,v}(z)$ which are modulated versions of $H_{u,0}(z)$ (which has real-valued coefficients), and they generally have complex-valued coefficients. In this section, we propose a method that reduces these subfilters to have real-valued coefficients. The resultant implementation has a lower complexity when compared to the original FFB. We consider the even-stacked FFB, and then extend the method to the odd-stacked FFB. The method is applicable to both real-valued input and complex-valued input signals. Here, we consider the more general case when the input signal is complex-valued.

We note that the input signal $x_{u,v}(n)$ and output signal $y_{u,v}(n)$ of the subfilter with transfer function $H_{u,v}(z)$ are related according to:

$$Y_{u,v}(z) = H_{u,v}(z^{\ell(u)})X_{u,v}(z), \tag{4.12}$$

and in the time-domain for odd-length filters (non-causal form) according to:

$$\begin{aligned} y_{u,v}(n) &= \sum_{n'=-(N-1)/2}^{(N-1)/2} h_{u,v}(n')x_{u,v}\left(n-\ell(u)n'\right) \\ &= \sum_{n'} e^{j2\pi\tilde{v}n'/M} h_{u,0}(n')x_{u,v}\left(n-\ell(u)n'\right) \\ &= e^{-j2\pi\tilde{v}n'/\ell(u)M} \sum_{n'} h_{u,0}(n')e^{-j2\pi\tilde{v}(n-\ell(u)n')/\ell(u)M} x_{u,v}\left(n-\ell(u)n'\right). \end{aligned} \tag{4.13}$$

Thus, we can reduce the operation of filtering $X_{u,v}(z)$ (complex-valued coefficients) with $H_{u,v}(z)$ (complex-valued coefficients) into an operation of filtering $X_{u,v}(z)$ (complex-valued coefficients) with $H_{u,0}(z)$ (real-valued coefficients) by adopting the following strategy:

For practical purposes, we consider the causal implementation of $H_{u,v}(z)$ here. First, modulate the input signal to the subfilter by a modulating signal $\lambda_{u.v}(n)$:

$$x'_{u,v}(n) = \lambda_{u,v}(n) x_{u,v}(n), \tag{4.14}$$

where $x'_{u,v}(n)$ is the modulated input signal to the subfilter with transfer function $H_{u,0}(z)$ and the modulating signal is given by:

$$\lambda_{u,v}(n) = e^{-j2\pi\tilde{v}n/\ell(u)M}. \tag{4.15}$$

The modulated input signal is then passed into the subfilter with transfer function $H_{u,0}(z)$ and the output is denoted by $y'_{u,v}(n)$. The desired $y_{u,v}(n)$ can be obtained by demodulating $y'_{u,v}(n)$:

$$y_{u,v}(n) = \lambda_{u,v}\left(-(n-d_u)\right) y'_{u,v}(n), \tag{4.16}$$

where d_u is the delay of the interpolated subfilter with transfer function $H_{u,0}(z^{\ell(u)})$.

If we denote the complexity of the first method by Γ ($X_{u,v}(z)$ has complex-valued coefficients and $H_{u,v}(z)$ has complex-valued coefficients), then the complexity of the second method ($X_{u,v}(z)$ has complex-valued coefficients and $H_{u,v}(z)$ has real-valued coefficients) is approximately equal to $\Gamma/2+8$.

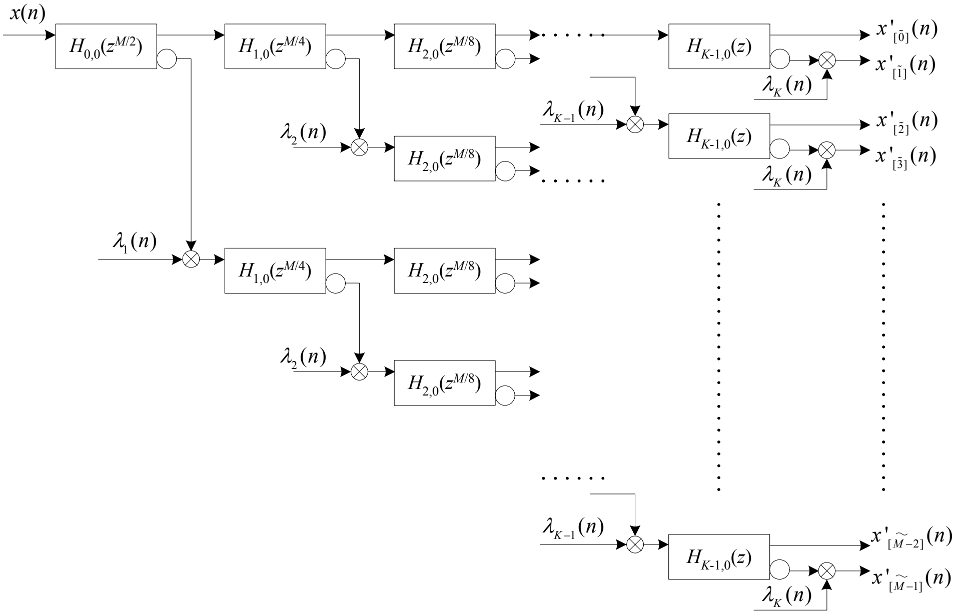


Figure 4-2 Scheme for the proposed node-modulated even-stacked FFB.

Figure 4-2 shows our proposed node-modulated FFB. The subfilters with transfer functions $H_{u,v}(z)$ are replaced by subfilters with transfer functions $H_{u,0}(z)$, and a modulation node $\lambda_u(n)$ is inserted prior to each subfilter with transfer function $H_{u,v}(z)$, where v is odd. Alternatively, one can view the modulation node $\lambda_{u+1}(n)$ as being inserted at the complementary output of each subfilter with transfer function $H_{u,v}(z)$. For the causal form of the FFB,

$$\lambda_{u+1}(n) = e^{-j\pi 2^{u+1-K}\left(n-\sum_{u'=0}^{u} d_{u'}\right)}, \text{ for } 0 \le u \le K-1. \tag{4.17}$$

By performing node-modulation for the complementary outputs of all subfilters of the FFB for $0 \le u \le K-1$, $H_{u,v}(z)$ (which has complex-valued coefficients) can be replaced by $H_{u,0}(z)$ (which has real-valued coefficients). Furthermore, the outputs of the filter bank $x'_{[m]}(n)$ will be baseband signals, i.e. their centre frequencies are zero. The output signals can be brought back to their original frequencies by applying a demodulating signal given by:

$$x_{[m]}(n) = \lambda'_{[m]}(n) x'_{[m]}(n), \tag{4.18}$$

where:

$$\lambda'_{[m]}(n) = e^{j\frac{m\pi}{2^{K-1}}\left(n-\sum_{u=0}^{K-1} d_u\right)}. \tag{4.19}$$

It is more efficient, however, to perform node-modulation for the complementary outputs of the subfilters with transfer function $H_{u,v}(z)$ for $0 \le u \le K-2$ only. The filter bank outputs $x'_{[m]}(n)$ will then have center frequencies equal to zero for $0 \le m \le M/2-1$, and equal to π for $M/2 \le m \le M-1$. The demodulating signal is then given by:

$$\lambda'_{[m]}(n) = e^{j\frac{2m\pi}{M}\left(n-\sum_{u=0}^{K-1} d_u\right)}, \text{ for } 0 \le m \le M/2-1, \tag{4.20}$$

and:

$$\lambda'_{[m]}(n) = e^{j\frac{(2m-M)\pi}{M}\left(n-\sum_{u=0}^{K-1} d_u\right)}, \text{ for } M/2 \le m \le M-1. \tag{4.21}$$

For the odd-stacked FFB, the same principle can be applied. However, since the transfer functions of the subfilters in the odd-stacked FFB are related to those of the even-stacked FFB by (4.7), the modulating signals are modified to:

$$\lambda_0(n) = e^{-j\pi 2^{-K} n},$$

$$\lambda_{u+1}(n) = e^{-j\pi 2^{u+1-K}\left(n-\sum_{u'=0}^{u} d_{u'}\right)}, \text{ for } 0 \le u \le K-1. \tag{4.22}$$

The structure of the node-modulated odd-stacked FFB is shown in Figure 4-3. The even-stacked, node-modulated FFB block in the diagram can be replaced with the structure shown in Figure 4-2.

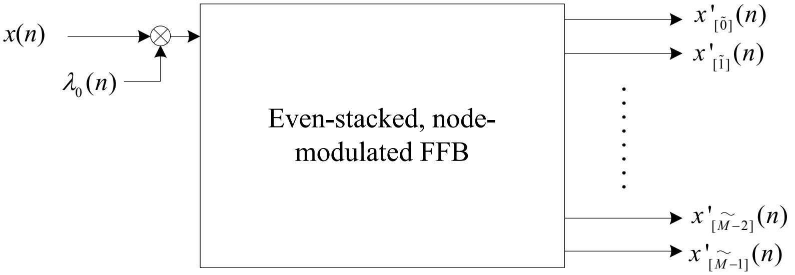

$x(n)$
$\lambda_0(n)$
Even-stacked, node-modulated FFB
$x'_{[\tilde{0}]}(n)$
$x'_{[\tilde{1}]}(n)$
$x'_{[\widetilde{M-2}]}(n)$
$x'_{[\widetilde{M-1}]}(n)$

Figure 4-3 Scheme for the proposed node-modulated odd-stacked FFB.

The demodulating signals are modified to:

$$\begin{aligned}\lambda'_{[m]}(n) &= e^{j\left(\frac{m\pi}{2^{K-1}}+\frac{\pi}{2^K}\right)\left(n-\sum_{u=0}^{K-1} d_u\right)} \\ &= e^{j\frac{m\pi}{2^{K-1}}\left(n-\sum_{u=0}^{K-1} d_u\right)} + e^{j\frac{\pi}{2^K}\left(n-\sum_{u=0}^{K-1} d_u\right)}. \end{aligned} \tag{4.23}$$

In the case when a synthesis filter bank is used to reconstruct the subband signals, and assuming that the synthesis filter bank is also implemented using node-modulation, the demodulation at the output stage of the analysis filter bank and at the input stage of the synthesis filter bank cancel each other and can be omitted.

4.3 Pruning method

In this section, we propose a pruning method for the FFB when the input signal $x(n)$ is real-valued. The pruning method involves the removal of redundant subfilters that lead to non-unique outputs for the real-valued input signal scenario. We look at both the even-stacked and odd-stacked FFB, and use a "bit-pattern" analysis to determine the redundant subfilters.

4.3.1 Even-stacked FFB

In the real-valued input signal scenario, the output signal at the $(2^K - m)^{\text{th}}$ subband is the complex-conjugate of the output signal at the m^{th} subband, i.e. $x_{[2^K-m]}(n) = x^*_{[m]}(n)$, and there are only $2^{K-1}+1$ independent outputs. The real-valued output signal at the m^{th} subband can be obtained by simply dropping the imaginary part of the complex-valued output signal.

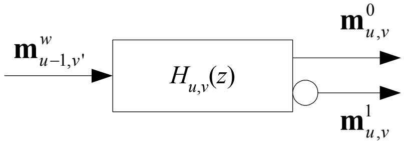


Figure 4-4 Bit-pattern used for a subfilter.

The M output subbands of the FFB, indexed by m ($0 \le m \le M-1$), are ordered in bit-reversed fashion in K bits. Let $\mathbf{m}_{bin}$ denote the binary representation of m. Each subfilter with the transfer function $H_{u,v}(z)$ has 2 output paths. Let us denote the upper path of the output of $H_{u,v}(z)$ using the multi-bit-pattern $\mathbf{m}_{u,v}^{0}$ (expressed as a row vector) and the lower path of the output of $H_{u,v}(z)$ using the multi-bit-pattern $\mathbf{m}_{u,v}^{1}$. This is shown in Figure 4-4. Define the single-bit terms $m_{0,0}^{0} = 0$ and $m_{0,0}^{1} = 1$. Assuming that the input to $H_{u,v}(z)$ is denoted as $\mathbf{m}_{u-1,v'}^{w}$, we can describe the outputs of $H_{u,v}(z)$ using the bit-pattern:

$$\mathbf{m}_{u,v}^{0} = [0 \quad \mathbf{m}_{u-1,v'}^{w}], \tag{4.24}$$

and:

$$\mathbf{m}_{u,v}^{1} = [1 \quad \mathbf{m}_{u-1,v'}^{w}]. \tag{4.25}$$

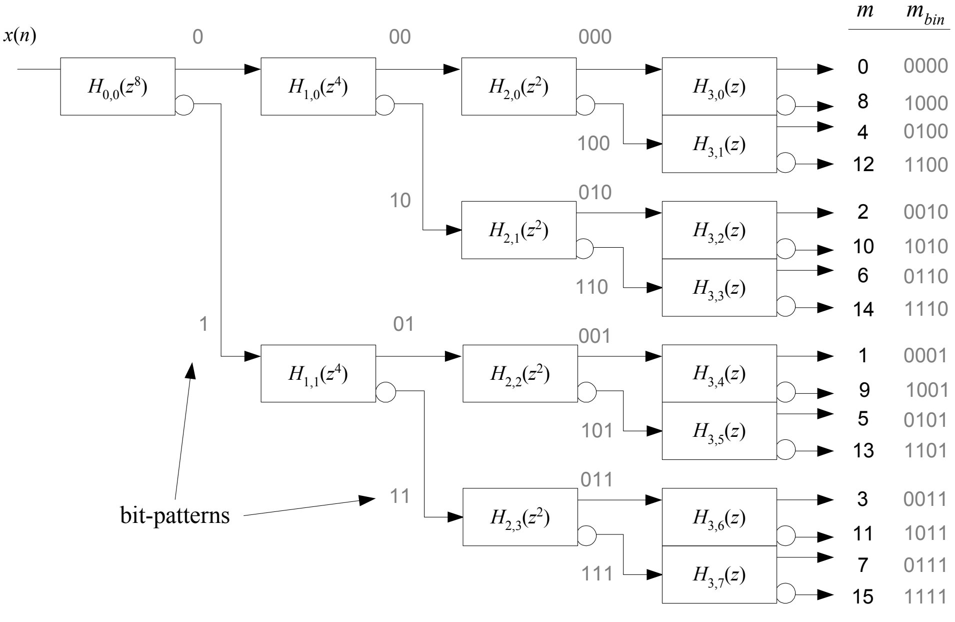


Figure 4-5 A 16-channel FFB with bit-pattern annotations.

Figure 4-5 shows the bit-patterns for a 16-channel FFB. Propagating the bit-pattern for each of the cascaded subfilters of the FFB finally leads to the bit-pattern $\mathbf{m}_{bin}$. As an example, the output signal of the 16-channel FFB for m=5 is the result of propagation through the subfilters with transfer functions $\overline{H}_{0,0}(z^8)$, $H_{1,1}(z^4)$, $\overline{H}_{2,2}(z^2)$ and $H_{3,5}(z)$. The bit-patterns are $m_{0,0}^{1} = '1'$, $\mathbf{m}_{1,1}^{0} = '01'$, $\mathbf{m}_{2,2}^{1} = '101'$ and $\mathbf{m}_{3,5}^{0} = '0101'$ respectively. Therefore, the output of the subfilter with transfer function $H_{3,5}(z)$ has a bit-pattern '0101'. This corresponds to the value of m=5, which converts to the binary value of $\mathbf{m}_{bin}$='0101'.

It can be further verified that the input to $H_{u,v}(z)$, which has the bit-pattern $\mathbf{m}^{w}_{u-1,v'}$, is a u-bit integer $\tilde{v}$ which is the bit-reversed version of v in u bits. Thus, (4.24) and (4.25) can be alternatively expressed as:

$$\mathbf{m}^{0}_{u,v} = [0 \quad \tilde{v}], \tag{4.26}$$

and:

$$\mathbf{m}^{1}_{u,v} = [1 \quad \tilde{v}]. \tag{4.27}$$

In the real-valued input signal scenario, the outputs $x_{[m]}(n)$ and $x_{[2^K-m]}(n)$ of the even-stacked FFB are complex-conjugates of each other. Thus, if the output $x_{[m]}(n)$ is computed, all the processing that finally leads only to the output $x_{[2^K-m]}(n)$ may be removed.

Statement: If $\left(\mathbf{m}^{w}_{u,v} + \mathbf{m}^{w'}_{u,v'}\right)$ truncated to u+1 bits is equal to 0, then either the signal path with bit-pattern $\mathbf{m}^{w}_{u,v}$, or the signal path with bit-pattern $\mathbf{m}^{w'}_{u,v'}$ may be removed.

Proof: Assume that $\left(\mathbf{m}^{w}_{u,v} + \mathbf{m}^{w'}_{u,v'}\right)$ truncated to u+1 bits is equal to 0, and since both $\mathbf{m}^{w}_{u,v}$ and $\mathbf{m}^{w'}_{u,v'}$ are non-negative (i.e. $\left(\mathbf{m}^{w}_{u,v} + \mathbf{m}^{w'}_{u,v'}\right) \neq [0 \quad \mathbf{0}_{u+1}]$), then:

$$\left(\mathbf{m}^{w}_{u,v} + \mathbf{m}^{w'}_{u,v'}\right) = [1 \quad \mathbf{0}_{u+1}], \tag{4.28}$$

where $\mathbf{0}_{u+1}$ is a bit-pattern consisting of u+1 zeros.

The signal path with bit-pattern $\mathbf{m}^{w}_{u,v}$ leads to the signal paths with bit-patterns $[1 \quad \mathbf{m}^{w}_{u,v}]$ and $[0 \quad \mathbf{m}^{w}_{u,v}]$. The signal path with bit-pattern $\mathbf{m}^{w'}_{u,v'}$ leads to the signal paths with bit-patterns $[1 \quad \mathbf{m}^{w'}_{u,v'}]$ and $[0 \quad \mathbf{m}^{w'}_{u,v'}]$. From (4.28), we find that:

$$\begin{aligned}[0 \quad \mathbf{m}^{w}_{u,v}] + [1 \quad \mathbf{m}^{w'}_{u,v'}] &= \left[1 \quad 0 \quad \mathbf{0}_{u+1}\right] \\ &= \left[1 \quad \mathbf{0}_{u+2}\right],\end{aligned} \tag{4.29}$$

and:

$$\begin{aligned}[1 \quad \mathbf{m}^{w}_{u,v}] + [0 \quad \mathbf{m}^{w'}_{u,v'}] &= \left[1 \quad 0 \quad \mathbf{0}_{u+1}\right] \\ &= \left[1 \quad \mathbf{0}_{u+2}\right].\end{aligned} \tag{4.30}$$

Applying the principle in (4.29) and (4.30) recursively: the signal path with bit-pattern $\mathbf{m}^{w}_{u,v}$ leads to the signal path with bit-pattern $[\mathbf{b} \quad \mathbf{m}^{w}_{u,v}]$ (where $\mathbf{b}$ is an arbitrary row vector of binary bits) and the signal path with bit-pattern $\mathbf{m}_{u,v'}^{w'}$ leads to the signal path with bit-pattern $[\mathbf{b}' \quad \mathbf{m}_{u,v'}^{w'}]$ (where **b'** is an arbitrary row vector of binary bits), such that for every vector **b**, there exists a corresponding vector **b'** satisfying the condition:

$$[\mathbf{b} \quad \mathbf{m}_{u,v}^{w}] + [\mathbf{b}' \quad \mathbf{m}_{u,v'}^{w'}] = \begin{bmatrix} 1 & \mathbf{0} \end{bmatrix}, \tag{4.31}$$

if $\left(\mathbf{m}_{u,v}^{w} + \mathbf{m}_{u,v'}^{w'}\right) = [1 \quad \mathbf{0}_{u+1}]$.

The value of $(2^K\text{-}m)+m$ is equal to 2^K, which when truncated to K bits is equal to 0. The signals $x_{[m]}(n)$ and $x_{[2^K-m]}(n)$ are complex-conjugates, and therefore any one of them can be removed without data loss.

Based on the arguments above, we conclude that for a signal $x_{[m]}(n)$, which passes through the signal path with bit-pattern $\mathbf{m}_{u,v}^{w}$, its complex-conjugate signal $x_{[2^K-m]}(n)$ must pass through the signal path with bit-pattern $\mathbf{m}_{u,v'}^{w'}$, where $\left(\mathbf{m}_{u,v}^{w} + \mathbf{m}_{u,v'}^{w'}\right) = [1 \quad \mathbf{0}_{u+1}]$.

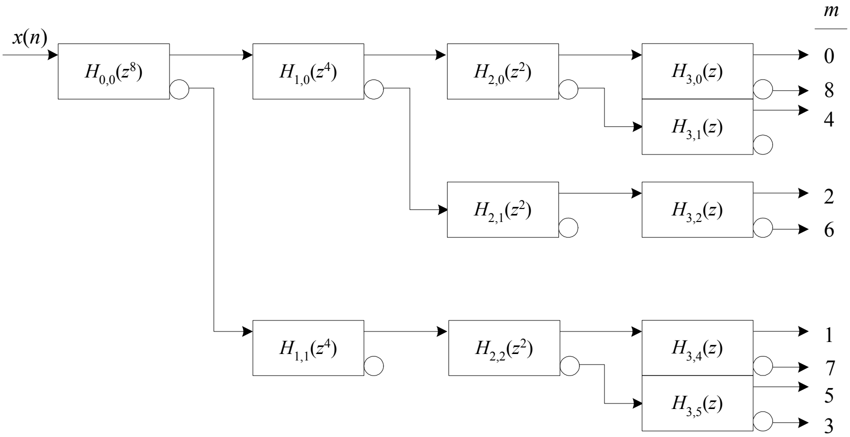


Figure 4-6 Pruning for an even-stacked 16-channel FFB.

If $\left(\mathbf{m}_{u,v}^{w} + \mathbf{m}_{u,v'}^{w'}\right)$ truncated to u+1 bits is equal to 0, then either the signal path with bit-pattern $\mathbf{m}_{u,v}^{w}$ or $\mathbf{m}_{u,v'}^{w'}$ may be removed. The signal paths with bit-patterns $\mathbf{m}_{u,v}^{0} = [0 \ 1 \ \mathbf{0}_{u-1}]$ and $\mathbf{m}_{u,v}^{1} = [1 \ 1 \ \mathbf{0}_{u-1}]$ form such a pair, where $\mathbf{0}_{u-1}$ is a bit-pattern consisting of u-1 zeros. Figure 4-6 shows the scenario where signal paths with bit-patterns $\mathbf{m}_{u,v}^{1} = [1 \ 1 \ \mathbf{0}_{u-1}]$ are removed. Alternatively, signal paths with bit-patterns $\mathbf{m}_{u,v}^{0} = [0 \ 1 \ \mathbf{0}_{u-1}]$ may be removed instead.

We can derive a simple rule-of-thumb based on the above argument for pruning the signal paths by removing the fourth subfilter $H_{u,3}(z)$ for each stage *u*, and all subsequent subfilters that are attached to it. The number of subfilters required for stage *u* is now given by $2^{u-1}+1$. The savings achieved for each stage *u* are $(2^{u-1}-1)/2^{u}$ for $1 \le u \le K-1$. Figure 4-6 shows a pruned even-stacked 16-channel FFB.

4.3.2 Odd-stacked FFB

For the implementation of the odd-stacked FFB, the efficiency of the pruning method can be further improved. Assume the case in Section 4.1.2, where each subfilter is modified by $H'_{u,v}(z) = H_{u,v}(e^{-j\pi 2^{-(u+1)}} z)$. Following the same reasoning as for the even-stacked FFB, we are able to prune the entire propagation path attached to the complementary output of $H_{0,0}(z)$. Figure 4-7 shows a pruned odd-stacked 16-channel FFB. The resultant complexity of the filter bank is approximately halved.

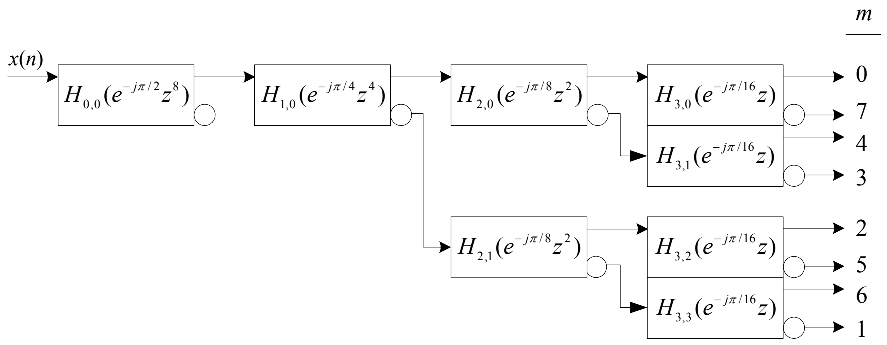


Figure 4-7 Pruning for an odd-stacked FFB with input modulation.

4.4 Odd-stacked 16-channel FFB

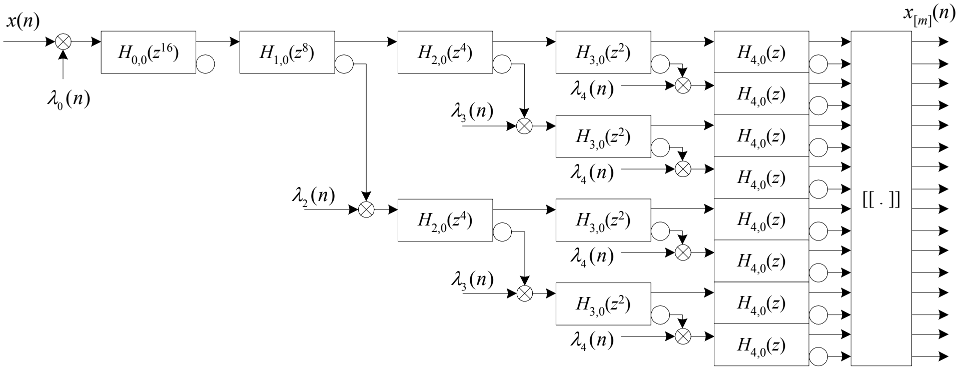


Figure 4-8 Using pruning and node-modulation to implement an efficient odd-stacked 16-channel FFB.

We consider the design of an odd-stacked 16-channel FFB for a real-valued input signal. We make use of the design in Section 3.5 (32-channel FFB for complex-valued input signal) and perform our proposed pruning and node-modulation techniques. The coefficients of the subfilters $H_{u,0}(z)$ were taken from Table 3-3. The design is shown in Figure 4-8.

The lengths of the subfilters $H_{0,0}(z)$, $H_{1,0}(z)$, ..., $H_{4,0}(z)$ are 31, 15, 7, 7, 3 respectively. A breakdown of the complexity by stage is tabulated in Table 4-1. The methods used to calculate the complexity of the subfilters are outlined in Appendix B. The proposed scheme has a complexity of about half of the original FFB.

The estimated length of the prototype filter required for a polyphase filter bank with the same specification is 521. The estimated total complexity for the polyphase filter bank would be approximately 553.

	Complexity breakdown by stage, u					
Method	0	1	2	3	4	Total
FFB	16	32	32	64	32	176
Pruning	16	16	16	32	16	96
Pruning + node-modulation	20	8	12	24	24	88

Table 4-1 Complexity breakdown by stage for pruned node-modulated FFB.

4.5 A matrix formulation

The FFB is organized in a tree structure. When the FFB is implemented on a computer or a digital signal processor, using a software programming method, we have to allocate resources to manage the signal flow through the individual subfilters. Data buffers of varying sizes must be created for each subfilter, due to the varying filter lengths. When the number of subbands M increases, allocating, addressing and manipulating these buffers becomes cumbersome. Thus, it is convenient to formulate the FFB processing in a matrix form to facilitate data processing. Using the node-modulation method, we find that we can express the filtering operation conveniently using 2-D matrices. Furthermore, only one data buffer needs to be implemented per stage u, instead of 2^u data buffers required per stage u previously.

In modern computers, the limiting factors in processing are usually not due to the rate at which arithmetic operations are performed, but rather due to the efficiency at which data accesses and transfers to and from memory occurs [36]-[37]. This is especially relevant in our filter bank processing, in which a lot of data is manipulated. By arranging the data in vectors and matrices, data pipelining and efficient memory usage is facilitated within the hardware. Furthermore, many software mathematical packages are available for computing matrix operations efficiently. These mathematical packages are highly-optimized for specific processors and make use of the special facilities and architectures inherent in the target processors. One such popular package is the Basic Linear Algebra Subprograms (BLAS) [38].

4.5.1 Matrix formulation for a subfilter

We denote the input signal into the subfilter with transfer function $H_{u,v}(z)$ as $x_{u,v}(n)$. The output signal is denoted as $x_{u+1,2v}(n)$, and the complementary output signal $\overline{x}_{u+1,2v}(n)$ as $x_{u+1,2v+1}(n)$. These signals are then processed as input signals into the subfilters with transfer functions $H_{u+1,2v}(z)$ and $H_{u+1,2v+1}(z)$ respectively. The delay of the interpolated subfilter with transfer function $H_{u,v}(z^{\ell(u)})$ is denoted as d_u. We assume the general complex-valued input signal scenario. If we further assume that $N_{u,v}$ is odd and that Condition 4 listed in Section 3.3.1 is true, then:

$$\overline{x}_{u+1,2v}(n) = x_{u,v}\left(n-d_u\right) - x_{u+1,2v}(n)\,. \tag{4.32}$$

The filtering operation for $H_{u,v}(z)$ is described by:

$$x_{u+1,2v}(n) = \sum_{n'=0}^{N_u-1} h_{u,v}(n')x_{u,v}\left(n-\ell(u)n'\right). \tag{4.33}$$

First, we arrange the input signal ranging from $x_{u,v}(n)$ to $x_{u,v}\left(n-N_u\ell(u)+1\right)$ in a $\ell(u)\times N_u$ matrix:

$$\mathbf{X}_{u,v}(n) = \left[\mathbf{x}_{u,v}(n) \quad \mathbf{x}_{u,v}\left(n-\ell(u)\right) \quad \cdots \quad \mathbf{x}_{u,v}\left(n-(N_u-1)\ell(u)\right)\right], \ \in \mathbb{C}^{\ell(u)\times(N_u-1)}, \tag{4.34}$$

where $\mathbf{x}_{u,v}(n)$ is a column vector of length $\ell(u)$:

$$\mathbf{x}_{u,v}(n) = \left[x_{u,v}(n) \quad x_{u,v}(n-1) \quad \cdots \quad x_{u,v}(n-\ell(u)+1)\right]^T. \tag{4.35}$$

The coefficients of $H_{u,v}(z)$ are arranged in a column vector of length N_u:

$$\mathbf{h}_{u,v} = \left[h_{u,v}(0) \quad h_{u,v}(1) \quad \cdots \quad h_{u,v}(N_u-1)\right]^T. \tag{4.36}$$

We then express the filtering operation in (4.33) in matrix form as:

$$\begin{bmatrix} \mathbf{x}_{u+1,2v}(n) \\ \mathbf{x}_{u+1,2v}\left(n-\ell(u+1)\right) \end{bmatrix} = \mathbf{X}_{u,v}(n)\mathbf{h}_{u,v}\,, \tag{4.37}$$

where the left-hand-side result is a column vector of length $\ell(u)$. Since $\ell(u)=2\ell(u+1)$, we divide this vector into 2 sub-vectors of length $\ell(u+1)$.

From (4.32), we express the complementary output of the subfilter as:

$$\begin{bmatrix} \mathbf{x}_{u+1,2v+1}(n) \\ \mathbf{x}_{u+1,2v+1}\left(n-\ell(u+1)\right) \end{bmatrix} = \mathbf{x}_{u,v}\left(n-d_u\right) - \begin{bmatrix} \mathbf{x}_{u+1,2v}(n) \\ \mathbf{x}_{u+1,2v}\left(n-\ell(u+1)\right) \end{bmatrix}, \tag{4.38}$$

where $d_u = \ell(u)(N_u-1)/2$ is the delay of the subfilter.

We perform the operations in (4.37) and (4.38) for $n = 0, \ell(u), 2\ell(u), \ldots$. In using this form, we easily avoid redundant multiplications for the zero-valued coefficients. The input to the subfilter is organized into a $\ell(u) \times N_u$ buffer, $\mathbf{X}_{u,v}$. The input signal $x_{u,v}(n)$ is shifted ($\ell(u)$ samples at a time) into the buffer $\mathbf{X}_{u,v}$.

4.5.2 Matrix formulation for the filter bank

From observation of the tree structure of the FFB, we realize that a straightforward method of implementing the filter bank processing would be to allocate 2^u sets of $\mathbf{X}_{u,v}$ and $\mathbf{h}_{u,v}$ matrices per stage and performing the convolution individually. Instead, we propose a method that requires only 1 set of $\mathbf{X}_u$ and $\mathbf{h}_u$ matrices per stage. We note that it is also desirable to have the same number of rows for all the $\mathbf{X}_u$ matrices.

First, we refer to the node-modulation method proposed earlier. Using this method, we can reduce the subfilters such that the coefficient vector $\mathbf{h}_{u,v}$=$\mathbf{h}_{u,0}$. We denote this coefficient vector as $\mathbf{h}_u$. Furthermore, $\mathbf{h}_u$ is a real vector, i.e. only 1 set of real-valued coefficients is now required per stage u. Let us next attempt to arrange the data matrices. Since there are 2^u sets of $\mathbf{X}_{u,v}$ and each has $\ell(u) = 2^{K-u-1}$ rows, we can group all the 2^u $\mathbf{X}_{u,v}$ matrices for the same u and arrange them into a single $(M/2) \times N_u$ matrix. We first define the column vector $\mathbf{x}_u(n)$ of length $M/2$ such that:

$$\mathbf{x}_u(n) = \begin{bmatrix} \mathbf{x}_u^{(n)} \\ \mathbf{x}_u^{(n-1)} \\ \vdots \\ \mathbf{x}_u^{(n-\ell(u)+1)} \end{bmatrix}, \tag{4.39}$$

where:

$$\mathbf{x}_u^{(n)} = \begin{bmatrix} x_{u,0}(n) & x_{u,1}(n) & \cdots & x_{u,2^u-1}(n) \end{bmatrix}^T. \tag{4.40}$$

We can thus form the matrix $\mathbf{X}_u(n)$ as:

$$\mathbf{X}_u(n) = \begin{bmatrix} \mathbf{x}_u(n) & \mathbf{x}_u\left(n-\ell(u)\right) & \cdots & \mathbf{x}_u\left(n-(N_u-1)\ell(u)\right) \end{bmatrix}. \tag{4.41}$$

The matrix $\mathbf{X}_u(n)$ is updated every $\ell(u)$ samples. The most recent $\ell(u)$ samples of $\mathbf{x}_u(n)$ are shifted into the data buffer of $\mathbf{X}_u(n)$ and the last column of $\mathbf{X}_u(n)$ is shifted out. This is illustrated in Figure 4-9.

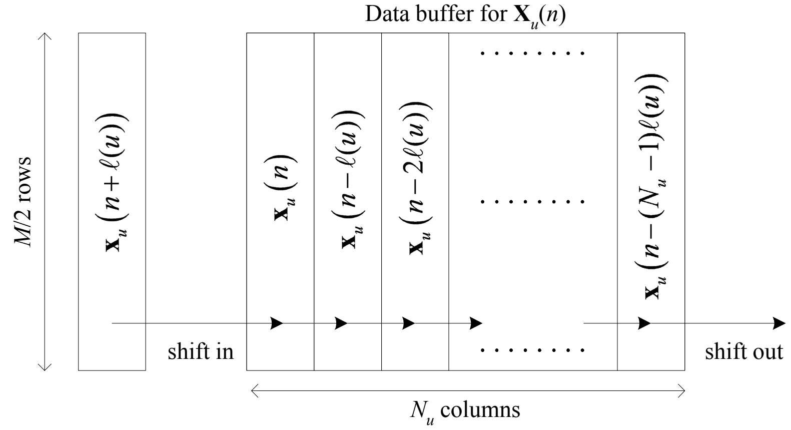

Data buffer for $\mathbf{X}_u(n)$
$M/2$ rows
$\mathbf{x}_u(n+\ell(u))$
$\mathbf{x}_u(n)$
$\mathbf{x}_u(n-\ell(u))$
$\mathbf{x}_u(n-2\ell(u))$
$\mathbf{x}_u(n-(N_u-1)\ell(u))$
shift in
shift out
N_u columns

Figure 4-9 Data buffer shifting.

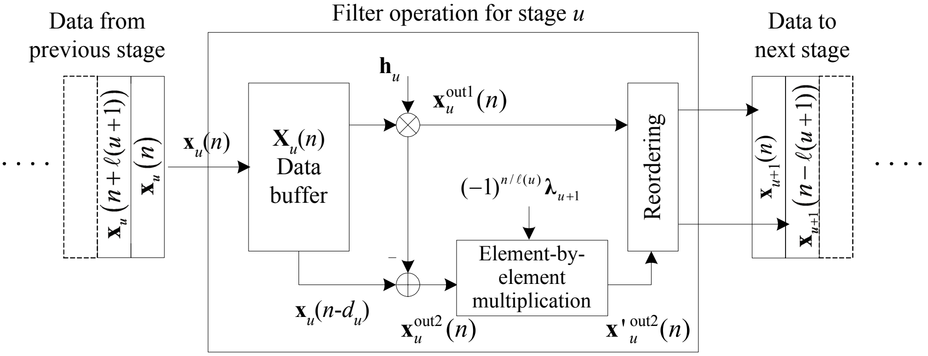

Data from previous stage
Filter operation for stage u
Data to next stage
$\mathbf{x}_u(n+\ell(u+1))$
$\mathbf{x}_u(n)$
$\mathbf{x}_u(n)$
$\mathbf{X}_u(n)$ Data buffer
$\mathbf{h}_u$
$\mathbf{x}_u^{\text{out1}}(n)$
$(-1)^{n/\ell(u)}\boldsymbol{\lambda}_{u+1}$
Reordering
Element-by-element multiplication
$-$
$\mathbf{x}_u(n\text{-}d_u)$
$\mathbf{x}_u^{\text{out2}}(n)$
$\mathbf{x}'^{\text{out2}}_u(n)$
$\mathbf{x}_{u+1}(n)$
$\mathbf{x}_{u+1}(n-\ell(u+1))$

Figure 4-10 Overview of matrix method for stage *u*.

Figure 4-10 shows an overview of our proposed method for a single stage. We perform the filter bank operations on a stage-by-stage basis and repeat for *u* from 0 to *K*-1. Filtering $\mathbf{X}_u$ with $\mathbf{h}_u$, we get a column vector $\mathbf{x}_u^{\text{out1}}(n)$ of length $M/2$:

$$\mathbf{x}_u^{\text{out1}}(n) = \mathbf{X}_u(n)\mathbf{h}_u . \tag{4.42}$$

From (4.38), the complementary output which we denote by $\mathbf{x}_u^{\text{out2}}(n)$ can be obtained from:

$$\mathbf{x}_u^{\text{out2}}(n) = \mathbf{x}_u(n-d_u) - \mathbf{x}_u^{\text{out1}}(n). \tag{4.43}$$

From Section 4.2, the complementary output should be modulated by a signal $\lambda_{u+1}(n)$. We note that the modulating signal $\lambda_{u+1}(n)$ is periodic with period $2\ell(u)$, and hence $\lambda_{u+1}(n) = \lambda_{u+1}(n-2\ell(u))$. Furthermore, $\lambda_{u+1}(n) = -\lambda_{u+1}(n-\ell(u))$.

Let us define the modulating signal as a column vector $\boldsymbol{\lambda}_{u+1}$ with a length of $M/2$, where:

$$\boldsymbol{\lambda}_{u+1} = \begin{bmatrix} \boldsymbol{\lambda}_{u+1}^{(0)} \\ \boldsymbol{\lambda}_{u+1}^{(1)} \\ \vdots \\ \boldsymbol{\lambda}_{u+1}^{(\ell(u)-1)} \end{bmatrix}, \tag{4.44}$$

and $\boldsymbol{\lambda}_{u+1}^{(n)}$ is a length 2^u column vector made up of repeating elements of $\lambda_{u+1}(n)$:

$$\boldsymbol{\lambda}_{u+1}^{(n)} = \left[\lambda_{u+1}(n) \quad \lambda_{u+1}(n) \quad \cdots \quad \lambda_{u+1}(n)\right]^T. \tag{4.45}$$

Modulation can then be achieved by the element-by-element multiplication of the 2 column vectors:

$$\mathbf{x}'^{\text{out2}}_u(n) = \mathbf{x}^{\text{out2}}_u(n) \cdot * (-1)^{n/\ell(u)} \boldsymbol{\lambda}_{u+1}, \tag{4.46}$$

where $\cdot *$ represents the element-by-element multiplication operator.

The input vectors to the next stage $\mathbf{x}_{u+1}(n)$ and $\mathbf{x}_{u+1}\left(n-\ell(u+1)\right)$ can be formed from $\mathbf{x}^{\text{out1}}_u(n)$ and $\mathbf{x}'^{\text{out2}}_u(n)$, where the even rows of $\mathbf{x}_{u+1}(n)$ and $\mathbf{x}_{u+1}\left(n-\ell(u+1)\right)$ are formed from $\mathbf{x}^{\text{out1}}_u(n)$ and the odd rows are formed from $\mathbf{x}'^{\text{out2}}_u(n)$. The re-ordering of the vectors is described in Figure 4-11.

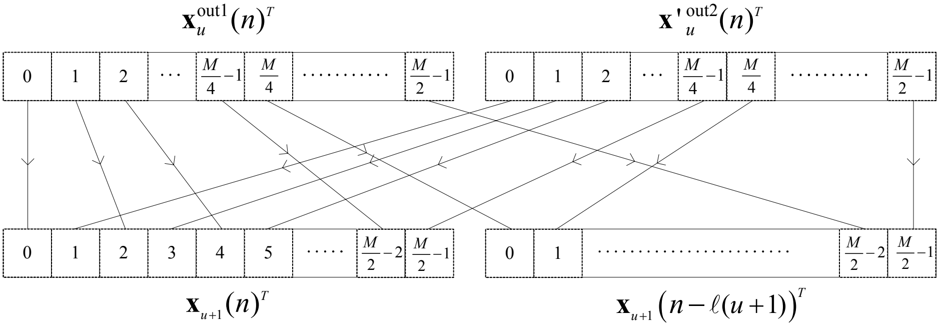


Figure 4-11 Re-ordering of data vectors.

4.5.3 *Algorithm*

An algorithm making use of recursive functions can be used to implement the data processing for the matrix method. Our proposed recursive function 'subfilter' is presented below. The initial inputs to the function are $\mathbf{x}_0(n)$ and u=0. The outputs $\mathbf{x}_K(n)$ are obtained after recursive iterations for $0 \le u \le K-1$.

```
function: subfilter ($\mathbf{x}_u(n)$, $u$)
    shift $\mathbf{x}_u(n)$ into buffer $\mathbf{X}_u(n)$;
    multiply $\mathbf{X}_u(n)\mathbf{h}_k$, obtain $\mathbf{x}_u^{\text{out1}}(n)$ and $\mathbf{x}_u^{\text{out2}}(n)$;
    modulate $\mathbf{x}_u^{\text{out2}}(n)$, obtain $\mathbf{x}'^{\text{out2}}_u(n)$;
    reorder $\mathbf{x}_u^{\text{out1}}(n)$ and $\mathbf{x}'^{\text{out2}}_u(n)$, obtain $\mathbf{x}_{u+1}(n)$ and $\mathbf{x}_{u+1}\left(n-\ell(u+1)\right)$;
    call subfilter ($\mathbf{x}_{u+1}\left(n-\ell(u+1)\right)$, $u$+1);
    call subfilter ($\mathbf{x}_{u+1}(n)$, $u$+1);
end of function.
```

4.5.4 Sample test results

We wrote a Fortran program to compare the performances of the FFB using the normal method and our proposed matrix formulation. For the normal method, we simply convolved the input signal with each individual subfilter, and repeated the process for each subfilter in the FFB. The computational platform used was an Intel Pentium 4-based 2.4 GHz workstation. For a 16-channel FFB sample run with 10K input data samples, we took approximately 0.0164s for the normal method, and approximately 0.0139s using the matrix method. For a 256-channel FFB sample run with 10K input data samples, the normal method took about 0.28s, and the matrix method took about 0.1s.

4.6 Summary

We proposed 2 methods of reducing the implementation complexity of the FFB. The node-modulation method applies a complex-valued modulating signal at various nodes in the FFB. As a result, the subfilters which had complex-valued coefficients are reduced to having real-valued coefficients. Furthermore, for systems with real-valued input signals, certain redundant signal paths in the FFB can be pruned. For an even-stacked filter bank, this can be achieved by removing every fourth subfilter and all subsequent subfilters connected to it for each stage. For an odd-stacked filter bank, the lower propagation path of the shaping subfilter with transfer function $H_{0,0}(z)$ can be conveniently removed. We designed a 16-channel filter bank, and our 2 methods reduced the overall complexity to about half.

We also proposed a matrix representation of the FFB. By representing the data operations in terms of matrices and vectors, an efficient filtering scheme was implemented. The resultant data buffers for processing the matrices and vectors are regularly-sized, with equal numbers of rows. By further making use of easily available mathematical packages such as BLAS, which are highly optimized for specific target architectures, the required computation time was decreased by up to a factor of 3 when compared to the normal method of evaluating the convolution result for each subfilter.

Chapter 5

Multi-rate Systems I: Frequency Response Masking Filter Banks

In this chapter, we focus on the multi-rate operation of odd-stacked filter banks for the real-valued input signal scenario (such as in typical audio applications). The number of subbands for the real-valued input signal, odd-stacked filter bank is given by $M = 2^{K-1}$, where K is the number of cascade stages (refer to Section 4.1.2). The outputs of the analysis filter bank are decimated by a factor of L, and in most cases we assume the outputs are critically decimated, i.e. $L = M$.

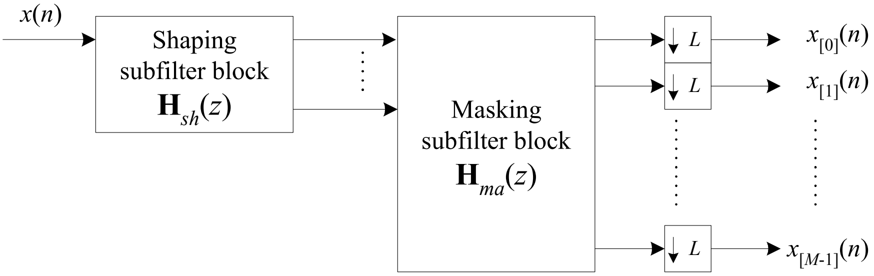


Figure 5-1 A general representation of the FRM-class filter bank for multirate operation.

Throughout this chapter, we adopt a general representation (shown in Figure 5-1) for the FRM-class of filter banks when used in multi-rate mode. This general representation comprises 2 blocks: the first block consists of the set of shaping subfilters $\mathbf{H}_{sh}(z)$ and the second block consists of the set of masking subfilters $\mathbf{H}_{ma}(z)$. In the later parts of this chapter, we divide the FRM-class of filter banks into 4 categories, based on the method of implementing the shaping and masking subfilter blocks.

5.1 Complexity considerations

5.1.1 Efficiency of FFB in multirate operations

The Fast Filter Bank is very efficient at single-rate processing of signals. However, we have shown that it does not match the polyphase filter bank in multi-rate operation in terms of efficiency. The complexity of the FFB scales non-linearly with the number of subbands:

$$\Gamma_{FFB}(K) \approx \eta \sum_{u=0}^{K-1} 2^u N_{u,0}, \tag{5.1}$$

where η is some complexity scaling variable depending on the design method we use to implement the FFB, K is the number of cascade stages, and $N_{u,0}$ is the filter length of $H_{u,0}(z)$. Since the transition width of the shaping subfilter with transfer function $H_{0,0}(z)$ is approximately M times that of the polyphase prototype filter, its filter length is approximately $1/M$ times that of the comparable polyphase prototype filter. The typical filter lengths of the masking subfilters for various designs are tabulated in Table 5-1.

u	Values of $N_{u,0}$			
	δ_s = -50 dB	δ_s = -60 dB	δ_s = -80 dB	δ_s = -100 dB
1	12	16	22	29
2	5	7	11	15
3	3	4	8	11
4	3	3	6	9
5	3	3	6	9
6	3	3	5	8

(a) $\Delta_{0,0} = 0.2\pi$, $\delta_p = \delta_s$.

u	Values of $N_{u,0}$			
	δ_s = -50 dB	δ_s = -60 dB	δ_s = -80 dB	δ_s = -100 dB
1	10	12	15	18
2	4	5	6	8
3	3	3	4	5
4	3	3	3	4
5	3	3	3	3
6	3	3	3	3

(b) $\Delta_{0,0} = 0.2\pi$, $\delta_p = 0.086$ dB.

Table 5-1 Lengths of subfilters with transfer functions $H_{u,0}(z)$ for various design conditions.

Previously, we compared the complexity of the polyphase filter bank and the FFB for single-rate processing. Let us now compare their complexities for multi-rate processing. From Appendix B, the pruned odd-stacked FFB with linear-phase subfilters has an approximate complexity:

$$\Gamma_{FFB}(K) \approx N_{0,0}/2 + 4\left(\sum_{u=1}^{K-2} 2^{u-1} \lfloor N_{u,0}/2 \rfloor\right) + 2.2^{K-2} \lfloor N_{K-1,0}/2 \rfloor. \tag{5.2}$$

The polyphase filter bank has an approximate complexity:

$$\Gamma_{PFB}(M,L) \approx \frac{MN_{0,0} + 0.5M\log_2 M}{L}, \tag{5.3}$$

where L is the decimation factor.

For single-rate operation, the complexity of the FFB is lower than that of the polyphase filter bank. However, as L increases, the complexity of the polyphase filter bank decreases proportionally to L, whereas the complexity of the FFB does not scale. We plot the value of L/M at which the complexity of the FFB is equal to the complexity of the polyphase filter bank, for various transition widths $\Delta_{0,0}$, in Figure 5-2. The peak passband ripple δ_p and the peak stopband ripple δ_s were chosen to be 0.086 dB and -80 dB respectively in this comparison.

Note: When $L/M < 1$, the polyphase filter bank is more efficient than the FFB when the outputs of both are critically decimated.

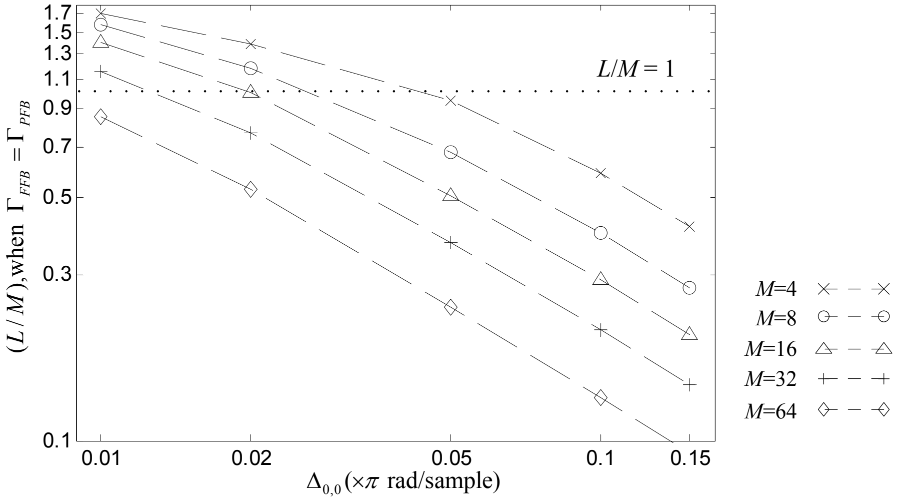


Figure 5-2 Plot of decimation factor ratio (L/M) vs $\Delta_{0,0}$, when $\Gamma_{FFB} = \Gamma_{PFB}$.

We observe that the FFB has a comparatively high complexity when the transition width $\Delta_{0,0}$ is large, or when the number of subbands M is large. At very small transition widths, the complexity of the FFB becomes comparable to that of the polyphase filter bank in a critically decimated system.

5.1.2 Simple method of complexity reduction

Although the complexity of the FFB does not scale with the decimation factor, some savings in complexity can be made when the outputs are decimated. First, we consider the last-stage subfilters with transfer functions $H_{K\text{-}1,v}(z)$, which have filter lengths of $N_{K\text{-}1,0}$. When the outputs are decimated by L, the filtering operations need only to be evaluated once every L samples, and the complexities of these subfilters scale downwards by a factor of L.

If $N_{K-1,0} \geq L$, then no further savings can be obtained from the subfilters with transfer functions $H_{K-2,v}(z)$. The complexity of the FFB becomes approximately:

$$\Gamma_{FFB}(K) \approx \eta \sum_{u=0}^{K-2} 2^u N_{u,0} + \eta 2^{K-1} \frac{N_{K-1,0}}{L}, \text{ if } N_{K-1,0} \geq L. \tag{5.4}$$

If $N_{K-1,0} < L$, then we require only $N_{K-1,0}$ input samples into the subfilters with transfer functions $H_{K-1,v}(z)$ per L output samples. Thus, only $N_{K-1,0}$ samples per L output samples need to be evaluated by the subfilters with transfer functions $H_{K-2,v}(z)$. The complexity of the FFB is further reduced to approximately:

$$\Gamma_{FFB}(K) \approx \eta \sum_{u=0}^{K-3} 2^u \Gamma_{H_{u,0}} + \eta 2^{K-2} \frac{N_{K-1,0} \Gamma_{H_{K-2,0}}}{L} + \eta 2^{K-1} \frac{\Gamma_{H_{K-1,0}}}{L},$$

$$\text{if } N_{K-1,0} < L. \tag{5.5}$$

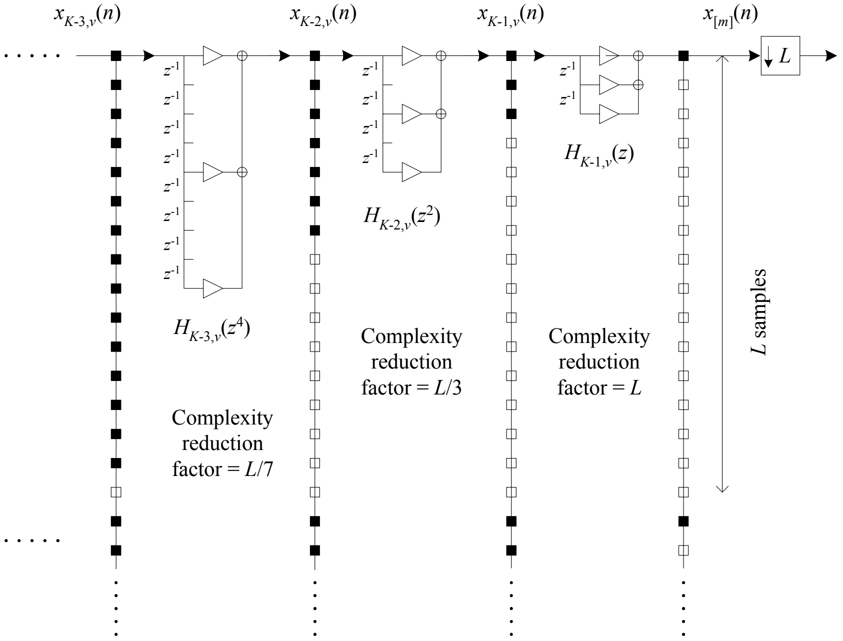


Figure 5-3 Illustration of subfilter complexity reduction when the outputs of the FFB are decimated.

Similarly, savings can be made on the subfilters with transfer functions $H_{K-3,v}(z)$, if $N_{K-1,0} + 2N_{K-2,0} - 2 < L$, and so on. This scheme is illustrated in Figure 5-3, where the black squares represent samples of $x_{u,v}(n)$ that need to be evaluated, and the white squares are samples that do not need to be evaluated, when the output $x_{[m]}(n)$ is decimated by a factor of L. For stage u, the complexity is reduced by a factor of:

$$L, \qquad \text{for } u=K\text{-}1, \tag{5.6}$$

and:

$$L \Big/ \left[1 + \sum_{u'=u+1}^{K-1} 2^{K-1-u'} (N_{u',0} - 1) \right], \qquad \text{for } u<K\text{-}1. \tag{5.7}$$

If the complexity reduction factor is evaluated to be less than or equal to one, then no savings can be attained for that stage.

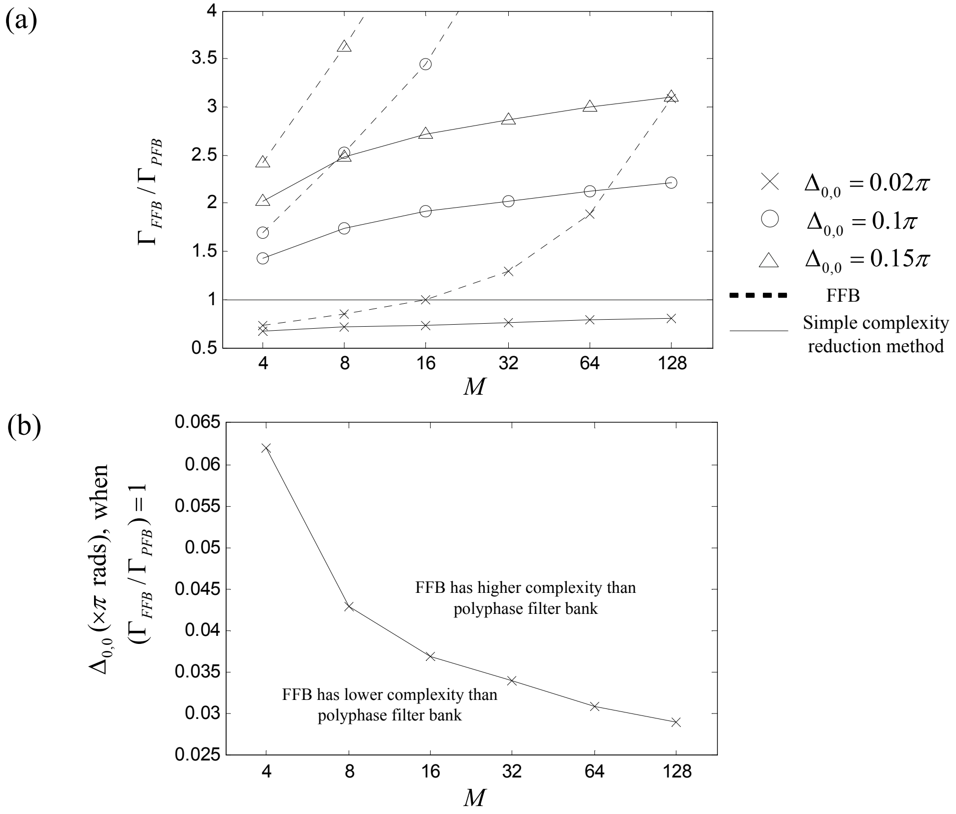


Figure 5-4 (a) Comparison between the complexity of the FFB and the proposed simple complexity reduction method; (b) Plot of the values of the transition width $\Delta_{0,0}$, at which the FFB (with simple complexity reduction) has equal complexity to the polyphase filter bank, when the outputs are critically decimated.

In Figure 5-4(a), we compare the complexities of the FFB with and without the simple complexity reduction method. We used the ratio of the complexity of the FFB to the complexity of the polyphase filter bank, $\Gamma_{FFB}/\Gamma_{PFB}$, when the filter bank outputs are critically decimated (L=M) as the unit of measurement. The peak passband ripple δ_p and the peak stopband ripple δ_s were chosen to be 0.086 dB and -80 dB respectively in this comparison. It can be seen that the proposed method becomes very effective as the number of subbands M increases.

In Figure 5-4(b), we compare the complexity of the FFB (with simple complexity reduction) to the polyphase filter bank, when the outputs are critically decimated. The plot shows values of $\Delta_{0,0}$, at which the FFB (with simple complexity reduction) has equal complexity to the polyphase filter bank. For transition widths that are smaller than this value, the FFB (with simple complexity reduction) has a lower complexity than the polyphase filter bank.

5.2 Different types of FRM-class filter banks

We defined the FRM-class of filter banks as comprising a shaping subfilter block which processes an input signal into output signals with multiple passbands, and a masking subfilter block which subsequently reduces these multiple passband signals into output signals with single passbands. The scheme was illustrated in Figure 5-1. We proceed to classify the FRM-class of filter banks into 4 Types according to the composition of the masking and shaping subfilter blocks. The FFB that was described in Chapters 2 and 3 is then classified as Type *F-I*. Using this classification scheme, we develop variants of the FFB (namely Type *F-II* and Type *F-III*) and we further propose implementation methodologies that lead to filter banks with reduced complexity.

5.2.1 Type F-I: Fast Filter Bank

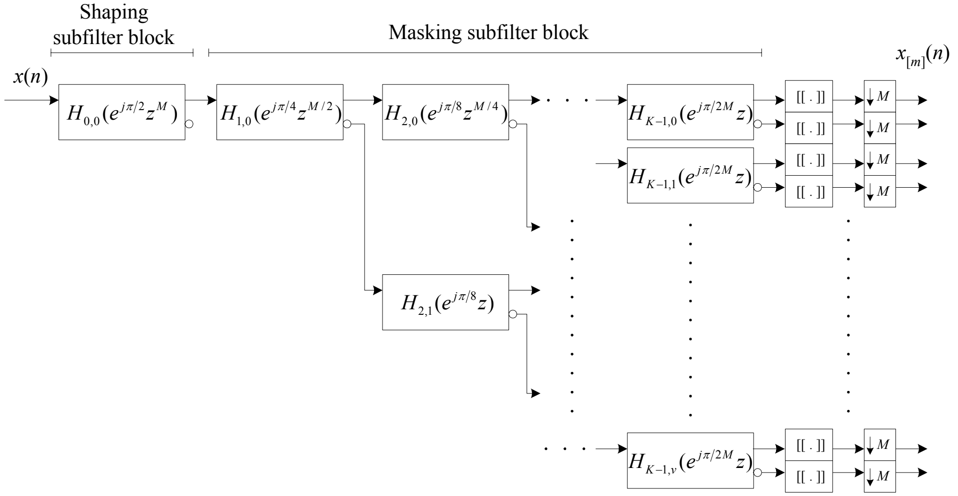

Shaping subfilter block
Masking subfilter block
$x(n)$
$x_{[m]}(n)$
$H_{0,0}(e^{j\pi/2}z^{M})$
$H_{1,0}(e^{j\pi/4}z^{M/2})$
$H_{2,0}(e^{j\pi/8}z^{M/4})$
$H_{2,1}(e^{j\pi/8}z)$
$H_{K-1,0}(e^{j\pi/2M}z)$
$H_{K-1,1}(e^{j\pi/2M}z)$
$H_{K-1,v}(e^{j\pi/2M}z)$
[[.]]
$\downarrow M$

Figure 5-5 *M*-channel odd-stacked Type *F-I* FFB.

The Type *F-I* filter bank has the following properties: (i) $\mathbf{H}_{sh}(z)$ is single-stage, and (ii) $\mathbf{H}_{ma}(z)$ is multi-stage. The FFB described in the previous chapter is a Type *F-I* filter bank. The shaping subfilter block is given by $\mathbf{H}_{sh}(z) = H_{0,0}(e^{j\pi/2}z^M)$, where $H_{0,0}(z)$ describes a lowpass filter with passband edge at approximately $\pi/2$. The masking subfilter block $\mathbf{H}_{ma}(z)$ is a branching network of subfilters given by $H_{u,v}(z^{M/2^u})$, for $1 \le u \le K-1$ and $0 \le v \le 2^{u-1}-1$. Figure 5-5 shows the implementation of the odd-stacked, pruned M-channel Type *F-I* filter bank. The [[.]] operator denotes taking the real part of the bracketed term.

5.2.2 Type F-II: Single-stage shaping subfilter, single-stage masking subfilter

The Type *F-II* filter bank has the following properties: (i) $\mathbf{H}_{sh}(z)$ is single-stage, and (ii) $\mathbf{H}_{ma}(z)$ is single-stage. The transfer function of the single-stage shaping subfilter block is given by $\mathbf{H}_{sh}(z) = H_{0,0}(e^{j\pi/2}z^M)$, where $H_{0,0}(z)$ describes a lowpass filter with passband edge at approximately $\pi/2$. The masking subfilter block $\mathbf{H}_{ma}(z)$ is single-stage and may thus be implemented using a polyphase filter bank with the prototype filter having a transfer function $H_{1,0}(e^{j\pi/2M}z)$. Due to the single-stage implementation of the masking subfilter block, the complexity of $\mathbf{H}_{ma}(z)$ can be scaled downwards with the decimation factor M.

Some multi-stage filter bank implementations such as in [40]-[42] fall into this category. In these papers, the front-end comprises a shaping subfilter and the back-end comprises a polyphase masking subfilter. The sampling rates of the shaping and masking subfilters were modified, typically by using non-integer decimation factors within the blocks, to achieve an overall decreased complexity. In this thesis, we approach the implementation of the odd-stacked Type *F-II* filter bank differently, by using subfilters with complex-valued coefficients, and by using the structure shown in Figure 5-6.

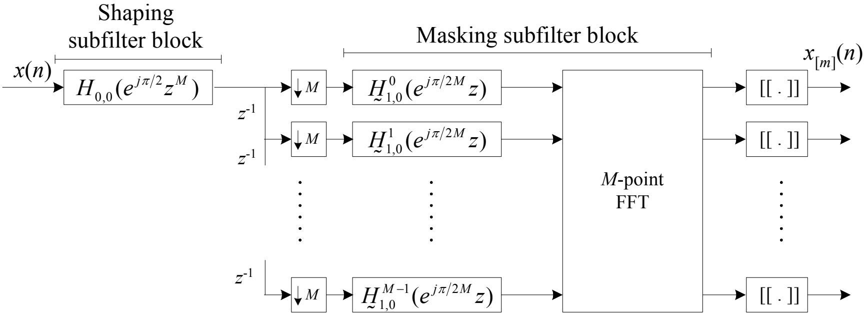


Figure 5-6 Type *F-II* FRM structure, with a polyphase masking subfilter block.

The transition width of the shaping subfilter with transfer function $H_{0,0}(z)$ is given by:

$$\Delta_{0,0} = M\Delta, \tag{5.8}$$

where Δ is the transition width of each subband of the filter bank. The transition width of the masking subfilter with transfer function $H_{1,0}(z)$ is given by:

$$\Delta_{1,0} = \left(\pi / M\right) - \Delta\,. \tag{5.9}$$

The Type *F-II* implementation is useful for a filter bank with many subbands, as the complexity of the Type *F-I* implementation increases very quickly with an increasing number of subbands. The complexity of the Type *F-II* implementation Γ_{F-II} can be approximated by:

$$\Gamma_{F-II} = \Gamma_{HS} + \frac{\Gamma_{HM} + \Gamma_{FFT}(M)}{M}, \tag{5.10}$$

where the variables Γ_{HS}, Γ_{HM}, $\Gamma_{FFT}(M)$ represent the complexity of the shaping subfilter, prototype masking subfilter and *M*-point FFT respectively.

5.2.3 *Type F-III: Multi-stage shaping subfilter, single-stage masking subfilter*

The Type *F-III* filter bank has the following properties: (i) $\mathbf{H}_{sh}(z)$ is multi-stage, and (ii) $\mathbf{H}_{ma}(z)$ is single-stage. Figure 5-7 illustrates the case when $\mathbf{H}_{sh}(z)$ is 2-stage, and $\mathbf{H}_{ma}(z)$ is implemented using the polyphase structure.

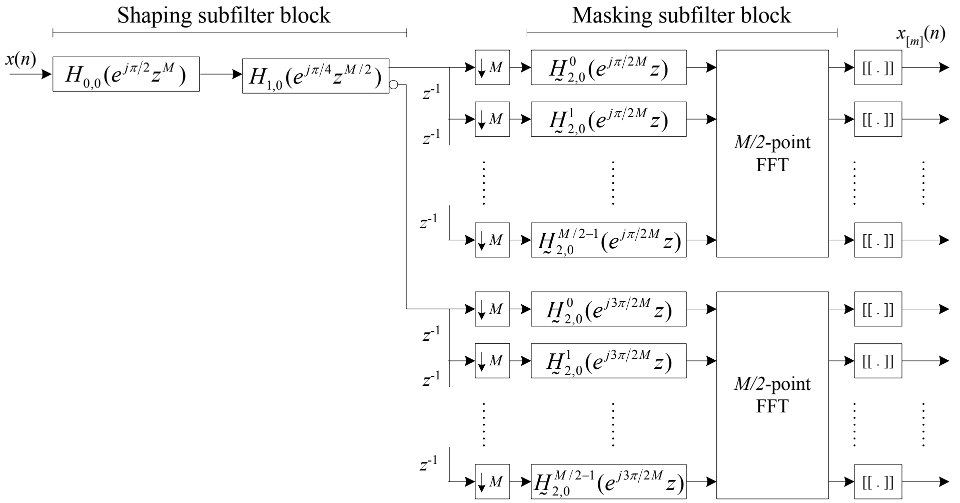

Shaping subfilter block
Masking subfilter block
$x_{[m]}(n)$
$x(n)$
$H_{0,0}(e^{j\pi/2}z^M)$
$H_{1,0}(e^{j\pi/4}z^{M/2})$
z^{-1}
$\downarrow M$
$\underset{\sim}{H}{}^{0}_{2,0}(e^{j\pi/2M}z)$
$\underset{\sim}{H}{}^{1}_{2,0}(e^{j\pi/2M}z)$
$\underset{\sim}{H}{}^{M/2-1}_{2,0}(e^{j\pi/2M}z)$
M/2-point FFT
[[.]]
$\underset{\sim}{H}{}^{0}_{2,0}(e^{j3\pi/2M}z)$
$\underset{\sim}{H}{}^{1}_{2,0}(e^{j3\pi/2M}z)$
$\underset{\sim}{H}{}^{M/2-1}_{2,0}(e^{j3\pi/2M}z)$
M/2-point FFT

Figure 5-7 Type *F-III* FRM structure, with polyphase masking subfilter blocks.

For a Type *F-III* filter bank with K_S stages in the shaping subfilter block, $\mathbf{H}_{sh}(z)$ is an odd-stacked Type *F-I* filter bank with K_S stages and $M_S = 2^{K_S-1}$ outputs. In our implementation of the masking subfilter block, we use M_S sets of masking subfilters in parallel. Each masking subfilter can be implemented using the polyphase filter bank, comprising a prototype filter and an (M/M_S)-point FFT. The prototype filters have transfer functions $H_{M_S,v}(z)$ for $0 \le v \le M_S - 1$, and are given by:

$$H_{M_S,v}(z) = H_{M_S,0}(e^{j\pi(1+\tilde{v})/2M} z), \tag{5.11}$$

where $\tilde{v}$ is the bit-reversed version of v in M_S bits. The transition width of the masking subfilter with transfer function $H_{M_S,0}(z)$ is:

$$\Delta_{M_S,0} = \left[\pi(2M_S - 1)/M\right] - \Delta . \tag{5.12}$$

The shaping subfilter block is operated in single-rate mode and the masking subfilter block is operated in multi-rate mode. When we increase the number of stages K_S for the shaping subfilter block, its complexity increases. However, the filter lengths and complexity of the masking subfilter block $\mathbf{H}_{ma}(z)$ decrease. For the case when the shaping subfilter block comprises a 2-stage cascade, the complexity is:

$$\Gamma_{F-III} = \Gamma_{F-I}(M_S) + M_S\left[\Gamma_{HM} + \Gamma_{FFT}(M/M_S)\right]/M , \tag{5.13}$$

where $\Gamma_{F-I}(M_S)$ is the complexity of a Type *F-I* filter bank with M_S outputs, Γ_{HM} is the complexity of the prototype masking subfilter with transfer function $H_{M_S,0}(z)$ and $\Gamma_{FFT}(M/M_S)$ is the complexity of the (M/M_S)-point FFT.

5.2.4 Type F-IV: Multi-stage shaping subfilter, multi-stage masking subfilter

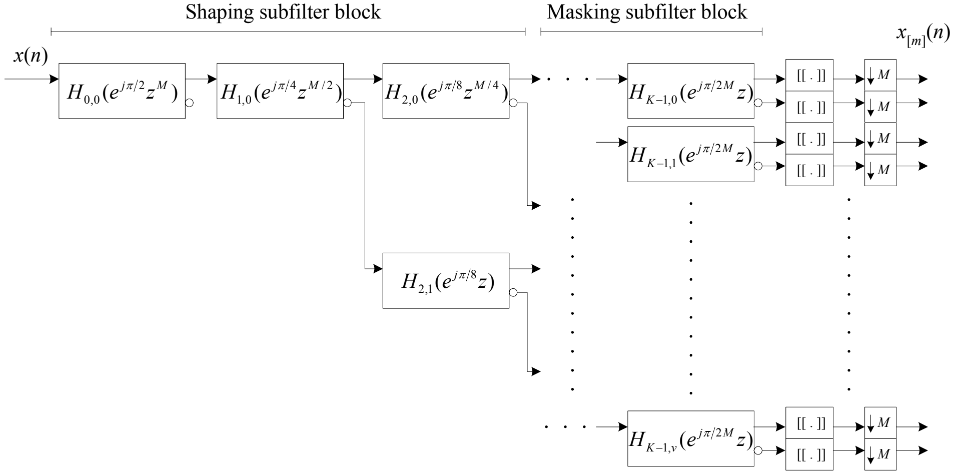

Shaping subfilter block
Masking subfilter block
$x(n)$
$x_{[m]}(n)$
$H_{0,0}(e^{j\pi/2}z^M)$
$H_{1,0}(e^{j\pi/4}z^{M/2})$
$H_{2,0}(e^{j\pi/8}z^{M/4})$
$H_{2,1}(e^{j\pi/8}z)$
$H_{K-1,0}(e^{j\pi/2M}z)$
$H_{K-1,1}(e^{j\pi/2M}z)$
$H_{K-1,v}(e^{j\pi/2M}z)$
[[.]]
$\downarrow M$

Figure 5-8 Type *F-IV* FFB structure.

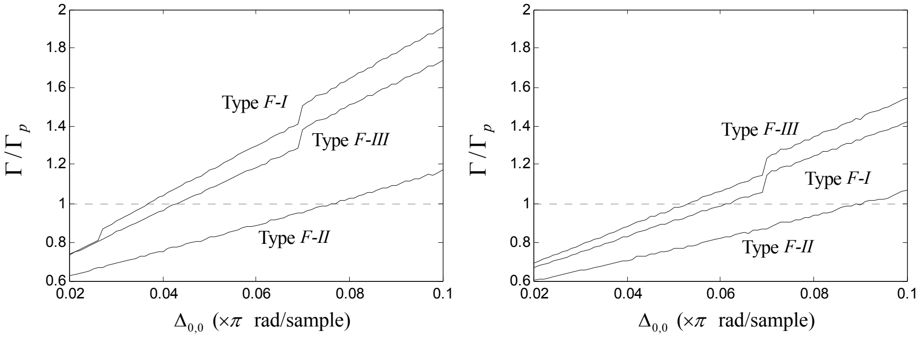
For completeness, we include the description for Type *F-IV*. Consider the Type *F-I* filter bank in Figure 5-5, and the Type *F-IV* filter bank in Figure 5-8. Although the grouping of the set of subfilters that constitute the shaping subfilter block and the masking subfilter block are different, the subfilters are structurally identical. In this thesis, we assume that our results (in terms of complexity and design) for the Type *F-I* structure can also be applied to the Type *F-IV* structure.

5.2.5 Comparisons of the various Types

Let us compare the complexities of the different types of FRM-class filter bank structures. In Table 5-2, we tabulate the filter length $N_{\max}$ of the longest component subfilter, and the complexity of the FRM-class filter bank as a ratio to the complexity of the polyphase filter bank, Γ / Γ_P. We consider different values of transition widths $\Delta_{0,0}$ and different numbers of subbands *M*. We assume a 2-stage shaping subfilter block for the Type *F-III* comparison. The peak passband and stopband ripples are 0.086 dB and -80 dB respectively.

	Polyphase	Type *F-I*		Type *F-II*		Type *F-III*	
$\Delta_{0,0}$, M	$N_{\max}$	$N_{\max}$	Γ / Γ_P	$N_{\max}$	Γ / Γ_P	$N_{\max}$	Γ / Γ_P
0.02π, 4	629	158	0.67	158	0.60	158	0.69
0.02π, 16	2512	158	0.74	158	0.63	158	0.73
0.02π, 32	5023	158	0.76	210	0.64	158	0.75
0.02π, 128	20087	158	0.81	837	0.66	272	0.77
0.1π, 4	127	32	1.42	32	1.07	32	1.54
0.1π, 16	504	32	1.91	126	1.17	35	1.74
0.1π, 32	1006	32	2.03	251	1.21	72	1.78
0.1π, 128	4019	32	2.22	1005	1.30	287	1.85

Table 5-2 Complexity comparisons of the various filter bank implementations.



(a) $M = 4$ (b) $M = 16$

Figure 5-9 Plots of complexity of different types of FRM filter banks, relative to the complexity of the polyphase filter bank, for *M*=4 and *M*=16.

In Figure 5-9, we plot the complexity ratio Γ/Γ_P for transition widths of $0.02\pi \le \Delta_{0,0} \le 0.1\pi$, for the case when M=4 and M=16. For values of $\Delta_{0,0}$ such that $\Gamma/\Gamma_P < 1$, the FRM-class filter bank has a lower complexity than the polyphase filter bank; and vice versa for $\Gamma/\Gamma_P > 1$.

The plots exhibit a decreasing complexity ratio Γ/Γ_P as the transition width decreases. It can be observed that there are minor discontinuities in the plots for the Type *F-I* and Type *F-III* filter banks, e.g. in the region $0.069\pi \le \Delta_{0,0} \le 0.07\pi$ for M=16. This is further explained at the end of this sub-section.

In general, the different filter banks exhibit an $N_{\max}$ and complexity trade-off. The polyphase filter bank has large values of $N_{\max}$, especially for a large value of M and a small value of $\Delta_{0,0}$. For the FRM-class filter bank, the individual filter lengths of its subfilters are much smaller than that of a polyphase filter bank having the same specifications. The Type *F-I* structure, which has the largest number of stages, also has the smallest value of N_{max}.

For very small transition widths (such as $\Delta_{0,0} < 0.03\pi$), the FRM-class filter banks are more efficient. For values of $\Delta_{0,0}$ that are approximately 0.03π to 0.08π, or larger, the polyphase filter bank implementation generally has a lower complexity than the FRM-class filter banks, as the polyphase structure can be fully operated in multi-rate mode. The FRM-class filter banks on the other hand, can only be partially operated in multi-rate mode and tend to have a higher complexity. We summarize the properties of the different filter banks in Table 5-3.

	Polyphase	Type *F-I*	Type *F-II* and *F-III*
Complexity	Generally lowest complexity.	Generally highest complexity. At very small transition widths, has a lower complexity than the polyphase filter bank.	Generally falls between the polyphase filter bank and the *Type F-I*.
Filter lengths	Prototype filter have very large filter lengths.	Subfilters have very small filter lengths.	Filter lengths vary according to number of stages K_S.

Table 5-3 Summary of the characteristics of the various FRM-class filter banks.

Note that although the FRM-class of filter banks generally exhibit a higher complexity than the polyphase filter bank, their complexity can be further reduced for hardware implementations, such as by using the signed-powers-of-two method. The FRM-class of filter banks are also efficient for implementing non-uniform filter banks with small transition widths. These will be discussed in Chapter 6.

Explanation of discontinuities observed in Figure 5-9:

As $\Delta_{0,0}$ decreases, Γ/Γ_P follows a decreasing trend. Furthermore, as $\Delta_{0,0}$ decreases, the requirement on $\Delta_{1,0}$ increases (i.e. becomes more relaxed). Hence, we expect the filter length of $H_{1,0}(z)$, and hence the complexity of $H_{1,0}(z)$, to decrease as $\Delta_{0,0}$ decreases.

The discontinuities in the plots for the Type *F-I* and *Type F-III* can be explained by examining their structures (Figure 5-5 and Figure 5-7). In both cases, $H_{0,0}(z)$ is interpolated by a factor of M and $H_{1,0}(z)$ is interpolated by a factor of $M/2$. The filter length of $H_{1,0}(z)$ is typically small (e.g. typically between the range of 15 to 25), and can only take on integer values. As a result, the filter length of $H_{1,0}(z)$ changes only at some values of $\Delta_{0,0}$, which becomes visible as a discontinuity in the plots. For example, as $\Delta_{0,0}$ decreases from 0.07π to 0.069π, the filter length of $H_{1,0}(z)$ decreases by one, and is visible as a discontinuity. As $\Delta_{0,0}$ further decreases from 0.069π to 0.027π, the estimated filter length of $H_{1,0}(z)$ does not change. The effect of different $\Delta_{0,0}$ on the filter lengths of the subfilters comprising the FFB was discussed at length in Section 3.3.3.

For the Type *F-II* filter bank (Figure 5-6) on the other hand, the change in Γ/Γ_P is somewhat more gradual. In this case, $H_{1,0}(z)$ is not interpolated and its filter length is large (e.g. typically between the range of 150 to 250 for M=16) and changes gradually with $\Delta_{0,0}$.

5.3 Alternate multi-rate structures

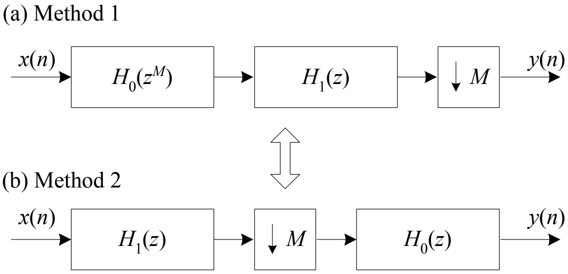


Figure 5-10 Equivalent multirate structure for cascaded filters.

Figure 5-10(a) shows a multirate system in which the output of the cascaded filters $H_0(z^M)$ and $H_1(z)$ is decimated by a factor of M. This system can be equivalently implemented as shown in Figure 5-10(b), in which the order of $H_0(z)$ and $H_1(z)$ has been reversed. The output of $H_1(z)$ is decimated by M, and is then input to $H_0(z)$. Let us denote the complexity of $H_0(z)$ and $H_1(z)$ as Γ_0 and Γ_1 respectively.

In the first method, since $H_0(z^M)$ has the same number of non-zero coefficients as $H_0(z)$, its complexity is Γ_0. Since the output of $H_1(z)$ is decimated by M, and only 1 output sample needs to be evaluated per M input samples, i.e. its complexity is Γ_1 / M. Thus, the complexity of method 1 is $\Gamma_0 + \Gamma_1 / M$. In the second method, the filters are arranged differently. Since the output of $H_1(z)$ is decimated by M, and only 1 output sample needs to be evaluated per M input samples, i.e. its complexity is Γ_1 / M. Since the signal entering $H_0(z)$ is already decimated by M, the complexity of the filter operation involving $H_0(z)$ is Γ_0 / M. Thus, the complexity of method 2 is $(\Gamma_0 + \Gamma_1) / M$.

The second method offers a lower complexity. Note that methods 1 and 2 are interchangeable only if the front-end filter in method 1 is interpolated by a factor of M that is equal to the output decimation factor M.

We can apply the same logic to the FRM filter banks. Figure 5-11 shows the equivalent structure for the Type *F-I* filter bank, and Figure 5-12 shows the equivalent structure for the Type *F-II* filter bank.

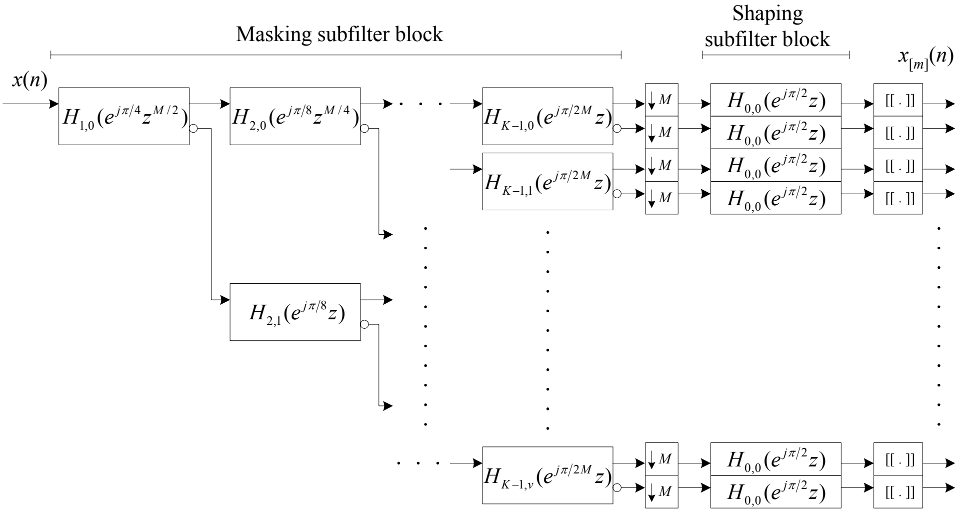

Masking subfilter block
Shaping subfilter block
$x(n)$
$x_{[m]}(n)$
$H_{1,0}(e^{j\pi/4}z^{M/2})$
$H_{2,0}(e^{j\pi/8}z^{M/4})$
$H_{K-1,0}(e^{j\pi/2M}z)$
$H_{K-1,1}(e^{j\pi/2M}z)$
$H_{2,1}(e^{j\pi/8}z)$
$H_{K-1,v}(e^{j\pi/2M}z)$
$\downarrow M$
$H_{0,0}(e^{j\pi/2}z)$
[[.]]

Figure 5-11 Equivalent multirate structure for Type *F-I* filter bank.

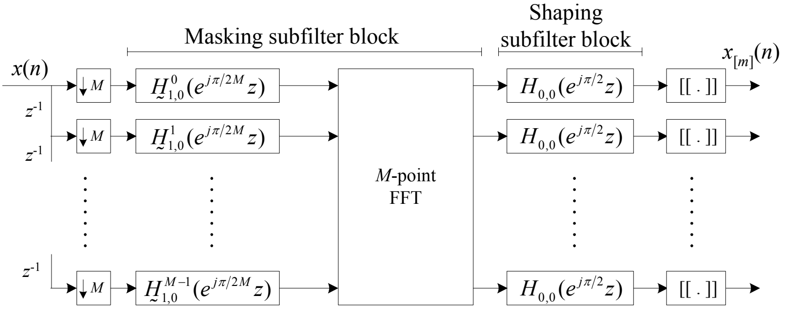

Masking subfilter block
Shaping subfilter block
$x(n)$
$x_{[m]}(n)$
z^{-1}
$\downarrow M$
$\underset{\sim}{H}^{0}_{1,0}(e^{j\pi/2M}z)$
$\underset{\sim}{H}^{1}_{1,0}(e^{j\pi/2M}z)$
$\underset{\sim}{H}^{M-1}_{1,0}(e^{j\pi/2M}z)$
M-point FFT
$H_{0,0}(e^{j\pi/2}z)$
[[.]]

Figure 5-12 Equivalent multirate structure for Type *F-II* filter bank.

Note that even though the number of subfilter blocks for $H_{0,0}(e^{j\pi/2}z)$ has increased by a factor of *M*, the sampling rate of these subfilter blocks has decreased by a factor of *M*. Therefore, the overall complexity of the filter bank is unchanged.

5.3.1 Application: Frequency-hopping system using the Type F-II implementation

The alternate equivalent multirate structure can be applied efficiently in a general frequency-hopping system such as in Figure 5-13. In this implementation, the outputs of the masking subfilter are passed through a detector. Due to the wider transition width of the masking subfilter block compared to the shaping subfilter block, the signal entering the detector has an increased overlap with adjacent subbands. Since signal detection involves rough approximations (unlike signal reconstruction which requires accurate approximations), the increased overlap may be acceptable.

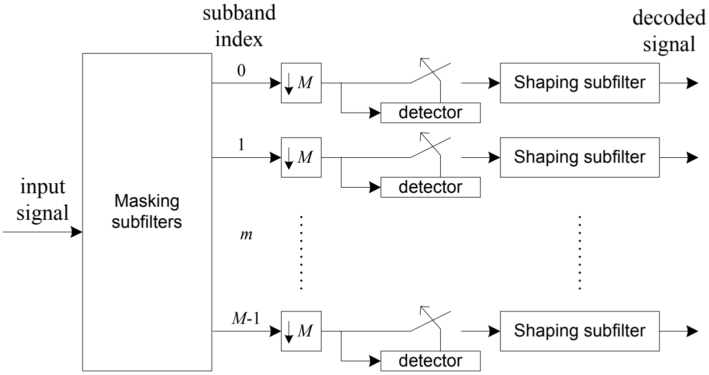


Figure 5-13 Applying the equivalent multirate structure efficiently in a general frequency-hopping scenario.

When a signal is detected, we pass the signal into the shaping subfilter to refine the transition edge for reconstruction purposes. When no signal is detected, the operations required by the shaping subfilter can be saved. For a system with low subband utilization, the savings achieved by this method can be significant.

5.4 Summary

The operation of the Fast Filter Bank in multi-rate systems is considered. When the system is critically decimated, the FFB typically has a much higher complexity than the polyphase filter bank. We proposed a method of reducing the FFB complexity by removing redundant computations required by the subfilters at the output stage. The improvements can be very significant when the number of subbands is large. For some implementations, such as designs with very small transition widths, our improved FFB designs can be even more efficient than the polyphase filter bank.

We further proposed a classification of filter banks that are based on frequency response masking. We labeled these classes of filter banks Type *F-I*, Type *F-II*, Type *F-III* and Type *F-IV*. The original FFB belongs to the Type *F-I* class. The Type *F-II* and Type *F-III* classes lead to implementations that have lower complexity than the FFB. The trade-offs between the different types lie primarily between the subfilter lengths and the overall complexity.

We then considered alternate multi-rate structures for FRM filter banks. These alternate structures can be applied in situations where the entire bandwidth is not fully utilized, such as in a frequency hopping scenario. As a case in point, we briefly described a scheme where the alternate structure was applied to decrease the overall complexity.

Chapter 6

Multi-rate Systems II: Reconstruction Analysis and Multiplierless Designs

In this chapter, we consider practical aspects of efficient filter bank implementation using our FRM filter bank structures. In the first part of the chapter, we derive the distortion and aliasing functions as a result of cascading a pair of analysis-synthesis FRM filter banks. We consider the odd-stacked filter bank with a real-valued input signal. We find that the masking subfilters have only a small impact on the distortion and aliasing functions as well as on the overall frequency response of the filter bank, as long as they are effective masking subfilters. We thus propose to use signed-powers-of-two (SPT) quantization on the masking subfilter coefficients. A variable bandwidth filter bank is then designed and optimized using the proposed techniques in this thesis.

6.1 Reconstruction analysis for 4 channels

We begin the analysis of signal reconstruction in a cascaded analysis-synthesis FRM filter bank pair with the general-case 4-channel filter bank, and extend it to the case of the *M*-channel filter bank later. The design of the analysis and synthesis filter banks for the NPR scenario is considered. Although recent literature [8] has focused on the design of PR multi-rate filter banks, our situation is different from those described. The FRM filter bank is a multi-stage cascade of subfilters, and the transfer function of each subband consists of the transfer functions of multiple subfilters. At this point, we are uncertain of the requirements on these subfilters, such that their interaction results in the PR property, if it is at all possible. Note that simply making each set of $H_{u,v}(z)$ and $G_{u,v}(z)$ a 2-channel PR pair, as in the octave filter bank, does not result in the overall **H** and **G** being a PR pair. Furthermore, our approach to the efficient design of the masking subfilters involves the reduction of their coefficients to signed-powers-of-two terms (in Section 6.3), which would also result in the PR property being infeasible.

Due to the multiple interactions of the subfilter transfer functions, the reconstruction equations become overly-complex. Fortunately, as we shall see (in arriving at (6.8)), some assumptions can be made to simplify these equations. This is also the reason why the 4-channel case was chosen as illustration. The condition for these assumptions to be made is not visible in the 2-channel case, i.e. in the 2-channel case, all subbands are adjacent.

6.1.1 Reconstruction equations

The 4-channel FRM filter bank pair is shown in Figure 6-1.

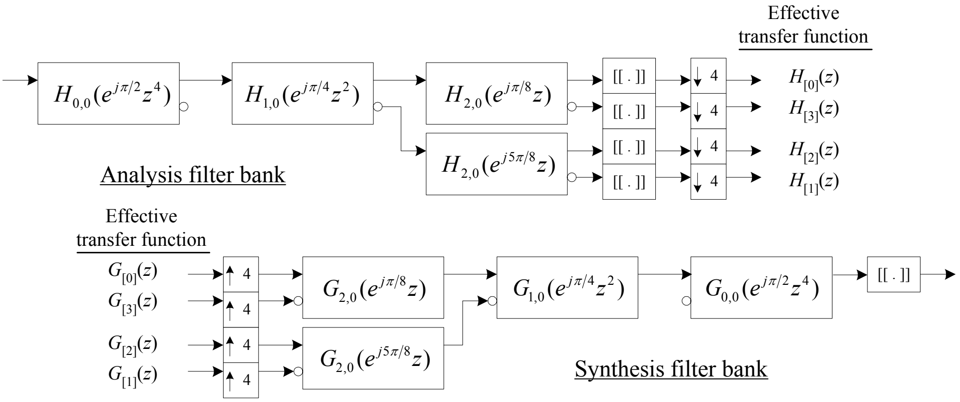


Figure 6-1 A pruned odd-stacked FFB for a real-valued input signal.

The transfer functions for the analysis part are given by:

$$\begin{aligned}
H_{[0]}(z) &= \llbracket H_{0,0}(e^{j\pi/2}z^4)H_{1,0}(e^{j\pi/4}z^2)H_{2,0}(e^{j\pi/8}z) \rrbracket, \\
H_{[1]}(z) &= \llbracket H_{0,0}(e^{j\pi/2}z^4)H_{1,0}(-e^{j\pi/4}z^2)H_{2,0}(-e^{j5\pi/8}z) \rrbracket, \\
H_{[2]}(z) &= \llbracket H_{0,0}(e^{j\pi/2}z^4)H_{1,0}(-e^{j\pi/4}z^2)H_{2,0}(e^{j5\pi/8}z) \rrbracket, \\
H_{[3]}(z) &= \llbracket H_{0,0}(e^{j\pi/2}z^4)H_{1,0}(e^{j\pi/4}z^2)H_{2,0}(-e^{j\pi/8}z) \rrbracket,
\end{aligned} \tag{6.1}$$

where $\llbracket\, . \,\rrbracket$ denotes taking the real part of the bracketed expression.

The transfer functions for the synthesis part are given by:

$$\begin{aligned}
G_{[0]}(z) &= \llbracket G_{0,0}(e^{j\pi/2}z^4)G_{1,0}(e^{j\pi/4}z^2)G_{2,0}(e^{j\pi/8}z) \rrbracket, \\
G_{[1]}(z) &= \llbracket G_{0,0}(e^{j\pi/2}z^4)G_{1,0}(-e^{j\pi/4}z^2)G_{2,0}(-e^{j5\pi/8}z) \rrbracket, \\
G_{[2]}(z) &= \llbracket G_{0,0}(e^{j\pi/2}z^4)G_{1,0}(-e^{j\pi/4}z^2)G_{2,0}(e^{j5\pi/8}z) \rrbracket, \\
G_{[3]}(z) &= \llbracket G_{0,0}(e^{j\pi/2}z^4)G_{1,0}(e^{j\pi/4}z^2)G_{2,0}(-e^{j\pi/8}z) \rrbracket.
\end{aligned} \tag{6.2}$$

Given the input signal $x(n)$ with z-transform $X(z)$, the output signal $\hat{x}(n)$ after passing through the critically decimated analysis-synthesis filter bank pair can be expressed using the following z-transform equation:

$$\begin{aligned}
\hat{X}(z) &= \sum_{m=0}^{3}\left\{ G_{[m]}(z)\left[\sum_{a=0}^{3} H_{[m]}(e^{j\pi a/2}z)X(e^{j\pi a/2}z)\right]\right\} \\
&= \sum_{a=0}^{3}\left\{ X(e^{j\pi a/2}z)\left[\sum_{m=0}^{3} H_{[m]}(e^{j\pi a/2}z)G_{[m]}(z)\right]\right\}.
\end{aligned} \tag{6.3}$$

The output signal with z-transform $\hat{X}(z)$ consists of a filtered version of $X(z)$, and of filtered versions of $X(e^{ja\pi/2})$ (for $1 \le a \le 3$), which are modulated 'images' of $X(z)$. The component of $\hat{X}(z)$ due to the modulated 'images' of $X(z)$ is termed the aliasing error function. The component of $\hat{X}(z)$ due to its fundamental $X(z)$ is termed the distortion error function and denoted $\Theta_T(z)$. This function can be found by setting a=0, and is given by:

$$\Theta_T(z) = X(z)\sum_{m=0}^{3} G_{[m]}(z)H_{[m]}(z)\,. \tag{6.4}$$

The aliasing error function $\Theta_A(z)$ is then given by:

$$\begin{aligned}\Theta_A(z) &= \sum_{m=0}^{3}\left\{G_{[m]}(z)\left[\sum_{a=1}^{3} H_{[m]}(e^{j\pi a/2}z)X(e^{j\pi a/2}z)\right]\right\} \\ &= \sum_{a=1}^{3}\left\{X(e^{j\pi a/2}z)\left[\sum_{m=0}^{3} H_{[m]}(e^{j\pi a/2}z)G_{[m]}(z)\right]\right\}.\end{aligned} \tag{6.5}$$

For perfect reconstruction (PR), i.e. $\hat{X}(z) = X(z)$, $\Theta_T(z) = 1$ and $\Theta_A(z) = 0$. We can express this in matrix form as:

$$\begin{bmatrix} G_{[0]}(z) \\ G_{[1]}(z) \\ G_{[2]}(z) \\ G_{[3]}(z) \end{bmatrix}^T \begin{bmatrix} H_{[0]}(z) & H_{[0]}(e^{j\pi/2}z) & H_{[0]}(e^{j2\pi/2}z) & H_{[0]}(e^{j3\pi/2}z) \\ H_{[1]}(z) & H_{[1]}(e^{j\pi/2}z) & H_{[1]}(e^{j2\pi/2}z) & H_{[1]}(e^{j3\pi/2}z) \\ H_{[2]}(z) & H_{[2]}(e^{j\pi/2}z) & H_{[2]}(e^{j2\pi/2}z) & H_{[2]}(e^{j3\pi/2}z) \\ H_{[3]}(z) & H_{[3]}(e^{j\pi/2}z) & H_{[3]}(e^{j2\pi/2}z) & H_{[3]}(e^{j3\pi/2}z) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}^T . \tag{6.6}$$

6.1.2 *Aliasing error*

Expanding from (6.5), we get:

$$\begin{aligned}\Theta_A(z)_a = &\frac{1}{4}X(e^{ja\pi/2}z)\Big\{\Big[H_{0,0}(e^{j\pi/2}z^4)H_{1,0}(e^{j(1+4a)\pi/4}z^2)H_{2,0}(e^{j(1+4a)\pi/8}z) \\ &+H_{0,0}(e^{-j\pi/2}z^4)H_{1,0}(e^{j(-1+4a)\pi/4}z^2)H_{2,0}(e^{j(-1+4a)\pi/8}z)\Big] \\ &.\Big[G_{0,0}(e^{j\pi/2}z^4)G_{1,0}(e^{j\pi/4}z^2)G_{2,0}(e^{j\pi/8}z) \\ &+G_{0,0}(e^{-j\pi/2}z^4)G_{1,0}(e^{-j\pi/4}z^2)G_{2,0}(e^{-j\pi/8}z)\Big]\Big\} \\ &+\frac{1}{4}X(e^{ja\pi/2}z)\Big\{\Big[H_{0,0}(e^{j\pi/2}z^4)H_{1,0}(e^{j(-3+4a)\pi/4}z^2)H_{2,0}(e^{j(-3+4a)\pi/8}z) \\ &+H_{0,0}(e^{-j\pi/2}z^4)H_{1,0}(e^{j(3+4a)\pi/4}z^2)H_{2,0}(e^{j(3+4a)\pi/8}z)\Big] \\ &.\Big[G_{0,0}(e^{j\pi/2}z^4)G_{1,0}(e^{-j3\pi/4}z^2)G_{2,0}(e^{-j3\pi/8}z) \\ &+G_{0,0}(e^{-j\pi/2}z^4)G_{1,0}(e^{j3\pi/4}z^2)G_{2,0}(e^{j3\pi/8}z)\Big]\Big\}\ldots\end{aligned}$$

$$
\begin{aligned}
&+\frac{1}{4}X(e^{ja\pi/2}z)\Big\{\Big[H_{0,0}(e^{j\pi/2}z^4)H_{1,0}(e^{j(-3+4a)\pi/4}z^2)H_{2,0}(e^{j(5+4a)\pi/8}z)\\
&+H_{0,0}(e^{-j\pi/2}z^4)H_{1,0}(e^{j(3+4a)\pi/4}z^2)H_{2,0}(e^{j(-5+4a)\pi/8}z)\Big]\\
&.\Big[G_{0,0}(e^{j\pi/2}z^4)G_{1,0}(e^{-j3\pi/4}z^2)G_{2,0}(e^{j5\pi/8}z)\\
&+G_{0,0}(e^{-j\pi/2}z^4)G_{1,0}(e^{j3\pi/4}z^2)G_{2,0}(e^{-j5\pi/8}z)\Big]\Big\}\\
&+\frac{1}{4}X(e^{j\pi/2}z)\Big\{\Big[H_{0,0}(e^{j\pi/2}z^4)H_{1,0}(e^{j(1+4a)\pi/4}z^2)H_{2,0}(e^{j(-7+4a)\pi/8}z)\\
&+H_{0,0}(e^{-j\pi/2}z^4)H_{1,0}(e^{j(-1+4a)\pi/4}z^2)H_{2,0}(e^{j(7+4a)\pi/8}z)\Big]\\
&.\Big[G_{0,0}(e^{j\pi/2}z^4)G_{1,0}(e^{j\pi/4}z^2)G_{2,0}(e^{-j7\pi/8}z)\\
&+G_{0,0}(e^{-j\pi/2}z^4)G_{1,0}(e^{-j\pi/4}z^2)G_{2,0}(e^{j7\pi/8}z)\Big]\Big\}. \qquad (6.7)
\end{aligned}
$$

We can simplify the above expression by assuming that products of transfer functions defining non-adjacent subbands do not have significant overlap and are approximately zero. For example, by inspection of the frequency responses, we get:

$$\left[H_{0,0}(e^{j\pi/2}z^4)H_{1,0}(e^{j5\pi/4}z^2)H_{2,0}(e^{j5\pi/8}z)\right]\left[G_{0,0}(e^{j\pi/2}z^4)G_{1,0}(e^{j\pi/4}z^2)G_{2,0}(e^{j\pi/8}z)\right]\approx 0\,. \qquad (6.8)$$

Let us further assume that the subfilters in the masking subfilter block of the analysis and the synthesis filter bank are mirror images, i.e. $G_{u,v}(z)=H_{u,v}(z)$ for u>0. Then we can simplify (6.7) for a=1 to:

$$
\begin{aligned}
\Theta_A(z)_{a=1} &\approx \frac{1}{4}X(e^{j\pi/2}z)\Big\{\\
&\Big[H_{0,0}(e^{-j\pi/2}z^4)H_{1,0}(e^{j(-1+4a)\pi/4}z^2)H_{2,0}(e^{j(-1+4a)\pi/8}z)\,G_{0,0}(e^{j\pi/2}z^4)H_{1,0}(e^{j\pi/4}z^2)H_{2,0}(e^{j\pi/8}z)\Big]\\
&+\Big[H_{0,0}(e^{j\pi/2}z^4)H_{1,0}(e^{j(-3+4a)\pi/4}z^2)H_{2,0}(e^{j(-3+4a)\pi/8}z)\,G_{0,0}(e^{-j\pi/2}z^4)H_{1,0}(e^{j3\pi/4}z^2)H_{2,0}(e^{j3\pi/8}z)\Big]\\
&+\Big[H_{0,0}(e^{j\pi/2}z^4)H_{1,0}(e^{j(-3+4a)\pi/4}z^2)H_{2,0}(e^{j5+4a)\pi/8}z)\,G_{0,0}(e^{-j\pi/2}z^4)H_{1,0}(e^{j3\pi/4}z^2)H_{2,0}(e^{-j5\pi/8}z)\Big]\\
&+\Big[H_{0,0}(e^{-j\pi/2}z^4)H_{1,0}(e^{j(-1+4a)\pi/4}z^2)H_{2,0}(e^{j(7+4a)\pi/8}z)\,G_{0,0}(e^{j\pi/2}z^4)H_{1,0}(e^{j\pi/4}z^2)H_{2,0}(e^{-j7\pi/8}z)\Big]\Big\}\\
&=\frac{1}{4}X(e^{j\pi/2}z)\Phi_A(z)_{a=1}\Big[H_{0,0}(e^{-j\pi/2}z^4)G_{0,0}(e^{j\pi/2}z^4)+H_{0,0}(e^{j\pi/2}z^4)G_{0,0}(e^{-j\pi/2}z^4)\Big], \qquad (6.9)
\end{aligned}
$$

where:

$$
\begin{aligned}
\Phi_A(z)_{a=1} &= H_{1,0}(e^{j3\pi/4}z^2)H_{2,0}(e^{j3\pi/8}z)H_{1,0}(e^{j\pi/4}z^2)H_{2,0}(e^{j\pi/8}z)\\
&+H_{1,0}(e^{-j5\pi/4}z^2)H_{2,0}(e^{-j5\pi/8}z)H_{1,0}(e^{-j7\pi/4}z^2)H_{2,0}(e^{-j7\pi/8}z)\,. \qquad (6.10)
\end{aligned}
$$

Similarly, by performing the same evaluation for a=2 and a=3, we obtain:

$$
\begin{aligned}
\Theta_A(z)_{a=2} &\approx \frac{1}{4}X(e^{ja\pi/2}z)\Phi_A(z)_{a=2}\cdots\\
&.\Big[H_{0,0}(e^{-j\pi/2}z^4)G_{0,0}(e^{j\pi/2}z^4)+H_{0,0}(e^{j\pi/2}z^4)G_{0,0}(e^{-j\pi/2}z^4)\Big], \qquad (6.11)
\end{aligned}
$$

where:

$$\begin{aligned}\Phi_A(z)_{a=2} = {} & H_{1,0}(e^{j3\pi/4}z^2)H_{2,0}(e^{j3\pi/8}z)H_{1,0}(e^{j5\pi/4}z^2)H_{2,0}(e^{j5\pi/8}z) \\ & + H_{1,0}(e^{-j3\pi/4}z^2)H_{2,0}(e^{-j3\pi/8}z)H_{1,0}(e^{-j5\pi/4}z^2)H_{2,0}(e^{-j5\pi/8}z),\end{aligned} \tag{6.12}$$

and:

$$\begin{aligned}\Theta_A(z)_{a=3} \approx {} & \frac{1}{4}X(e^{ja\pi/2}z)\Phi_A(z)_{a=3}\cdots \\ & \cdot\left[H_{0,0}(e^{-j\pi/2}z^4)G_{0,0}(e^{j\pi/2}z^4) + H_{0,0}(e^{j\pi/2}z^4)G_{0,0}(e^{-j\pi/2}z^4)\right],\end{aligned} \tag{6.13}$$

where:

$$\begin{aligned}\Phi_A(z)_{a=3} = {} & H_{1,0}(e^{-j3\pi/4}z^2)H_{2,0}(e^{-j3\pi/8}z)H_{1,0}(e^{-j\pi/4}z^2)H_{2,0}(e^{-j\pi/8}z) \\ & + H_{1,0}(e^{j5\pi/4}z^2)H_{2,0}(e^{j5\pi/8}z)H_{1,0}(e^{j7\pi/4}z^2)H_{2,0}(e^{j7\pi/8}z) \quad .\end{aligned} \tag{6.14}$$

Generalizing from (6.9)-(6.14), we obtain the aliasing error function as:

$$\begin{aligned}\Theta_A(z) \approx {} & \frac{1}{4}\left[\sum_{a=1}^{3} X(e^{ja\pi/2}z)\Phi_A(z)_a\right]\cdots \\ & \cdot\left[H_{0,0}(e^{-j\pi/2}z^4)G_{0,0}(e^{j\pi/2}z^4) + H_{0,0}(e^{j\pi/2}z^4)G_{0,0}(e^{-j\pi/2}z^4)\right],\end{aligned} \tag{6.15}$$

where:

$$\begin{aligned}\Phi_A(z)_a = {} & H_{1,0}(e^{j(2a+1)\pi/4}z^2)H_{2,0}(e^{j(2a+1)\pi/8}z)H_{1,0}(e^{j(2a-1)\pi/4}z^2)H_{2,0}(e^{j(2a-1)\pi/8}z) \\ & + H_{1,0}(e^{j(2a-9)\pi/4}z^2)H_{2,0}(e^{j(2a-9)\pi/8}z)H_{1,0}(e^{j(2a-7)\pi/4}z^2)H_{2,0}(e^{j(2a-7)\pi/8}z).\end{aligned} \tag{6.16}$$

We find that the aliasing error function is approximately zero, $\Theta_A(z) \approx 0$, when the following equation is satisfied:

$$H_{0,0}(e^{-j\pi/2}z^4)G_{0,0}(e^{j\pi/2}z^4) + H_{0,0}(e^{j\pi/2}z^4)G_{0,0}(e^{-j\pi/2}z^4) = 0\,. \tag{6.17}$$

The above equation can be satisfied when:

$$H_{0,0}(e^{-j\pi/2}z)G_{0,0}(e^{j\pi/2}z) + H_{0,0}(e^{j\pi/2}z)G_{0,0}(e^{-j\pi/2}z) = 0\,. \tag{6.18}$$

The above is a property of a non-aliasing, 2-channel analysis-synthesis filter bank pair, $H_{0,0}(z)$ and $G_{0,0}(z)$. Therefore, if we choose the shaping subfilter pair to be non-aliasing, then the filter bank is approximately non-aliasing. If the shaping subfilter pair is aliasing, then the total alias component in the filter bank can be found from (6.15).

6.1.3 Distortion error

Following the assumption that $G_{u,v}(z) = H_{u,v}(z)$ for u>0, the terms in the summation brackets of the distortion error function can be expanded and simplified to:

$$\begin{aligned}\frac{2\Theta_T(z)}{X(z)} &= G_{0,0}(e^{j\pi/2}z^4)H_{0,0}(e^{j\pi/2}z^4)\left\{H_{1,0}^2(e^{j\pi/4}z^2)\left[H_{2,0}^2(e^{j\pi/8}z)+H_{2,0}^2(e^{-j7\pi/8}z)\right]\right.\\ &\quad \left.+H_{1,0}^2(e^{-j3\pi/4}z^2)\left[H_{2,0}^2(e^{-j3\pi/8}z)+H_{2,0}^2(e^{j5\pi/8}z)\right]\right\}+\\ &\quad G_{0,0}(e^{-j\pi/2}z^4)H_{0,0}(e^{-j\pi/2}z^4)\left\{H_{1,0}^2(e^{-j\pi/4}z^2)\left[H_{2,0}^2(e^{-j\pi/8}z)+H_{2,0}^2(e^{j7\pi/8}z)\right]\right.\\ &\quad \left.+H_{1,0}^2(e^{j3\pi/4}z^2)\left[H_{2,0}^2(e^{j3\pi/8}z)+H_{2,0}^2(e^{-j5\pi/8}z)\right]\right\}.\end{aligned} \tag{6.19}$$

If the transfer function $\left[H_{u,0}(z)\right]^2$ is that of a half-band filter for $1 \le u \le K-1$ (i.e. if $\left[H_{u,0}(z)\right]^2+\left[H_{u,0}(-z)\right]^2=1$), then (6.19) reduces to:

$$\frac{2\Theta_T(z)}{X(z)} = G_{0,0}(e^{j\pi/2}z^4)H_{0,0}(e^{j\pi/2}z^4)+G_{0,0}(e^{-j\pi/2}z^4)H_{0,0}(e^{-j\pi/2}z^4). \tag{6.20}$$

Thus, the filter bank can be made distortionless if both the following conditions are met: $\left[H_{u,0}(z)\right]^2$ for u>0 are all half-band filters, and the 2-channel shaping subfilter pair is distortionless:

$$G_{0,0}(e^{j\pi/2}z)H_{0,0}(e^{j\pi/2}z)+G_{0,0}(e^{-j\pi/2}z)H_{0,0}(e^{-j\pi/2}z)=1. \tag{6.21}$$

Consider the case when $\left[H_{u,0}(z)\right]^2$ for u>0 are not half-band filters. Manipulating (6.19), this becomes:

$$\begin{aligned}\frac{2\Theta_T(z)}{X(z)} &= G_{0,0}(e^{j\pi/2}z^4)H_{0,0}(e^{j\pi/2}z^4)\Phi_T(z^4)\\ &\quad +G_{0,0}(e^{-j\pi/2}z^4)H_{0,0}(e^{-j\pi/2}z^4)\Phi_T(e^{j\pi}z^4),\end{aligned} \tag{6.22}$$

where:

$$\Phi_T(z^4)=\sum_{a=0}^{3}H_{1,0}^2(e^{j(1+4a)\pi/4}z^2)H_{2,0}^2(e^{j(1+4a)\pi/8}z). \tag{6.23}$$

We note that $\Phi_T(z^4)$ comprises mirrored images of $H_T(z)$ at regular intervals of $\pi/2$, where we define:

$$H_T(z)=H_{1,0}^2(e^{j\pi/4}z^2)H_{2,0}^2(e^{j\pi/8}z). \tag{6.24}$$

Therefore, the coefficients of $\Phi_T(z)$ must be given by:

$$\Phi_T(n)=h_T(4n), \tag{6.25}$$

where $h_T(n)$ can be found by taking the inverse z-transform of (6.24).

The distortion error function of the filter bank can be found by first calculating the function:

$$\Theta_D(z)=G_{0,0}(e^{j\pi/2}z)H_{0,0}(e^{j\pi/2}z)\Phi_T(z)+G_{0,0}(e^{-j\pi/2}z)H_{0,0}(e^{-j\pi/2}z)\Phi_T(-z), \tag{6.26}$$

and then:

$$\Theta_T(z) = \frac{1}{2} X(z) \Theta_D(z^4). \tag{6.27}$$

The distortion error function $\Phi_T(z)$ can then be found using (6.24) and (6.25).

We find that as long as we use effective masking subfilters (the concept of effective masking subfilters was explained in Section 3.3.1), then the passband and transition bands of $G_{0,0}(e^{j\pi/2}z)H_{0,0}(e^{j\pi/2}z)$ fall within the passband of $\Phi_T(z)$. Thus, we can approximate $\Theta_D(z)$ according to:

$$\Theta_D(z) \approx G_{0,0}(e^{j\pi/2}z)H_{0,0}(e^{j\pi/2}z) + G_{0,0}(e^{-j\pi/2}z)H_{0,0}(e^{-j\pi/2}z), \tag{6.28}$$

when effective masking subfilters are used.

The contribution of the masking subfilters to the overall distortion error function of the filter bank is small when effective masking subfilters are used, and the overall distortion error function can be approximated from the distortion error function of the cascaded pair of analysis-synthesis 2-channel shaping subfilter.

6.2 Reconstruction analysis for *M* channels

We extend the results of our reconstruction analysis to the general *M*-channel case.

6.2.1 Distortion and aliasing

The input signal $x(n)$ with z-transform $X(z)$, after passing through the critically decimated analysis-synthesis M-channel filter bank pair, can be expressed using the following z-transform equation:

$$\hat{X}(z) = \sum_{m=0}^{M-1} \left\{ G_{[m]}(z) \left[\sum_{a=0}^{M-1} H_{[m]}(e^{j2\pi a/M} z) X(e^{j2\pi a/M} z) \right] \right\}. \tag{6.29}$$

The distortion error function $\Theta_T(z)$ is given by:

$$\Theta_T(z) = \frac{1}{2} X(z) \Theta_D(z^M), \tag{6.30}$$

where:

$$\Theta_D(z) = G_{0,0}(e^{j\pi/2}z)H_{0,0}(e^{j\pi/2}z)\Phi_T(z) + G_{0,0}(e^{-j\pi/2}z)H_{0,0}(e^{-j\pi/2}z)\Phi_T(-z). \tag{6.31}$$

The coefficients of $\Phi_T(z)$ are given by:

$$\Phi_T(n) = h_T(Mn), \tag{6.32}$$

where $h_T(n)$ is the inverse z-transform of:

$$H_T(z)=\prod_{u=1}^{K-1} H_{u,0}^2\left(e^{j\pi/2^{u+1}} z^{2^{K-1-u}}\right). \tag{6.33}$$

The aliasing error function is given by:

$$\begin{aligned}\Theta_A(z) \approx \frac{1}{4}\Big[&H_{0,0}(e^{-j\pi/2}z^M)G_{0,0}(e^{j\pi/2}z^M)+H_{0,0}(e^{j\pi/2}z^M)G_{0,0}(e^{-j\pi/2}z^M)\Big]\ldots \\ &\sum_{a=1}^{M-1} X(e^{j2a\pi/M}z)\Phi_A(z)_a,\end{aligned} \tag{6.34}$$

where:

$$\begin{aligned}\Phi_A(z)_a = &\prod_{u=1}^{K-1} H_{u,0}\left(e^{j(2a+1)\pi/2^{u+1}} z^{2^{K-1-u}}\right) H_{u,0}\left(e^{j(2a-1)\pi/2^{u+1}} z^{2^{K-1-u}}\right) \\ &+\prod_{u=1}^{K-1} H_{u,0}\left(e^{j(2a+1-2M)\pi/2^{u+1}} z^{2^{K-1-u}}\right) H_{u,0}\left(e^{j(2a-1-2M)\pi/2^{u+1}} z^{2^{K-1-u}}\right).\end{aligned} \tag{6.35}$$

By the use of effective masking subfilters, the distortion and aliasing error functions become approximately determinable from the transfer functions of the shaping subfilters. Denoting the peak magnitude response in the passband of $H_{0,0}(z)$ as $(1+\delta_{p(0,0)})$, and the peak magnitude response in the passband of $H_T(z)$ as $(1+\delta_{pT})$, the ripple in the distortion error function can be approximately bounded to first order by $(\delta_{p(0,0)}+\delta_{pT})$ if we assume $\delta_{p(0,0)} << 1$ and $\delta_{pT} << 1$.

6.2.2 *Selection of shaping subfilters*

The selection of masking subfilters is trivial in the consideration of the reconstruction properties of the filter bank, as long as they are effective masking subfilters. It is possible to use half-band subfilters, for example. Since the overall reconstruction properties of the M-channel filter bank pair are primarily dependent on the reconstruction properties of the 2-channel shaping subfilter pair, we consider their design here. In general, we are able to replace the shaping subfilters by a pair of PR or NPR 2-channel filter banks. There are many sources of literature available on this subject, such as [13]-[15].

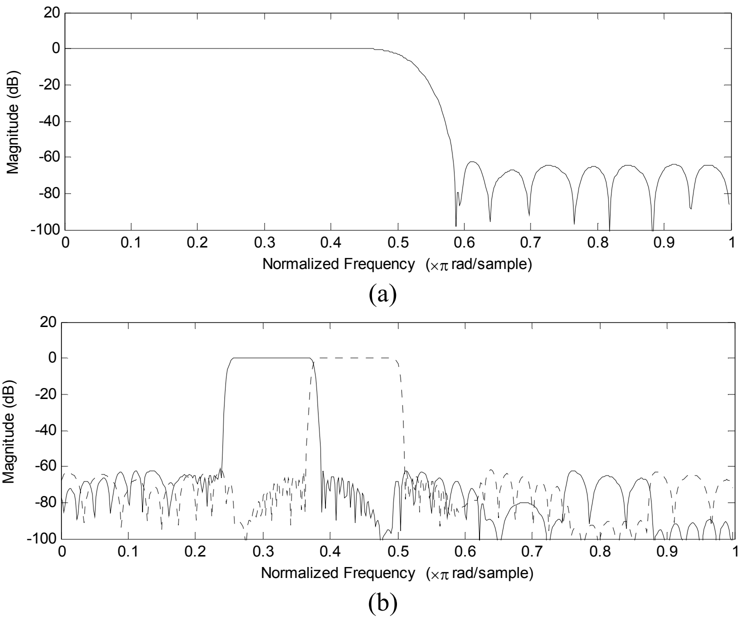
Here, we would like to highlight the 2-channel quadrature mirror filter (QMF) banks [16], where the transfer functions of the outputs are related by $\overline{H}_{0,0}(z)=H_{0,0}(-z)$. This requires less processing because the coefficients of $\overline{H}_{0,0}(z)$ are related to the coefficients of $H_{0,0}(z)$.

As a side note, we recall from literature that typical 2-channel PR/NPR filter bank pair designs have the following properties: (i). $G_{0,0}(z)=H_{0,0}(z^{-1})$, (ii). $H_{0,0}(z)H_{0,0}(z^{-1})$ is approximately a half-band filter, and (iii). the phase of $H_{0,0}(z)$ at $\pi/2$ frequency is approximately equal to $\pm\pi/4$. As an example, the Johnston 64D filter in [17] can be used as a shaping subfilter in our multirate filter bank. This filter has the added advantage that

$H_{0,0}(z)$ has linear phase and therefore $G_{0,0}(z) = H_{0,0}(z^{-1}) = H_{0,0}(z)$. The synthesis filter bank is then the exact mirror-image of the analysis filter bank. Other filter designs are available [18] where the filter length is exactly twice the number of subbands.

6.2.3 Eight-channel filter bank using the Johnston 64D filter

In this example, we design an 8-channel odd-stacked Type *F-I* FFB. The outputs of the analysis filter bank are critically decimated by a factor of 8. We used the linear-phase Johnston 64D filter for our shaping subfilters, $H_{0,0}(z) = G_{0,0}(z)$. Its frequency response is shown in Figure 6-2(a). The masking subfilters are designed using half-band filters with lengths of $N_{1,0} = 15$, $N_{2,0} = 11$ and $N_{3,0} = 7$. The solid line in Figure 6-2(b) shows the frequency response of the filter bank $H_{[m]}(z)$ for *m*=2 and the dotted line for *m*=3.



(a)

(b)

Figure 6-2 (a) Frequency response of $H_{0,0}(z)$,
(b) Frequency responses of $H_{[m]}(z)$ for *m*=2 and 3.

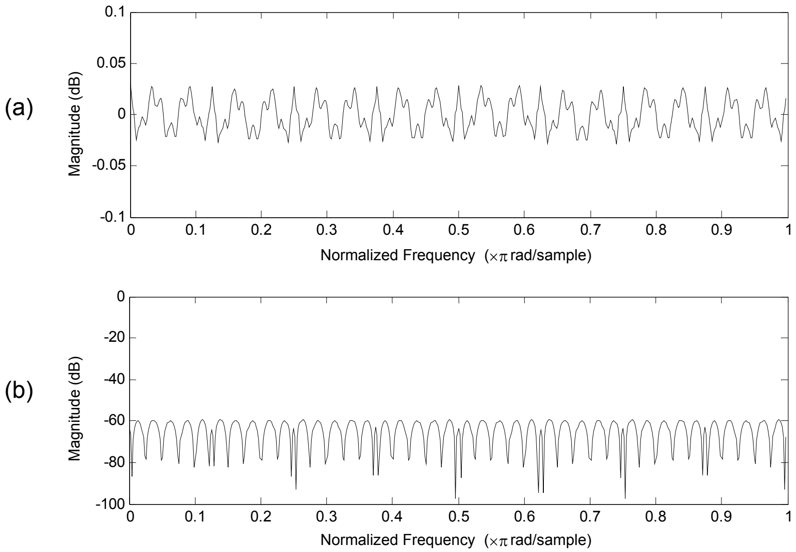


Figure 6-3 (a) Spectrum of distortion error function of filter bank,
(b) Spectrum of aliasing error function of filter bank for a=1.

The spectrum of the distortion error function $\Theta_T(z)$ is plotted in Figure 6-3(a) and the spectrum of the aliasing error function $\Theta_A(z)_{|a=1}$ is plotted in Figure 6-3(b). Note that the use of a PR or NPR filter bank pair for the shaping subfilters typically leads to a small increase in the overall complexity of the filter bank, compared to the use of an equiripple filter (for example). This is because the PR/NPR filter pair has a larger set of constraints, and hence larger filter length than an equivalent equiripple filter matching the pre-defined specifications. One such constraint is that the peak stopband ripple of the PR/NPR filter is approximately the square root of the peak passband ripple (in order that $H_{0,0}(z)G_{0,0}(z)+H_{0,0}(-z)G_{0,0}(-z)\approx 1$). For example, the Johnston 64D has a filter length of 64. The filter length of an equipripple filter with similar transition width, peak passband and stopband ripple is approximately 50. For the FFB however, the contribution of the shaping subfilter to the overall complexity is small, compared to the masking subfilters. For this example, the overall increase in complexity of the FFB, by using the Johnston 64D instead of the equiripple filter, is approximately 5-8%.

6.3 Design of filters using signed-powers-of-two terms

6.3.1 Signed-powers-of-two overview

A real number x, in the range of $-1<x<1$, can be expressed to a precision 2^{-B} as:

$$x_{SPT}=\sum_{b=1}^{B}s(x,b)2^{-b}, \qquad s(x,b)\in\{-1,0,1\}\,. \tag{6.36}$$

The number x_{SPT} is a sum of B ternary digits $s(x,b)$ weighted by 2^{-b}, where $s(x,b)$ is used to denote the value of the b-th ternary digit. The ternary digit $s(x,b)$, $s \in \{-1,0,1\}$, is called a signed-powers-of-two (SPT) term.

The SPT-r reduction of the number x is denoted as $x_{SPT\text{-}r}$, and consists of not more than r non-zero SPT terms. SPT terms which are equal to zero, s=0, do not add to x_{SPT} and can be removed. As an example, the SPT-2 reduction of the number $x = -0.19$ for B=5 is $x_{SPT-2} = -2^{-2} + 2^{-4} = -2^{-3} - 2^{-4} = -0.1875$. Note that the set of $s(x,b)$ that gives rise to the same $x_{SPT\text{-}r}$ is not necessarily unique. Two sets of values of $s(x,b)$ for the given example are tabulated in Table 6-1. In this case, the error of the SPT-2 reduction of x is equal to 0.0025.

b	5	4	3	2	1
$s(x,b)$	0	+1	0	-1	0
	0	-1	-1	0	0

Table 6-1 Example showing the values of $s(x,b)$ for the SPT-2 reduction of the number x= -0.19, for B=5.

Using a smaller value of r leads to a larger average error in the SPT-r reduction of x as fewer values of x are represented, but has the advantage that fewer computations are required during the processing of operations involving $x_{SPT\text{-}r}$. For example, the conventional multiplication of two B-bit binary numbers x and y can be expressed as:

$$xy = \sum_{b=1}^{B} 2^{-b} x_b y, \tag{6.37}$$

where x_b is the b-th bit of x. Since the computation of the terms in the summation bracket uses binary shifts and does not require any additions ($x_b y$=y if x_b=1 and $x_b y$=0 if x_b=0), evaluation of xy requires B shifts and B-1 additions. On the other hand, the multiplication of the B-bit binary number y with the number $x_{SPT\text{-}r}$ requires only r shifts and r-1 (r<B) additions. For example, assuming that the number $x_{SPT-2} = s(x,b_1)2^{-b_1} + s(x,b_2)2^{-b_2}$, the multiplication of x and y can be computed as $x_{SPT-2}y = s(x,b_1)2^{-b_1}y + s(x,b_2)2^{-b_2}y$, which requires 2 shifts and 1 addition.

Given a number x, the following algorithm (taken from [45]) can be used to efficiently find the nearest value of $x_{SPT\text{-}r}$ to a precision of 2^{-B}:

1. Initialize b = 1 and $x_0 = x$.
2. Find $s(x,b)2^{-g(b)}$ which minimizes x_{b-1} - $s(x,b)2^{-g(b)}$, where $g(b)$ is an integer describing the bit position of the SPT term, $1 \le g(b) \le B$.
3. If either $s(x,b)$ = 0 or b = r, go to step 6, otherwise go to step 4.
4. Update $x_b = x_{b-1}$ - $s(x,b)2^{-b}$.
5. Set b = b+1. Go to step 2.
6. $x_{SPT-r} = \sum_{b'=1}^{b} s(x,b')2^{-g(b')}$. Stop.

6.3.2 On the design of FIR filters

In the design of finite-precision FIR filters having coefficients with a precision of 2^{-B}, it is known that the minimum peak stopband ripple is constrained by B. For a fixed B, increasing the filter length beyond a certain value provides no improvement to the peak stopband ripple [43]. It is also known that directly rounding the filter coefficients to their nearest quantized value may not yield the optimal frequency response. To overcome this problem, integer programming methods (using objective functions which minimize the peak passband and stopband ripples) have been proposed to design finite-precision FIR filters [44].

Attempts to further reduce the filter coefficients using SPT terms have been proposed more recently [45]-[47]. Similarly, integer programming methods can also be used to find solutions to the SPT reduction of the filter coefficients [48]-[52]. Filters designed using these integer-programming methods typically exhibit better frequency response characteristics such as smaller peak stopband ripples, when compared to direct rounding of the filter coefficients to their nearest SPT terms. Filters which have coefficients that are reduced to SPT terms can then be implemented 'multiplierlessly', relying instead on the use of shifters and adders which are more cost-effective to implement.

For a long filter (e.g. having a filter length greater than 50), optimizing its coefficients to SPT-r terms, with respect to its frequency response, can be a tedious process. Consider the optimization of a single coefficient $h(n)$, $-1 < h(n) < 1$, to $h_{SPT-r}(n)$ having a precision of 2^{-B}. The number of possible values that $h_{SPT-r}(n)$ can take on is a function of r and B, and is denoted as $f(B,r)$. For example, $f(B,1) = 2B + 1$ and $f(B,B) = 2^{B+1} - 1$. For a filter having a length of N, an exhaustive search of all possible combinations of SPT-r filter coefficients requires $\eta f(B,r)^N$ operations, where η is the number of operations required to evaluate each possibility (e.g. calculating the frequency response, measuring the peak stopband ripple, etc.). The search space increases exponentially with the filter length N. For example, the search space for N=20, B=16, r=3 is equal to 3421^{20}.

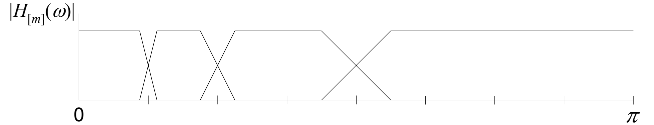
Fortunately, with algorithms which localize the search space and make use of mixed integer linear programming [52], the time required for optimizing a set of SPT terms can be greatly reduced. However, the time required is still prohibitive for very long filters. The FRM filter banks proposed in this thesis use subfilters with short filter lengths, and are thus suitable for design using SPT techniques.

6.4 Variable bandwidth filter bank

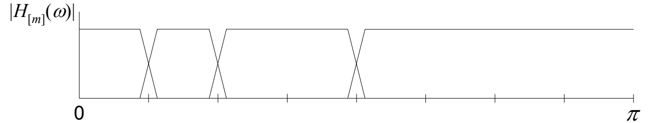
We showed that the overall frequency response and reconstruction characteristics of an FRM filter bank are heavily influenced by the frequency response of its shaping subfilter. Its masking subfilters are tolerant to changes in their coefficient values. The coefficients of the masking subfilters can thus be reduced to B-bit fixed-point values, such as by quantization, e.g. rounding to the nearest B-bit term, integer programming, etc. Alternatively, the coefficients of the masking subfilters may be reduced to SPT terms (which have a lower complexity but also lower numerical precision than quantization), such as in the mentioned literature [55]-[59]. Research into the design of multiplierless filter banks, such as by using the SPT reduction technique, is primarily focused on the 2-channel case [55]-[59]. We extend this work to the scenario with multiple channels using the FRM filter bank, which comprises cascaded 2-channel subfilters.

Consider the following variable bandwidth scenarios (Figure 6-4). Scenario 1 is typical of applications such as speech or image coding, where the bandwidth increases with frequency. Scenario 2 is similar to Scenario 1, except that the transition widths are equal. Scenario 3 depicts subbands with arbitrary bandwidths, with the limitation that the transition edges are clearly defined on multiples of the unit frequency value given by $2^{-K+1}\pi$, where K is an integer value greater than 1. Scenario 4 depicts subbands with arbitrary bandwidths and arbitrary transition edges.

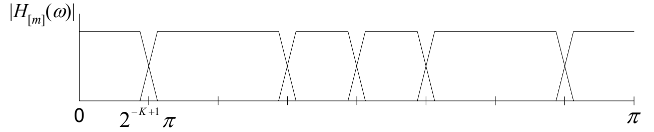
Octave filter banks are popularly used in (and restricted to) the Scenario 1 application [13], [53]-[54]. Designs of octave filter banks usually involve a tree-structured cascade of 2-channel QMF filter banks (section 2.5). Since each 2-channel QMF filter bank is critically decimated, octave filter banks are restricted to critically decimated outputs.



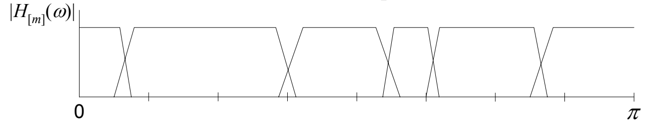
(a) Scenario 1: Varying transition width.



(b) Scenario 2: Equal transition width.



(c) Scenario 3: Arbitrary bandwidths constrained to multiples of $2^{-K+1}\pi$.



(d) Scenario 4: Arbitrary bandwidths.

Figure 6-4 Some variable bandwidth scenarios.

In this section, we consider the design of a general-purpose variable bandwidth filter bank with arbitrary decimation factor. We apply the Type *F-I* filter bank to the implementation of variable bandwidth filter banks. The masking subfilter coefficients are optimized using SPT algorithms. The resultant implementation can support a flexible variety of non-uniform bandwidth combinations and an arbitrary output decimation factor (oversampled or critically sampled). Very small transition widths between adjacent subbands can be achieved at a very low computational complexity. We then demonstrate the results using a 5-channel design.

6.4.1 Overview of proposed method

Our method can be applied to Scenarios 1-3, as long as the transition bands between adjacent subbands are centered on multiples of the frequency $2^{-K+1}\pi$, where K is the number of cascade stages. The proposed architecture for a 5-stage (K=5) variable bandwidth system for Scenario 2 is shown in Figure 6-5. The architecture shown uses the Type *F-I* structure, with the shaping stage brought to the back-end instead of the front-end. The summing stage of the filter bank can also be configured in other combinations for different bandwidth requirements. The corresponding synthesis filter bank can be obtained by the mirror image of the analysis filter bank and is thus omitted from our discussion.

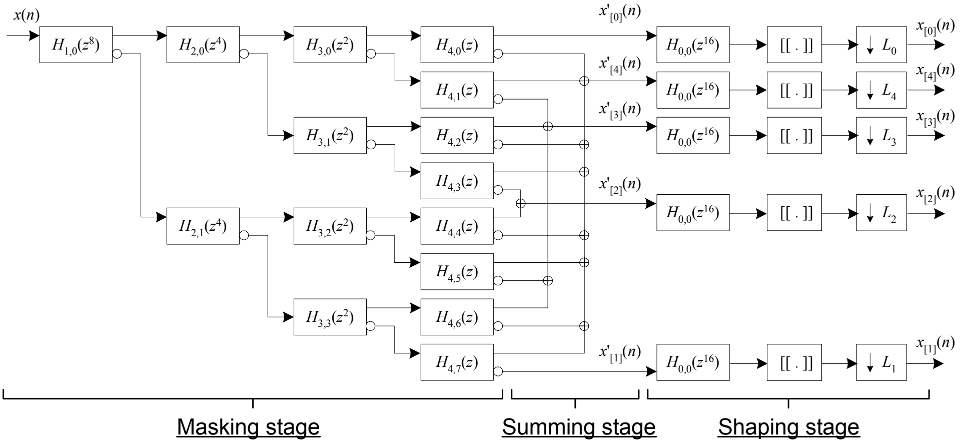


Figure 6-5 Proposed 5-stage architecture for variable bandwidth application.

For the case when the outputs are critically decimated, we are able to simplify the shaping stage to Figure 6-6.

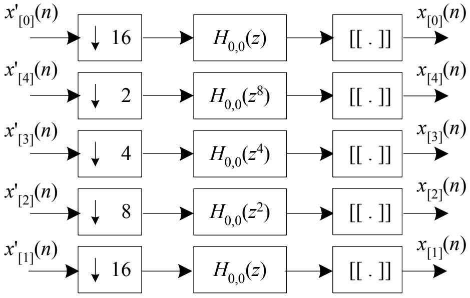


Figure 6-6 Further simplification of the shaping stage.

6.4.2 Design procedure

The design procedure for the filter bank is as follows:

(i). Design the pair of prototype shaping subfilters with transfer functions $H_0(z)$ and $G_0(z)$. The 2-channel shaping subfilter pair should fulfill the reconstruction criteria for the filter bank.

(ii). Design a set of prototype masking subfilters $H_u(z)$ for each of the cascade stages, for $1 \le u \le K-1$. The transfer functions $H_u(z)$ are lowpass and have real-valued coefficients. The transition widths of $H_u(z)$ can be found from (3.13).

(iii). The transfer functions $H_{u,v}(z)$ can then be found from:

$$H_{u,v}(z) = H_u\left(e^{-j\left(\frac{\pi}{2^{u+1}}+\frac{2\pi\tilde{v}}{2^K}\right)}z\right), \text{ for } 0 \le u \le K-1, \tag{6.38}$$

where $\tilde{v}$ is the bit-reversed version of v in K-1 bits.

(iv). The set of masking subfilters for the synthesis filter bank is the mirror image of the analysis filter bank, $G_{u,v}(z) = H_{u,v}(z)$ for $1 \le u \le K-1$.

(v). Reduce the masking subfilter coefficients for $H_{u,v}(z)$, $1 \le u \le K-1$ to their SPT terms.

6.4.3 Design result

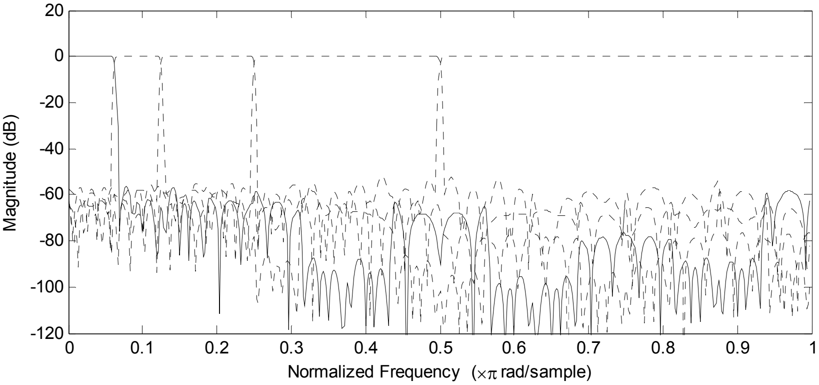
In our 5-channel variable bandwidth filter bank design example, we used the Johnston 64D filter [17] for the prototype shaping subfilters $H_0(z)$ and $G_0(z)$. The transfer functions of the shaping subfilters $H_{0,0}(z)$ and $G_{0,0}(z)$ were obtained using (6.38). Since $H_{0,0}(z)$ and $G_{0,0}(z)$ were modulated by $\pi/2$, we note that each coefficient of $H_{0,0}(z)$ and $G_{0,0}(z)$ is either purely real or purely imaginary. This observation can be used to further reduce the complexity of the filtering operation, compared to the case where coefficients have both real and imaginary parts (details are provided in Appendix B).

We used half-band filters for the masking subfilters. A search of the local SPT space around the infinite-precision coefficients was used. We used mixed SPT-2 and SPT-3 terms with a 16-bit wordlength. The resultant SPT coefficients are listed in Table 6-2. Since the subfilters have linear phase and are therefore conjugate-symmetrical about the center coefficient, we listed only the coefficients $h_{u,v}(n)$ for $(N_{u,v}-1)/2 \le n \le N_{u,v}-1$, where $N_{u,v}$ is the length of the subfilter with transfer function $H_{u,v}(z)$.

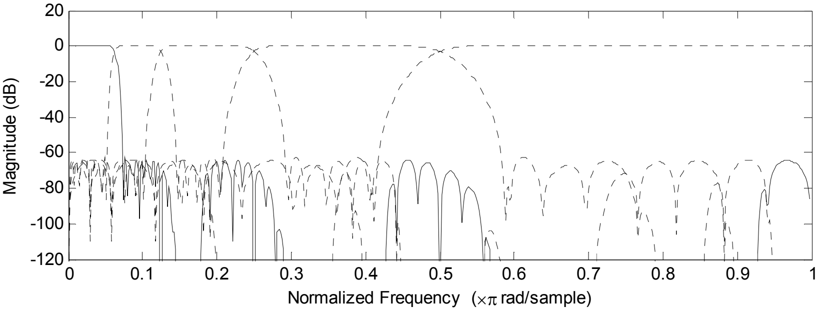
u,v	SPT terms of $h_{u,v}(n)$	u,v	SPT terms of $h_{u,v}(n)$
1,0	$[2^{-1}]$	2,0	$[2^{-1}]$
	$[2^{-2}-2^{-5}-2^{-10}] . [1+j]$		$[2^{-2}+2^{-6}] +j[2^{-3}-2^{-6}]$
	0		0
	$[2^{-4}-2^{-7}] . [1-j]$		$-[2^{-6}-2^{-10}]-j[2^{-5}-2^{-8}]$
	0	2,1	$[2^{-1}]$
	$[2^{-6}+2^{-9}] . [1+j]$		$-[2^{-3}-2^{-6}]+j[2^{-2}+2^{-6}]$
	0		0
	$[2^{-8} - 2^{-11}] . [1-j]$		$-[2^{-5}-2^{-8}]+j[2^{-6}-2^{-10}]$
3,0	$[2^{-1}]$	4,0	$[2^{-1}]$
	$[2^{-2}+2^{-5}] +j[2^{-4}-2^{-7}]$		$[2^{-2}-2^{-11}] +j[2^{-6}+2^{-7}]$
	0	4,1	$[2^{-1}]$
	$-[2^{-5}+2^{-15}] -j[2^{-6}+2^{-8}]$		$-[2^{-6}+2^{-7}] +j[2^{-2}-2^{-11}]$
3,1	$[2^{-1}]$	4,2	$[2^{-1}]$
	$-[2^{-4}-2^{-7}] +j[2^{-2}+2^{-5}]$		$[2^{-3}+2^{-5}+2^{-10}] +j[2^{-3}+2^{-4}+2^{-7}]$
	0	4,3	$[2^{-1}]$
	$-[2^{-6}+2^{-8}] +j[2^{-5}+2^{-15}]$		$-[2^{-3}+2^{-4}+2^{-7}] +j[2^{-3}+2^{-5}+2^{-10}]$
3,2	$[2^{-1}]$	4,4	$[2^{-1}]$
	$[2^{-3}+2^{-5}] +j[2^{-2}-2^{-6}]$		$[2^{-2}-2^{-5}] +j[2^{-3}-2^{-8}+2^{-10}]$
	0	4,5	$[2^{-1}]$
	$[2^{-5}-2^{-11}] -j[2^{-8}+2^{-9}]$		$-[2^{-3}-2^{-8}+2^{-10}] +j[2^{-2}-2^{-5}]$
3,3	$[2^{-1}]$	4,6	$[2^{-1}]$
	$-[2^{-2}-2^{-6}] +j[2^{-3}+2^{-5}]$		$[2^{-4}+2^{-7}+2^{-10}] +j[2^{-2}-2^{-7}-2^{-9}]$
	0	4,7	$[2^{-1}]$
	$-[2^{-8}+2^{-9}] -j[2^{-5}-2^{-11}]$		$-[2^{-2}-2^{-7}-2^{-9}] +j[2^{-4}+2^{-7}+2^{-10}]$

Table 6-2 SPT coefficients for masking subfilters used in 5-channel variable bandwidth filter bank.

The frequency responses of the subbands, $H_{[m]}(z)$ are shown in Figure 6-7(a). The stopband attenuation is approximately 60 dB. Consider the unequal transition width *Design A*: a 5-channel octave filter bank (see Figure 2-8), where the Johnston 64D filter is used in place of the subfilters $H_0(z)$, $H_1(z)$, $H_2(z)$ and $H_3(z)$. Such a system has a frequency response as shown in Figure 6-7(b). We notice the larger transition widths associated with subbands that have larger bandwidths. Alternatively, consider *Design B*: a narrow transition width 5-channel octave filter bank, having transition widths similar to our proposed design. The filter lengths required for $H_0(z)$, $H_1(z)$, $H_2(z)$ and $H_3(z)$ would be approximately 1024, 512, 256 and 128 respectively.



(a) Frequency response of our 5-channel variable bandwidth filter bank.



(b) Frequency response of unequal transition width *Design A*: Octave filter bank with Johnston 64D shaping subfilters.

Figure 6-7 Frequency responses for different variable bandwidth filter banks.

6.4.4 Comparisons

Compare our 5-channel variable bandwidth filter bank with the octave filter bank *Design B*. Their frequency responses are similar: the transition widths are approximately 0.01π, and the stopband attenuations are approximately 60 dB. The number of adds, shifts, multiplies and the delay for the analysis filter banks are tabulated in Table 6-3. Our design has a greatly reduced complexity and delay when compared to the octave filter bank (*Design B*). The number of multipliers required is reduced by a factor of 10 and the delay is reduced by a factor of 4.

	Multiplies	Adds	Shifts	Delay
Our design	32	272	168	579
Design B	340	340	-	2040.5

Table 6-3 Comparisons between our variable bandwidth filter bank and the narrow transition width octave filter bank.

Compare our 5-channel variable bandwidth filter bank with the octave filter bank *Design A*. Their stopband attenuations are 60 dB. For *Design A*, different subbands have different transition widths. The transition widths for the different subbands, from the lowest frequency to the highest frequency, are approximately 0.02π, 0.04π, 0.08π, and 0.17π. On the other hand, our design has a transition width of 0.01π for all subbands. The number of adds, shifts, multiplies and delay for the analysis filter banks are tabulated in Table 6-4. Their complexities are similar, assuming that adders and shifters are inexpensive compared to multipliers. Our design has about a 20% larger delay than *Design A*.

	Multiplies	Adds	Shifts	Delay
Our design	32	272	168	579
Design A	30	30	-	472.5

Table 6-4 Comparisons between our variable bandwidth filter bank and the unequal transition width octave filter bank.

6.4.5 Discussions

The FRM filter bank structure is suited for the implementation of multi-rate uniform and non-uniform filter banks, when small transition widths are required, or in hardware implementations that can benefit from multiplierless reductions of the subfilter coefficients. When implemented in software, shifts and adds do not gain significant efficiency over multiplications, as computers are typically not optimized for such operations.

When the number of channels is increased, the difficulty of the design is also increased, especially if SPT reduction is to be performed on all the masking subfilters. However, this is an initial expenditure, and we would expect the FRM filter bank to operate efficiently, for small transition widths, even for a large number of channels.

It is important to note that the objective of the work done here is mainly to establish some groundwork on the application of FRM filter banks to multi-rate applications, which has not been done before. There is certainly a large scope of material for future work and improvements. One such aspect is to quantify the sensitivities of the overall frequency responses and reconstruction properties of the filter bank to variations (including SPT reduction) in the masking subfilter transfer functions. Also, our design in Section 6.4.3 uses a trial-and-error approach to determine which coefficients are reduced to SPT-2 terms and which are reduced to SPT-3 terms. It would be interesting to analyze the effect of different combinations of mixed SPT-r terms, in the context of multi-stage cascaded subfilters; i.e. methods to optimize each subfilter, in terms of determining r and $h_{SPT\text{-}r}(n)$ for each coefficient $h(n)$, such that the overall frequency responses and reconstruction properties when they are cascaded fall within the desired specifications.

6.5 Summary

We analyzed the effects of distortion and aliasing caused by a pair of FRM analysis-synthesis filter banks. We derived the distortion and aliasing error functions for the M-channel FFB, and found that the masking subfilters make only a minor contribution to these functions. Therefore, the design of NPR FRM filter banks can be suitably achieved by using PR/NPR 2-channel filter banks as the shaping subfilters. Furthermore, since the masking subfilters have only a small impact on the distortion and aliasing error functions as well as on the overall filter bank transfer functions, we proposed to reduce the coefficients of the masking subfilters to SPT terms. The proposed methods were applied to the design of a 5-channel variable bandwidth filter bank with small transition widths. The result was a reduction in complexity by a factor of up to 10, when compared to an octave filter bank.

References for Part I

[1] LR. Rabiner, and O. Herrmann, "The Predictability of Certain Optimum Finite Impulse Response Digital Filters," *IEEE Trans. on Circuit Theory*, Vol. CT-20, No. 4, pp. 401-408, July 1973.

[2] L.R. Rabiner, J.H. McClellan, T.W. Parks, "FIR Digital Filter Design Techniques Using Weighted Chebyshev Approximation", *IEEE Proc*, vol 63 (1975), pp 595-610.

[3] R.W. Schafer and L.R. Rabiner, "A digital signal processing approach to interpolation", *Proeedings.of the IEEE*, vol. 61, pp. 692-702, June 1973.

[4] P.P Vaidyanathan, "Theory and design of *M*-channel maximally decimated quadrature mirror filters with arbitrary *M*, having the perfect-reconstruction property", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-35, pp. 476-492, Apr 1987.

[5] P.P. Vaidyanathan, "A tutorial on multirate digital filter banks", *Proc. IEEE Int. Symposium on Circuits and Systems,* pp. 2241-2248, June 1988.

[6] M. Vetterli, "A theory of multirate filter banks", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-35, pp. 356-372, Mar 1987.

[7] M. Bellanger, G. Bonnerot and M. Coudreuse, "Digital filtering by polyphase network: Application to sample rate alteration and filter banks," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 24, pp. 109-114, Apr. 1976.

[8] P. P.Vaidyanathan, *Multirate Systems and Filter Banks*, Englewood Cliffs, NJ: Prentice Hall, 1992.

[9] J.G Proakis, D.G. Manolakis, *Digital Signal Processing: Principles, Algorithms and Applications*, 3rd edition, Prentice-Hall, 1996.

[10] W.H. Chen, C.H. Smith and S.C. Fralick, "A fast computational algorithm for the discrete cosine transform", *IEEE Trans. Commun.*, vol. 25, pp. 1004-1008, Sep 1977.

[11] B.G. Lee, "A new algorithm to compute the discrete cosine transform," *IEEE Trans. Acoust. Speech Signal Processing*, vol. 32, pp. 1243-1245, Dec 1984.

[12] M. Vetterli and H.Nussbaumer, "Simple FFT and DCT algorithms with reduced number of operations", *Signal Processing*, vol. 6 no. 4, pp. 267-278, Aug 1984.

[13] N.J. Fliege, *Multirate digital signal processing*, Wiley, New York NY, 1994.

[14] M.J.T. Smith and T.P. Barnwell III, "A procedure for designing exact reconstruction filter banks for tree structured sub-band coders", *Proc. Int. Conf. Acoust., Speech, Signal Processing*, pp. 434-441, 1986.

[15] F. Mintzer, "Filters for distortion-free two-band multirate filter banks", *IEEE Trans. Acoust. Speech Signal Processing*, pp. 626-630, 1985.

[16] T.Q. Nguyen and P.P. Vaidyanathan, "Two-channel perfect-reconstruction FIR QMF structures which yield linear-phase analysis and synthesis filters," *IEEE Trans. Acoust. Speech Signal Processing*, vol. 37, pp. 676-690, May 1989.

[17] J.D. Johnston, "A filter family designed for use in quadrature mirror filter banks", *IEEE Int. Conf. Acoust., Speech, Signal Processing*, vol. 5, pp. 291-294, Apr 1980.

[18] H.S. Malvar, *Signal Processing with Lapped Transforms*, Artech House: Norwood MA, 1992.

[19] E.B. Richardson and N.S. Jayant, "Subband coding with adaptive prediction for 56 kbit/s audio", *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. ASSP-34, pp. 691-696, Aug 1986.

[20] D. Esteban and C. Galand, "Application of quadrature mirror filters to split band voice coding schemes", *Proc. of IEEE Int. Conf Acoust., Speech, Signal Processing*, pp. 191-195, May 1977.

[21] Y.C Lim and B. Farhang-Boroujeny, "Fast Filter Bank (FFB)", *IEEE Transactions on Circuits and Systems II*: vol. 39, no. 5, pp. 316-318, May, 1992.

[22] Y.C Lim and B. Farhang-Boroujeny, "Analysis and Optimum Design of the FFB", *IEEE International Symposium on Circuits and Systems*: vol. 2, pp. 509-512, 1994.

[23] Y.C Lim, "Frequency response masking approach for the synthesis of sharp linear phase digital filters", *IEEE Transactions on Circuits and Systems*: vol. 33, No. 4, pp. 357-364, Apr 1986.

[24] Y.C Lim and Y. Lian, "The Optimum Design of One- and Two-Dimensional FIR Filters Using the Frequency Response Masking Technique", *IEEE Transactions on Circuits and Systems II*: vol. 40, No. 2, pp. 88-95, Feb 1993.

[25] F. Mintzer, "On half-band, third-band, and Nth-band FIR filters and their design", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 30, pp. 734-738, Oct 1982.

[26] M. Bellanger, "Computation rate and storage estimation in multirate digital filtering with half-band filters", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 25, pp. 344-346, Aug 1977.

[27] K. Nayebi, TP. Barnwell, MJT. Smith, "Low delay FIR filter banks: design and evaluation", *IEEE Trans. on Signal Processing*, vol. 42, pp. 24-31, Jan. 1994.

[28] P. Millar, "Mirror filters with minimum delay responses for use in subband coders", *IEEE International Conference on ASSP*, vol. 9, pp. 444-447, Mar 1984.

[29] R. Bregovic, T. Saramaki, "Design of two-channel low-delay FIR filter banks using constrained optimization", *First Int'l Workshop on Image and Signal Processing and Analysis*, pp. 223-228, June 2000.

[30] B. Farhang-Boroujeny, *Adaptive Filters: Theory and Applications*, pp. 309-314, John Wiley & Sons, 2000.

[31] B. Farhang-Boroujeny and Y.C Lim, "A Comment on the Computational Complexity of Sliding FFT", *IEEE Trans. Circuits and Systems II*: vol. 39, no. 12, pp. 875-876, Dec 1992.

[32] L.R. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall, 1975.

[33] J.W. Lee and Y.C. Lim, "Efficient Implementation of Real Filter Banks using Frequency Response Masking Techniques", *IEEE APCCAS Conference 2002*, vol. 1, pp 69-72, Oct. 2002.

[34] J.W. Lee and Y.C. Lim, "Designing the Fast Filter Bank with a Minimum Complexity Criterion",
IEEE ISSPA Conference 2003, vol. 2, pp 279-282, July 2003.

[35] Y.C. Lim and J.W. Lee, "Matrix Formulation: Fast Filter Bank", *IEEE International Conference on ASSP 2004*, vol. 5, pp 133-136, May 2004.

[36] D.R. Wille, *Advanced Scientific Fortran*, John Wiley & Sons, 1995.

[37] C.H. Koelbel, D.B. Loveman, R.S. Schreiber, G.L. Steele Jr., and M.E. Zosel, *The High Performance Fortran Handbook*, The MIT Press, 1994.

[38] "Basic Linear Algebra Subprograms (BLAS)", [online] http://www.netlib.org/blas.

[39] J.W. Lee and Y.C. Lim, "A multiplierless filter bank with deep stopband suppression and narrow transition width", *IEEE ISCAS 2005*, May 2005.

[40] Netto S.L., Diniz, P.S.R., Barcellos L.C.R., “Efficient implementation for cosine-modulated filter banks using the frequency response masking approach”, *IEEE ISCAS 2002*, vol. 3, pp 229-232, May 2002.

[41] Diniz, P.S.R.; Barcellos, L.C.R.; Netto, S.L., “Design of cosine-modulated filter bank prototype filters using the frequency-response masking approach”, *IEEE Proceedings of ICASSP 2001*, Vol. 6 , pp. 3621 -3624, May 2001

[42] Rosenbaum, L.; Lowenborg, P.; Johansson, H., “Cosine and sine modulated FIR filter banks utilizing the frequency-response masking approach”**,** *IEEE Proceedings of ISCAS 2003*, Vol. 3,pp 882-885, May 2003.

[43] D. Kodek and K. Steiglitz, “Filter-length wordlength tradeoffs in FIR digital filter design”, *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-28, pp. 739-744, Dec 1980.

[44] D. Kodek, “Design of optimal finite wordlength FIR digital filters using integer programming techniques”, *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-28, pp. 304-308, June 1980.

[45] Lim, Y.C., Evans, J.B., Liu, B., "Decomposition of binary integers into signed power-of-two terms", *IEEE Transactions on Circuits and Systems*, vol. 38, pp 667-672, June 1991.

[46] YC. Lim, AG Constantinides, “Linear phase FIR digital filter without multipliers,” *Proc. IEEE Int. Symp. Circuits and Systems*, pp. 185-188, 1979.

[47] Y.C. Lim, Parker S., "FIR filter design over a discrete powers-of-two coefficient space", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 31, pp 583–591, Jun 1983.

[48] Y.C. Lim, R. Yang, D.N. Li, J.J. Song, "Signed power-of-two term allocation scheme for the design of digital filters", *IEEE Trans. on Circuits and Systems II*, vol. 46, pp. 577–584, May 1999.

[49] Lim Y.C., Liu B., "Design of cascade form FIR filters with discrete valued coefficients", *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 36, pp. 1735-1739, Nov. 1988.

[50] Zhao Q., Tadokoro Y., "A simple design of FIR filters with powers-of-two coefficients", *IEEE Trans. on Circuits and Systems*, vol. 35 , pp. 566–570, May 1988.

[51] C.L. Chen, Willson A.N. Jr, "A trellis search algorithm for the design of FIR filters with signed-powers-of-two coefficients", *IEEE Trans. on Circuits and Systems II*, vol. 46, pp. 29–39, Jan. 1999.

[52] Y. C. Lim, S. R. Parker, and A. G. Constantinides, “Finite wordlength FIR filter design using integer programming over a discrete coefficient space,” *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-30, pp. 661–664, Aug. 1982.

[53] E.B. Richardson and N.S. Jayant, "Subband coding with adaptive prediction for 56 kbit/s audio", *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. ASSP-34, pp. 691-696, Aug 1986.

[54] D. Esteban and C. Galand, "Application of quadrature mirror filters to split band voice coding schemes", *Proc. of IEEE Int. Conf Acoust., Speech, Signal Processing*, pp. 191-195, May 1977.

[55] Chan S.C., Liu W., Ho K.L., "Multiplierless perfect reconstruction modulated filter banks with sum-of-powers-of-two coefficients", *IEEE Signal Processing Letters*, vol. 8, pp. 163–166, June 2001.

[56] Horng BR., Samueli H., Wilson AN. Jr., "The design of two-channel lattice-structure perfect-reconstruction filter banks using powers-of-two coefficients", *IEEE Trans. on Circuits and Systems I*, vol. 40, pp. 497–499, July 1993.

[57] Liu, W., Chan S.C., Ho K.L., "Low-delay perfect reconstruction two-channel FIR/IIR filter banks and wavelet bases with SOPOT coefficients", *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 109 - 112 5-9 June 2000.

[58] Wah B.W., Yi Shang, Zhe Wu, "Discrete Lagrangian method for optimizing the design of multiplierless QMF filter banks", *IEEE International Conference on Application-Specific Systems, Architectures and Processors*, pp. 529–538, July 1997.

[59] Mao J.S., Lu W.S., Chan S.C., Antoniou A., "Design and multiplierless implementation of two-channel biorthogonal IIR filter banks with low system delay", *IEEE International Symposium on Circuits and Systems*, vol 2, pp. 465-468, , 6-9 May 2001.

PART II:
AUDIO TRANSCODING

Chapter 7

Overview of Audio Transcoding

In Part II of this thesis, we focus on the subject of audio transcoding. To re-iterate from the introduction in Chapter 1, the initial objective behind our work on audio transcoding was to apply efficient filter banks to the implementation of fast transcoders, with the expectation that more efficient filter banks would lead to faster transcoders. As we progressed, we found that an ultra-fast transcoder (subject to fulfilling the sample-synchronization condition) that does not require filter banks at all can be achieved. Further to this, we also realized that sample-synchronous transcoding presents several problems, in terms of audio quality, particularly for MP3.

As a result, we re-directed our work to focus on the implementation of 'filter bank-less' transcoders. This is covered in detail in Chapter 10. Since audio quality is an important aspect of transcoding, we further delved into several issues relevant to audio quality, that arise during transcoding. These issues, which comprise Chapters 8 and 9, are considered in our implementation of the ultra-fast transcoder.

In this chapter, we first provide an overview of the audio coding process, which uses compression methods to reduce the bit-rate. This is followed by an overview of audio transcoding. The cascaded quantization model which we use to represent the audio transcoding process is also introduced. We then define and explain some important terms that are frequently used in this thesis.

7.1 Introduction to audio coding

7.1.1 Audio compression methods

Digital audio is stored in the form of a string of bits, called a bitstream. Its bit-rate indicates the number of bits used to transmit or store the information for 1 second of audio. Audio coding uses audio compression methods to obtain high quality audio at low bit-rates so as to facilitate transmission or storage.

Lossless compression methods [61], in which the encoded audio can be decoded perfectly, are often able to achieve a compression ratio of 1.2 to 1.7 over the original PCM source. Some well-known lossless compression methods include the Free Lossless Audio Codec (FLAC) [62], and the MPEG 4-Audio Lossless Coding (ALS) extension [63]-[64].

Lossy compression methods [65]-[68], on the other hand, can reach compression ratios greater than 10 (about 128 kbps for stereo audio sampled at 44.1 kHz). Errors are introduced during encoding, however, these errors are usually controlled by a psychoacoustic modeling process to ensure a reasonable standard of reproduction quality. Many lossy compression methods are available today, such as Windows Media Audio (WMA) [69] and Ogg-Vorbis (OGG) [70]. More popularly known, is the MPEG 1 ISO/IEC standard [71]-[72] and its extensions – Layer 1, Layer 2 and Layer 3 (MP3). More recently, this has evolved into the improved MPEG 2 format [73], and its Advanced Audio Coding (AAC) extension [74]. In this thesis, we focus on the MPEG 1 Layer 2 and MPEG 1 Layer 3 (MP3) compression methods.

7.1.2 Psychoacoustic models

It is well-known that the human auditory system (HAS) responds to audio stimuli in a frequency-selective manner [75]-[79]. Figure 7-1 shows the absolute hearing threshold of the human ear in a quiet environment, measured in terms of dB SPL (decibels, sound pressure level) [80]. Sound pressure levels which are below the curve cannot be detected by the average human listener. It can be observed that the human ear tends to be less sensitive towards the lower and upper ends of the hearing spectrum.

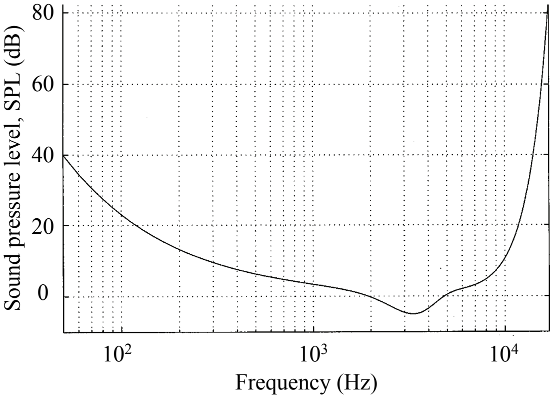


Figure 7-1 Absolute threshold of hearing in quiet.

Furthermore, in the presence of a tonal stimulus (masker), the hearing threshold is modified. The hearing threshold in the neighborhood frequencies of the masker are raised, and noise and other tones falling below the raised hearing threshold cannot be heard (i.e. are masked). This concept is known as auditory masking. Figure 7-2 shows the modified hearing threshold (masking threshold) in the presence of 3 tonal maskers at 0.25 kHz, 1 kHz and 4 kHz.

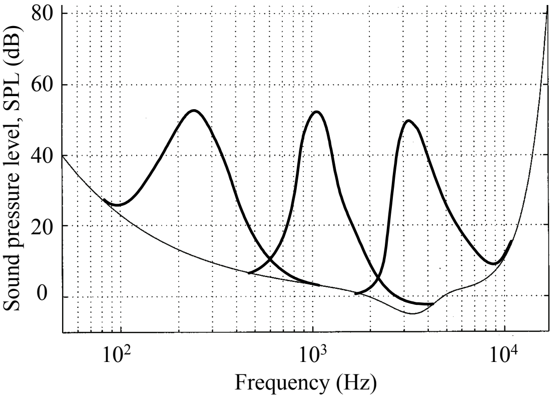


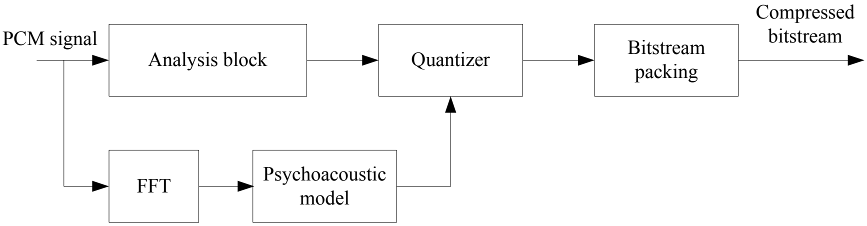
Figure 7-2 Modified hearing threshold in the presence of certain tonal stimuli.

In lossy audio coders, a psychoacoustic model is commonly used to analyze the frequency content of a segment of audio samples. The masking threshold is then estimated based on known models of the HAS. The audio samples are quantized so that they can be encoded using fewer bits. The quantization is controlled by the encoder such that the noise caused by the quantization error remains below the masking threshold. The noise is then inaudible to the human ear, and the compressed audio is 'perceptually transparent'.

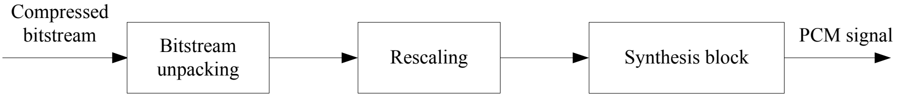
In lossy audio coding, the input digital audio signal is usually represented using as few bits as possible, while achieving perceptually transparent output audio quality. For the MP3 method, perceptually transparent quality can be achieved at approximately 128-192 kbps, which corresponds to a compression ratio of about 10. Newer and improved methods such as the AAC method claim perceptual transparency at about 64-96 kbps.

7.1.3 Audio encoder and decoder overview

Figure 1-1 (repeated here) shows a simplified architecture of a typical lossy audio encoder. For most lossy audio encoders, the PCM input signal is first separated into its constituent subbands by an analysis block. The total number of subbands varies, depending on the compression method that is used.



(a) Typical audio encoder



(b) Typical audio decoder

(Repeat of Figure 1-1.)

Note: The analysis block may comprise a set of multi-stage analysis filter bank operations. In some cases, this block comprises only a single analysis filter bank, e.g. for MPEG 1 Layer 2. In other cases, the analysis block comprises cascaded filter bank processing. For MP3, as an example, the analysis block comprises an analysis filter bank cascaded with an MDCT and an aliasing reduction stage (see Appendix C).

Referring to Figure 1-1, the PCM inputs are analyzed using a psychoacoustic model to estimate the masking thresholds for different time-segments of the audio. The subband samples are then quantized using the masking threshold information from the psychoacoustic model. In subbands where the masking thresholds are higher, coarser quantizers are used and vice versa. The quantized subband samples are packed, along with the reconstruction information (e.g. global gain, scalefactors, etc.) and other descriptors (e.g. bit-rate, sampling frequency, compression method used, song title, etc.), into the output bitstream. The output bitstream is checked to ensure that the target bit-rate is met. A more detailed explanation and block diagram for the MP3 method can be found in Appendix C.

The decoder is usually more straightforward to implement. The compressed bitstream is first unpacked, and the subband samples are then rescaled by using the reconstruction information. The PCM output is reconstructed from the rescaled subband samples by means of the synthesis block.

7.2 Introduction to audio transcoding

Audio transcoding is broadly defined as the conversion of compressed audio from an encoded bitstream to another encoded bitstream. Audio transcoding might involve a change in any of the following: compression method (e.g. from MP3 to AAC), bit-rate, sampling frequency, number of channels, etc.

In this thesis, we focus on audio transcoding in the context of changing the bit-rate (usually a reduction in the bit-rate), while the other properties remain primarily unchanged. Bit-rate reduction is used to decrease the storage space required by the audio, at the expense of a decreased audio quality. Possible applications of bit-rate reduction include:

(i). Transfer of songs from the computer to hardware portable players. Since storage space is abundant on the computer harddisk, audio stored in computers tend towards high quality and high bit-rate. Hardware portable players are often storage-expensive; medium-quality and low bit-rate are preferred.

(ii). Transfer of songs over the internet. High quality and high bit-rate audio are stored on a server. The client might wish to preview some songs at a lower quality and bit-rate (such as prior to deciding which song to buy), so as to improve data transfer speed over the internet.

7.2.1 Perceptual quality in audio transcoding

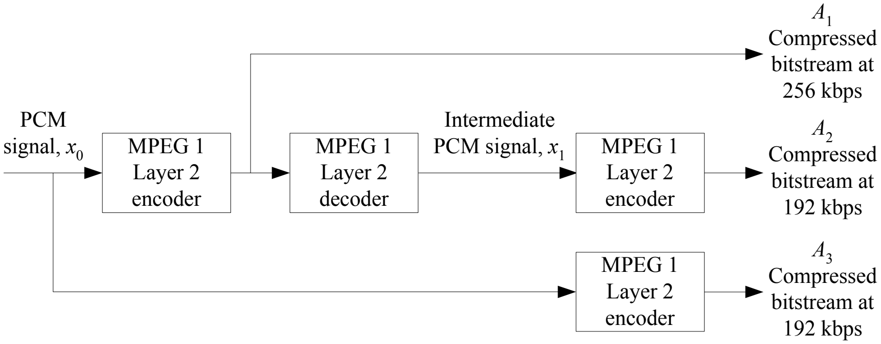

PCM
signal, x_0
MPEG 1 Layer 2 encoder
MPEG 1 Layer 2 decoder
Intermediate PCM signal, x_1
MPEG 1 Layer 2 encoder
MPEG 1 Layer 2 encoder
A_1 Compressed bitstream at 256 kbps
A_2 Compressed bitstream at 192 kbps
A_3 Compressed bitstream at 192 kbps

Figure 7-3 Transcoding from 256 kbps to 192 kbps for the MPEG 1 Layer 2 method.

Figure 7-3 shows an example of bit-rate transcoding where an MPEG 1 Layer 2 bitstream at 256 kbps (labeled A_1) is recompressed to 192 kbps (labeled A_2). As a reference, we included the bitstream A_3 which is directly compressed to 192 kbps from the original source.

The result of transcoding on audio quality has been studied [81]-[83], and it is generally well known that the audio quality of A_2 is inferior to that of A_3. In Chapter 10, we conduct a series of listening tests (detailed results included in Appendix E) which also demonstrates this behaviour. In Figure E-3, listeners generally rated *LT-2* (representative of A_3 in this example) to have better audio quality than *LT-3* and *LT-4* (representative of A_2 in this example). The effect of transcoding on audio quality is further investigated in Chapters 8 and 9.

7.2.2 Literature on audio transcoding

The main areas of interest in the subject of transcoding are: i) improvement in audio quality, and ii) complexity reduction in transcoder implementation. To achieve these objectives, different transcoding methods have been proposed. Some of these transcoding methods conform to the specifications of the original compression method (in the sense that conventional decoders are sufficient to play back the transcoded audio), others are proprietary (in the sense that specialized decoders are required to play back the transcoded audio). Here, we provide an overview of the background work. In subsequent chapters, more details will be provided as necessary to the topic of each chapter.

Some methods that have been proposed to improve transcoding audio quality are based on recognizing certain conditions which are known to result in better audio quality. These conditions are then enforced during transcoding. In [84] for example, the audio quality of

MPEG 1 Layer 2 audio was related to the total input-to-output delay of the analysis/synthesis filter bank block. By inserting an 'optimal delay' between the decoder and the encoder during transcoding, audio quality can be improved. The effect of delay on transcoding is studied in Chapters 8 and 9.

Other methods to improve transcoding audio quality rely on the inclusion of additional information in the encoded bitstream. The European Advanced Television at Low Bit-rates And Networked Transmission over Integrated Communication systems (ATLANTIC) project [90]-[91], is a system that adopts the MPEG 1 audio and video coding schemes for the production and distribution of television programs. For the MPEG 1 Layer 2 method, a "MOLE" signal is proposed to be embedded into the bitstream [92]. The "MOLE" signal contains information that is specifically used during transcoding to improve audio quality in the output.

In [93]-[95], specially designed 'new' compression methods are proposed. The data in these methods is encoded, or structured, in such a way that it is robust to transcoding. For example, the "Audio Layered Transcoder" [99] structures the data in layers of increasing bit-rate. Transcoding can be performed by simply truncating the high bit-rate layers. We discuss the Audio Layered Transcoder in Chapter 10. Literature on complexity reduction in transcoder implementation is also covered in Chapter 10.

7.3 Cascaded quantization model

7.3.1 Overview

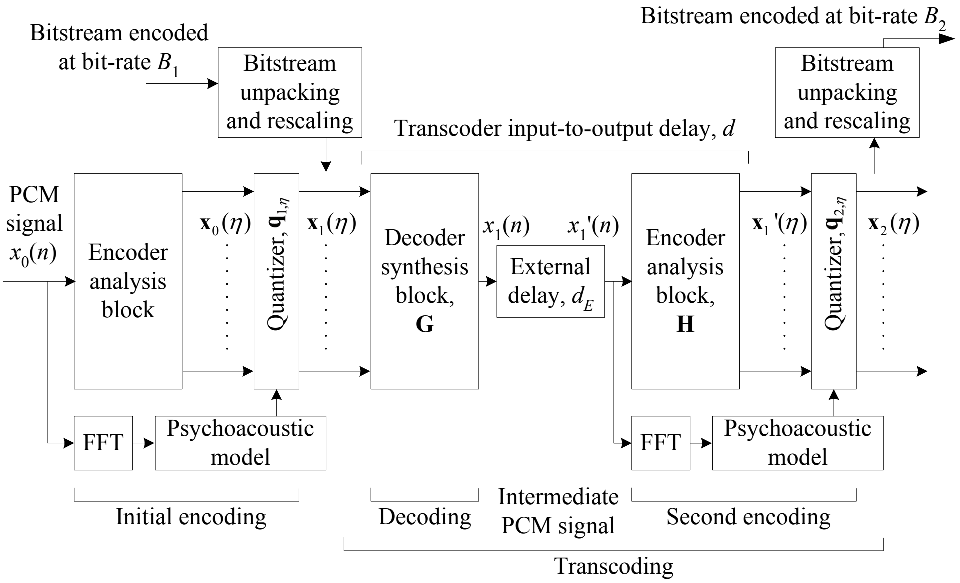

Bitstream encoded at bit-rate B_1
Bitstream unpacking and rescaling
Bitstream encoded at bit-rate B_2
Bitstream unpacking and rescaling
Transcoder input-to-output delay, d
PCM signal $x_0(n)$
Encoder analysis block
$\mathbf{x}_0(\eta)$
Quantizer, $\mathbf{q}_{1,\eta}$
$\mathbf{x}_1(\eta)$
Decoder synthesis block, $\mathbf{G}$
$x_1(n)$
$x_1'(n)$
External delay, d_E
Encoder analysis block, $\mathbf{H}$
$\mathbf{x}_1'(\eta)$
Quantizer, $\mathbf{q}_{2,\eta}$
$\mathbf{x}_2(\eta)$
FFT
Psychoacoustic model
FFT
Psychoacoustic model
Initial encoding
Decoding
Intermediate PCM signal
Second encoding
Transcoding

Figure 7-4 Cascaded quantization model for audio transcoding.

A transcoder can be represented as a cascade of a decoder with an encoder (see repeat of Figure 1-1, and Figure 7-3). Since the bitstream packing and unpacking does not change the information, they are removed from our transcoder model. We represent the transcoder using the cascaded quantization model shown in Figure 7-4. The source PCM signal is denoted as $x_0(n)$. The input to the transcoder is denoted as $\mathbf{x}_1(\eta)$, and contains the subband components retrieved from the bitstream encoded at bit-rate B_1. The output of the transcoder is denoted as $\mathbf{x}_2(n)$, and is then placed into the bitstream encoded at bit-rate B_2.

The total number of subbands is denoted by M, and m indexes the m-th subband. Some values of M for different compression methods are shown in Table 7-1.

	MPEG 1 Layer 1 & 2	MP3	AAC	WMA
M	32	576	1024	2048

Table 7-1 Some examples of the number of subbands M for different compression methods.

We use n and η to denote sample time. Since, for the compression methods of interest (i.e. MPEG 1 Layer 2 and MP3), the outputs of the analysis block and the inputs to the synthesis block are critically decimated ($L=M$, where L is the decimation factor), we assume in this thesis that η corresponds to a sampling rate $1/M$ times that of n.

7.3.2 Initial encoding

The PCM signal $x_0(n)$ is separated into its subband components $\mathbf{x}_0(\eta)$:

$$\mathbf{x}_0(\eta) = \left[x_{0[0]}(\eta) \quad x_{0[1]}(\eta) \quad \cdots \quad x_{0[M-1]}(\eta) \right]^T, \tag{7.1}$$

where $x_{0[m]}(\eta)$ is the subband component for the subband m.

The vector of subband components $\mathbf{x}_0(\eta)$ is then quantized to a vector $\mathbf{x}_1(\eta)$ during the initial encoding. The set of quantizers used is represented by the symbol $\mathbf{q}_{1,\eta}$. The vector $\mathbf{x}_1(\eta)$ represents the quantized subband components of the compressed audio at bit-rate B_1:

$$\begin{aligned} \mathbf{x}_1(\eta) &= \left[x_{1[0]}(\eta) \quad x_{1[1]}(\eta) \quad \cdots \quad x_{1[M-1]}(\eta) \right]^T \\ &= \left[q_{1[0],\eta}\left(x_{0[0]}(\eta)\right) \quad q_{1[1],\eta}\left(x_{0[1]}(\eta)\right) \quad \cdots \quad q_{1[M-1],\eta}\left(x_{0[M-1]}(\eta)\right) \right]^T, \end{aligned} \tag{7.2}$$

where $q_{1[m],\eta}$ denotes the quantizer used for the subband m, at the sample time η.

7.3.3 Transcoding

The synthesis and analysis blocks in the transcoder are represented by **G** and **H** respectively. The outputs $\mathbf{x}'_1(\eta)$ of the transcoder analysis block can be obtained from the inputs $\mathbf{x}_1(\eta)$:

$$\mathbf{x}'_1(\eta) = \mathbf{F}_d\left(\mathbf{x}_1(\eta)\right). \tag{7.3}$$

The set of transform functions $\mathbf{F}_d$ describes the input-to-output transformation due to the synthesis-analysis block pair of the transcoder, and varies depending on the value of the input-to-output delay *d*. The input-to-output delay *d* of the transcoder can be modified by means of an external delay d_E. The external delay can be easily introduced by delaying the decoded intermediate PCM samples $x_1(n)$. Appendix D provides a more complete explanation of the operation of $\mathbf{F}_d$, and the effect of adjusting *d* on $\mathbf{F}_d$ (a summary is available in Section D.4).

A second quantization, which is represented by $\mathbf{q}_{2,\eta}$, is applied to $\mathbf{x}'_1(\eta)$. The vector $\mathbf{x}_2(\eta)$ represents the quantized subband components of the compressed audio at bit-rate B_2, where:

$$\mathbf{x}_2(\eta) = \left[q_{2[0],\eta}\left(x'_{1[0]}(\eta)\right) \quad q_{2[1],\eta}\left(x'_{1[1]}(\eta)\right) \quad \cdots \quad q_{2[M-1],\eta}\left(x'_{1[M-1]}(\eta)\right)\right]^T. \tag{7.4}$$

7.3.4 Expanded view

The bit-rate transcoding process is sometimes referred to as a cascaded quantization process, because the input signal undergoes two successive quantizations $\mathbf{q}_{1,\eta}$ and $\mathbf{q}_{2,\eta}$. Since the errors incurred during audio transcoding are mainly due to these quantizations, let us focus on their relationship, using the expanded view shown in Figure 7-5(a).

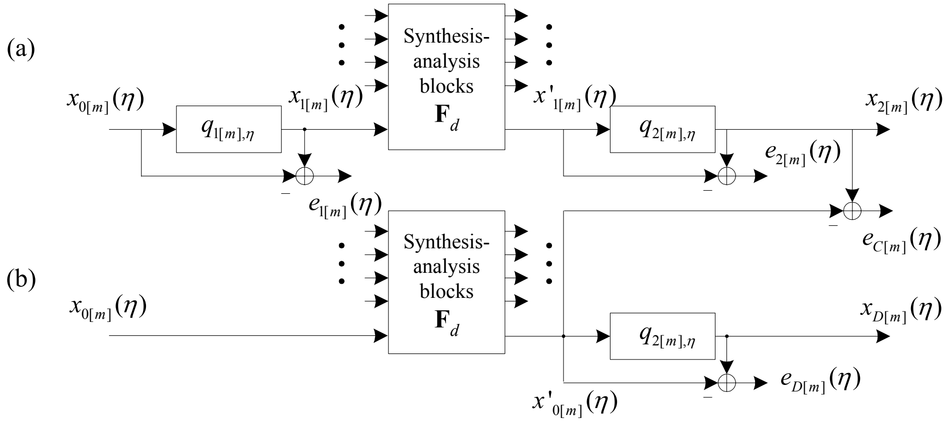


Figure 7-5 Expanded view of cascaded quantization model.

The subband components $x_{0[m]}(\eta)$, $x_{1[m]}(\eta)$, $x'_{1[m]}(\eta)$ and $x_{2[m]}(\eta)$ were defined earlier. The quantizers represented by $q_{1[m],\eta}$ and $q_{2[m],\eta}$ were also defined earlier.

The quantization error in the initial encoder due to $q_{1[m],\eta}$ is denoted by:

$$e_{1[m]}(\eta) = x_{1[m]}(\eta) - x_{0[m]}(\eta) . \tag{7.5}$$

The quantization error in the transcoder due to $q_{2[m],\eta}$ is denoted by:

$$e_{2[m]}(\eta) = x_{2[m]}(\eta) - x'_{1[m]}(\eta) . \tag{7.6}$$

For comparison purposes, we show the same system in Figure 7-5(b) with $q_{1[m],\eta}$ removed. The subband components $x'_{0[m]}(\eta)$ are transformed versions of $x_{0[m]}(\eta)$ using $\mathbf{F}_d$, and are subsequently quantized to $x_{D[m]}(\eta)$ using $q_{2[m],\eta}$.

Since the samples $x_{D[m]}(\eta)$ are only quantized once for the system in Figure 7-5(b), the system can also be viewed as a single encoding process (i.e. no transcoding takes place). The quantization error in the single encoding process due to $q_{2[m],\eta}$ is denoted by:

$$e_{D[m]}(\eta) = x_{D[m]}(\eta) - x'_{0[m]}(\eta) . \tag{7.7}$$

The total quantization error due to the cascaded quantization using $q_{1[m],\eta}$ and $q_{2[m],\eta}$ (with the transformation $\mathbf{F}_d$ due to the synthesis-analysis blocks in Figure 7-5(a)) is denoted by:

$$e_{C[m]}(\eta) = x_{2[m]}(\eta) - x'_{0[m]}(\eta) . \tag{7.8}$$

7.3.5 Note on variables used

We note that for most audio compression schemes, the quantizers $q_{1[m],\eta}$ and $q_{2[m],\eta}$ are time-varying with the sample time η. However, in part of our work, we use a fixed arbitrary sample time instant, and are not concerned with the time-varying properties of the signals or quantizers. When this is the case, it is convenient to hide the variable η in our equations. In this case, we use $q_{1[m]}$, $q_{2[m]}$ to equivalently represent the quantizers $q_{1[m],\eta}$, $q_{2[m],\eta}$. We use $x_{0[m]}$, $x_{1[m]}$, $x_{2[m]}$, $x_{D[m]}$, $e_{1[m]}$, $e_{2[m]}$, $e_{D[m]}$, $e_{C[m]}$ to equivalently represent samples of the signals $x_{0[m]}(\eta)$, $x_{1[m]}(\eta)$, $x_{2[m]}(\eta)$, $x_{D[m]}(\eta)$, $e_{1[m]}(\eta)$, $e_{2[m]}(\eta)$, $e_{D[m]}(\eta)$, $e_{C[m]}(\eta)$ respectively.

7.4 Definitions and conventions

For the purpose of standardization, we assume all audio in this thesis to be stereo and to have a sampling frequency of 44.1 kHz.

7.4.1 Quantization frames (q-frames)

The quantizers represented by $q_{1[m],\eta}$ and $q_{2[m],\eta}$ vary with subband *m* and sample time η. It is overly complicated to assign a different quantizer for each *m* and η, as it would then become necessary to record the information for each selected quantizer into the bitstream. Typically, a common quantizer is selected for a defined range of subbands *m* and/or times η. For example, the MP3 method uses 576 subbands. The subband samples are grouped into 22 scalefactor bands, and a quantizer is selected for each scalefactor band (details in Appendix C).

In this thesis, we define a quantization frame (q-frame) as a unit consisting of *M* by *b* subband samples, where *b* is the number of consecutive subband samples that are acted upon by a common quantizer.

7.4.2 Frames

A frame is a term which is commonly used in audio compression, and is clearly defined within the specifications of each compression method. A frame refers (such as in MPEG 1 Layer 2 or MP3) to a time-segment of encoded data which contains 1 or more q-frames, in addition to supplementary information used by the decoder to interpret the bitstream. For example, this supplementary information can contain information on the bit-rate, and the method used to pack the bitstream. More information is available in [71]. A short description of frames for the MP3 method is also provided in Appendix C.

The q-frame that we defined earlier is not to be confused with a frame. A q-frame for the MPEG 1 Layer 2 method consists of (M=32) by 12 subband samples, whereas a frame consists of 3 consecutive q-frames. A q-frame for the MP3 method consists of (M=576) by 1 subband samples, whereas a frame consists of 2 consecutive q-frames.

7.4.3 Sample-synchronization

During transcoding, the synthesis block is cascaded with the analysis block (Figure 7-4). The inputs $x_{1[m]}(\eta)$ of the synthesis block and the outputs $x'_{1[m]}(\eta)$ of the analysis block are usually critically decimated ($L=M$). When the total input-to-output delay *d* is equal to *kM* (where *k* is an integer), the outputs are delayed versions of the inputs:

$$x'_{1[m]}(\eta + d/M) = x_{1[m]}(\eta), \tag{7.9}$$

assuming that $\mathbf{F}_d$ is a pure delay.

The system is then said to be sample-synchronized. When *d* is neither 0 nor a multiple of *M*, the outputs are related to the inputs by a set of transforms $\mathbf{F}_d$ (see (7.3)). The system is then said to be non-sample-synchronized. When the transcoding system is not sample-synchronized, the external delay d_E can be varied to enforce sample-synchronization. These concepts are explained in Appendix D in greater detail.

Note that the value of d is known a-priori to the transcoder, since the transcoder is in control of the encoding and decoding processes. Hence, the transcoder can opt to operate in either the sample-synchronous or non-sample-synchronous mode.

For sample-synchronized transcoding, the cascaded quantizations become:

$$\begin{aligned} x_{2[m]}(\eta) &= q_{2[m],\eta}\left(x'_{1[m]}(\eta)\right) \\ &= q_{2[m],\eta}\left(q_{1[m],\eta-d/M}\left(x_{0[m]}(\eta-d/M)\right)\right). \end{aligned} \tag{7.10}$$

The errors due to the quantizations $q_{2[m],\eta}$ and $q_{1[m],\eta-d/M}$ become correlated in a predictable manner, and we call this a tandem quantization scenario. In Chapter 8, this is further analyzed.

It is generally known that for different audio coding methods, sample-synchronization can have different effects on transcoding. For transcoding in the MPEG 1 Layer 2 method, sample-synchronization is preferred over non-sample-synchronization. For the MP3 method, on the other hand, the reverse is true. The effect of sample-synchronization on transcoding for these 2 methods will be investigated in more detail in Chapters 8 and 9. Here, we show the effect of sample-synchronization on transcoding using a listening test example.

Figure 7-6 compares the effect of sample-synchronization for these 2 compression methods. The comparisons show the results of a listening test conducted with a variety of music material. In each case, listeners are placed in a quiet environment and are asked to listen to a piece of music encoded using different methods over a set of headphones. The encoding methods are as follows (with reference to Figure 7-3 and Figure 7-6):

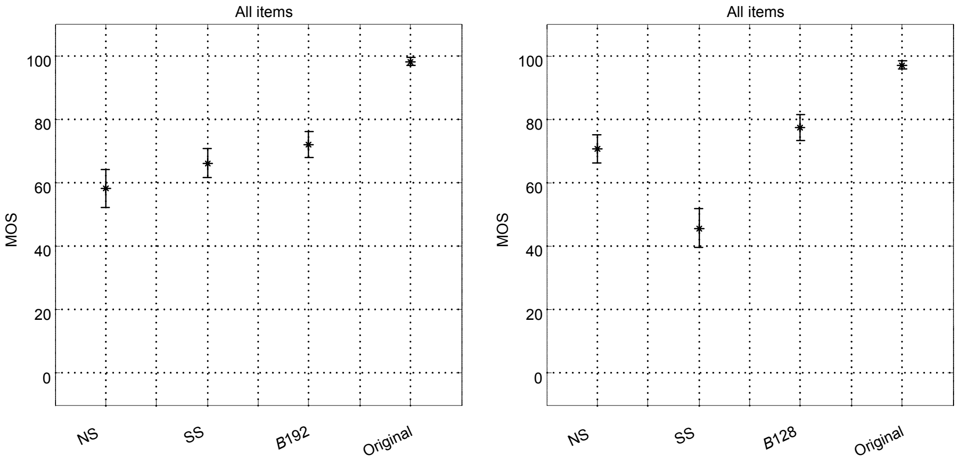
Original - PCM format of the original source material x_0;

B192 / B128 - A_3, which is directly encoded from the PCM source to 192 kbps (for MPEG 1 Layer 2) / 128 kbps (for MP3);

NS - A_2, non-sample-synchronized transcoding from 256 kbps to 192 kbps (Figure 7-6(a), for MPEG 1 Layer 2) or from 192 kbps to 128 kbps (Figure 7-6(b), for MP3);

SS - A_2, sample-synchronized transcoding from 256 kbps to 192 kbps (Figure 7-6(a), for MPEG 1 Layer 2) or from 192 kbps to 128 kbps (Figure 7-6(b), for MP3).

The original is placed as a hidden reference in the listening test. The listeners are asked to rate the perceived quality (Opinion Score) of each music item on a scale of 0 to 100. The plots show the Mean Opinion Score (MOS) and the 95% confidence interval for all listeners. It can be observed that for music items which are encoded using the MPEG 1 Layer 2 method, sample-synchronized transcoding leads to better audio quality (higher MOS). The reverse is true for music items which are encoded using the MP3 method.



(a) MPEG 1 Layer 2.

(b) MP3.

Figure 7-6 Comparison of the effect of sample-synchronization on transcoding for MPEG 1 Layer 2 and MP3.

7.4.4 *Frame-synchronization*

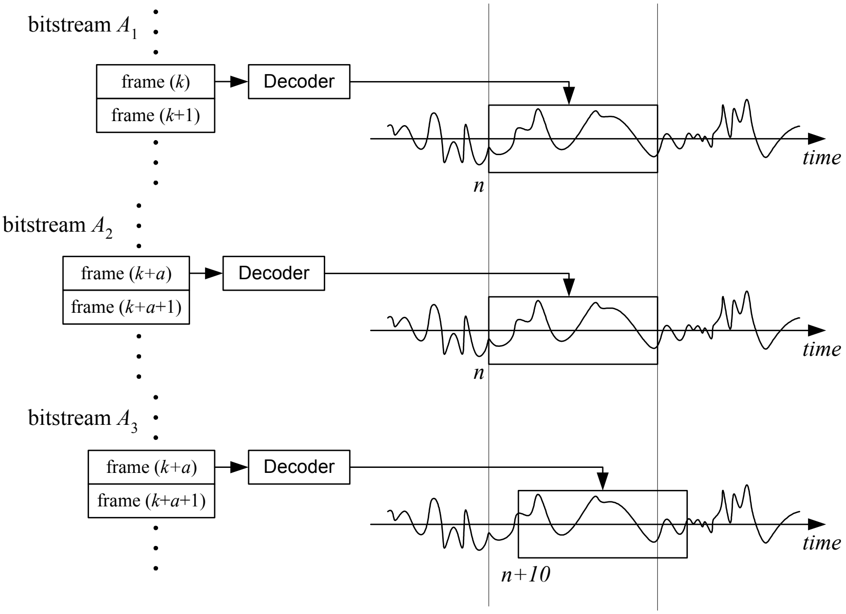


Figure 7-7 Illustration of frame-synchronization.

In this thesis, we define "frame-synchronization" to refer to the condition when each frame in the bitstream A_1 is related to each frame in the transcoded bitstream A_2 by a delay which is a multiple of the frame size (number of samples in each frame). Thus, for each frame in bitstream A_1 which is decoded to time samples within a time region, there is a corresponding frame in the bitstream A_2 which is decoded to time samples within the same time region. This is illustrated in Figure 7-7. We note that since the frame size is a multiple of M, frame-synchronized bitstreams are always sample-synchronized.

Example: For MP3, each frame consists of 1152 samples. For delays that are not multiples of M=576, the bitstreams are neither frame-synchronized nor sample-synchronized. For delays that are not multiples of 1152, the bitstreams are not frame-synchronized. In Figure 7-7, the indicated frames are decoded to samples which are contained in the boxed regions. Bitstreams A_1 and A_2 are frame-synchronized. Bitstreams A_1 and A_3 are neither frame-synchronized nor sample-synchronized.

Note that we usually assume non-frame-synchronization and non-sample-synchronization when it is not specified, because the input-to-output delay d of the transcoder is usually neither a multiple of M nor a multiple of the frame size. However, frame-synchronization or sample-synchronization can be easily enforced by inserting an external delay d_E between the decoder and encoder, since d is known a-priori to the transcoder.

7.4.5 Glossary and Appendix information

Frequently used notational conventions are provided in the Glossary.

In Appendix C, a brief description of the MP3 method is provided. In Appendix D, a mathematical relationship between the inputs $\mathbf{x}_1(\eta)$ and the outputs $\mathbf{x}'_1(\eta)$ of the cascaded quantization model is derived. In Appendix E, we conducted and recorded the results for 2 formal listening tests, based on audio material that is encoded or transcoded using different methods.

7.5 Overview of quantization schemes

The quantization schemes for the MPEG 1 Layer 2 and MP3 method are central to our work. In this section we provide an overview of these quantization schemes and explain the notations used. We hide the sample-time variable n and η in this section.

7.5.1 Overview of quantization used in MPEG 1 Layer 2

The MPEG 1 Layer 2 quantization scheme is shown in Figure 7-8. The vector $\mathbf{x}_0$ consists of 32 subband samples. A q-frame consists of 32 by 12 subband samples. The subband samples are first grouped into a q-frame, and are then normalized to the range of -1 to 1 by dividing by a scaling factor $\zeta_{1[m]}$. The scaling factors are selected from a set of values pre-defined by the ISO specifications. The action of the scaling factor $\zeta_{1[m]}$ on the subband signal $x_{0[m]}$ is illustrated in Figure 7-9.

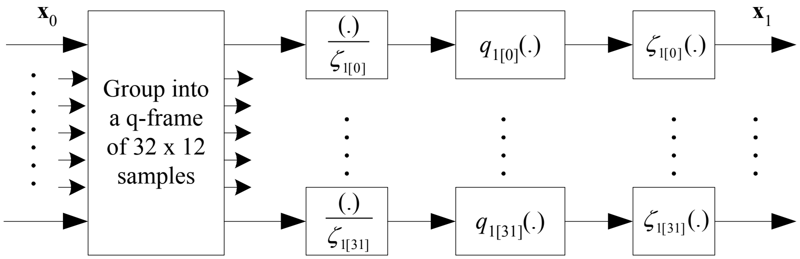

$\mathbf{x}_0$
Group into a q-frame of 32 x 12 samples
$\frac{(.)}{\zeta_{1[0]}}$
$q_{1[0]}(.)$
$\zeta_{1[0]}(.)$
$\mathbf{x}_1$
$\frac{(.)}{\zeta_{1[31]}}$
$q_{1[31]}(.)$
$\zeta_{1[31]}(.)$

Figure 7-8 A block diagram showing the MPEG 1 Layer 2 quantization scheme.

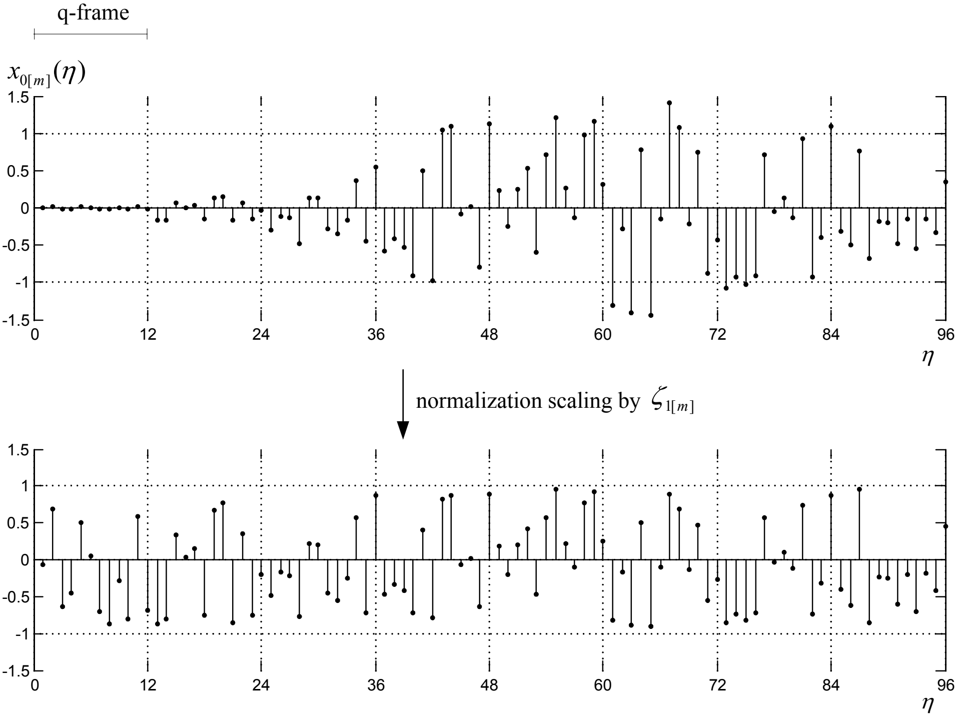
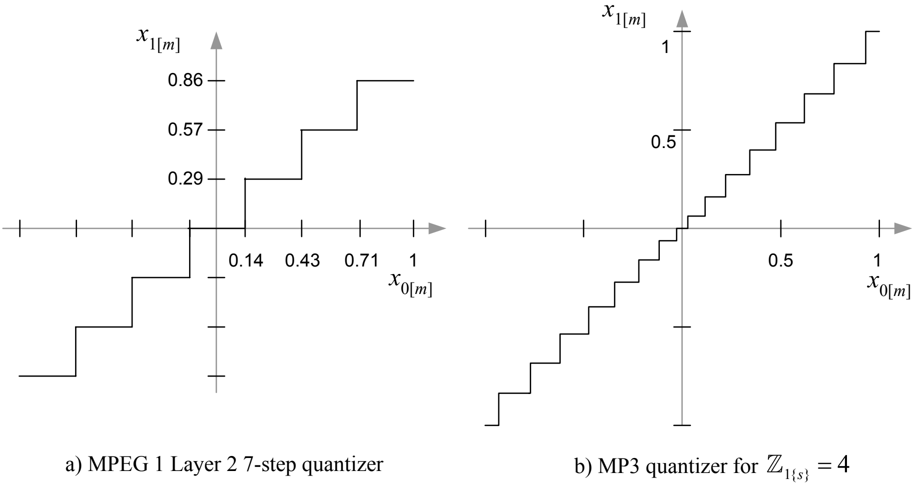


Figure 7-9 Illustration of normalization scaling used in MPEG 1 Layer 2.

The normalized samples are then quantized using $q_{1[m]}$. The quantizer is selected from a set of uniform quantizers pre-defined by the ISO specifications. As an illustration, Figure 7-10(a) shows the quantization characteristics of the 7-step quantizer that is used in the Layer 2 method.



a) MPEG 1 Layer 2 7-step quantizer b) MP3 quantizer for $\mathbb{Z}_{1\{s\}} = 4$

Figure 7-10 Plot of quantization characteristics.

7.5.2 *Overview of quantization used in MP3*

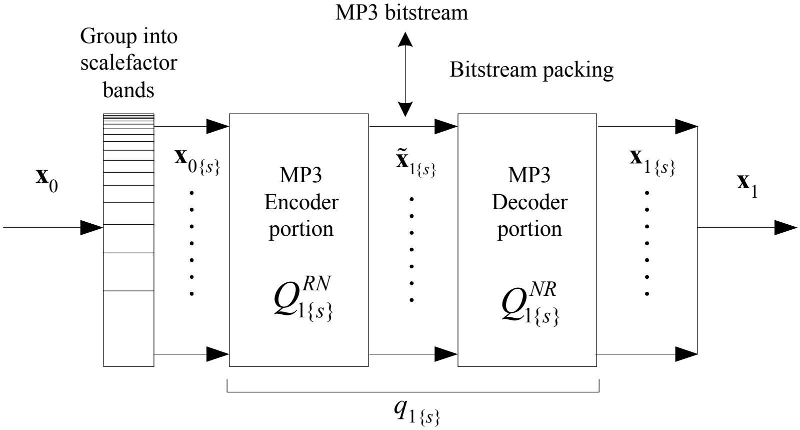


Figure 7-11 A block diagram showing the MP3 quantization scheme.

The MP3 quantization scheme (Figure 7-11) differs significantly from that of MPEG 1 Layer 2. The vector $\mathbf{x}_0$ consists of 576 subband samples $x_{0[m]}$ per unit sample time. A q-frame consists of 576 by 1 subband samples. These 576 subband samples are first grouped into scalefactor bands (22 for long windows and 13 for short windows). In the 'group into scalefactor bands' block, the horizontal lines pictorially show the allocated widths of the scalefactor bands (refer to Appendix C, Figure C-2 for more information). We denote the set of subband samples $x_{0[m]}$ that is grouped into the s-th scalefactor band as the vector $\mathbf{x}_{0\{s\}}$. As an illustration, for a long window:

$$\mathbf{x}_0 = \left[\mathbf{x}_{0\{0\}}{}^T \ \cdots \ \mathbf{x}_{0\{s\}}{}^T \ \cdots \ \mathbf{x}_{0\{21\}}{}^T \right]^T . \tag{7.11}$$

In MP3 quantization, which we represent using $q_{1\{s\}}$, $x_{0[m]}$ is quantized to $x_{1[m]}$ in the following manner. In the MP3 encoder, the subband samples $x_{0[m]}$ that are in the s-th scalefactor band (i.e. in the vector $\mathbf{x}_{0\{s\}}$), are scaled from the real-valued $x_{0[m]}$ to the integer-valued $\tilde{x}_{1[m]}$. We represent this scaling using the function $Q_{1\{s\}}^{RN}$ (*RN* indicates that the scaling takes place from a real value to an integer value):

$$\begin{aligned} \tilde{x}_{1[m]} &= Q_{1\{s\}}^{RN}\left(x_{0[m]}, \mathbb{Z}_{1\{s\}} \right) \\ &= \pm \Re\left[\left(2^{\mathbb{Z}_{1\{s\}}} \left| x_{0[m]} \right| \right)^{3/4} \right], \end{aligned} \tag{7.12}$$

where $\mathbb{Z}_{1\{s\}}$ is the lumped quantization parameter for the quantizer $q_{1\{s\}}$, $\mathbb{Z}_{1\{s\}}$ consists of the global gain, scalefactor band gain and encoder-specific gains (see Appendix C); $\Re(.)$ rounds the bracketed term to its nearest integer; and the sign of $\tilde{x}_{1[m]}$ is taken equal to the sign of $x_{0[m]}$.

A larger value of $\mathbb{Z}_{1\{s\}}$ scales $\mathbf{x}_{0\{s\}}$ to a larger range of values before rounding, leading to a better resolution of values (i.e. having smaller quantization errors). The set of $\tilde{x}_{1[m]}$ that is grouped into the s-th scalefactor band is denoted as the vector $\tilde{\mathbf{x}}_{1\{s\}}$. The integer-valued $\tilde{\mathbf{x}}_{1\{s\}}$ is packed into the MP3 bitstream.

In the MP3 decoder, the integer-valued $\tilde{x}_{1[m]}$ that are in the s-th scalefactor band are scaled to the real-valued $x_{1[m]}$. We represent this scaling using the function $Q_{1\{s\}}^{NR}$ (*NR* indicates that the scaling takes place from an integer value to a real value):

$$\begin{aligned} x_{1[m]} &= Q_{1\{s\}}^{NR}\left(\tilde{x}_{1[m]}, \mathbb{Z}_{1\{s\}} \right) \\ &= \pm \left| \tilde{x}_{1[m]} \right|^{4/3} 2^{-\mathbb{Z}_{1\{s\}}} \quad , \end{aligned} \tag{7.13}$$

where the sign of $x_{1[m]}$ is taken equal to the sign of $\tilde{x}_{1[m]}$.

The quantization from the vector $\mathbf{x}_0$ to the vector $\mathbf{x}_1$ can be visualized as applying a non-uniform power-law quantizer (which we represent as $q_{1\{s\}}$) to $\mathbf{x}_{0\{s\}}$ in each of the scalefactor bands, where the quantizer $q_{1\{s\}}$ is defined by the lumped quantization parameter $\mathbb{Z}_{1\{s\}}$. If we assume that the inputs $\mathbf{x}_{0\{s\}}$ to the quantizer range between -1 and 1, then the effective number of quantization steps is $2\left(2^{3\mathbb{Z}_{1[s]}/4}\right)+1$. Figure 7-10(b) illustrates the quantizer characteristics for the case of $\mathbb{Z}_{1\{s\}}$=4.

Figure 7-12 shows the absolute values of $\tilde{\mathbf{x}}_1$ in a typical q-frame which is encoded using the MP3 method. We observe that in frequency regions where the human auditory system is more sensitive to noise (i.e. $0 \le m \le 100$), the subband samples tend to be scaled to larger values, so that the quantization introduces smaller errors. On the other hand, for the mid-to-high frequency region, quantization is usually very coarse and typically involves only one or two quantization steps.

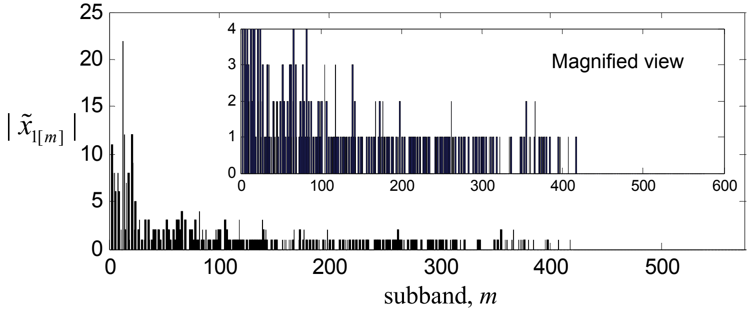


Figure 7-12 Plot of $|\tilde{\mathbf{x}}_1|$ for a typical MP3 q-frame.

7.6 Outline of our work

In this chapter, we introduced some of the concepts behind audio coding and transcoding. Many audio compression methods are available, and we focused mainly on the MPEG 1 Layer 2 and MP3 methods. We used a cascaded quantization model to describe the transcoding process, and important concepts such as sample-synchronization and frame-synchronization were defined.

For Part II of this thesis, our work is summarized as follows. In Chapter 8, we focus on the analysis of tandem quantization error, which occurs during sample-synchronized transcoding. In Chapter 9, we investigate the impact of sample-synchronization on the audio quality of transcoded material, for the MPEG 1 Layer 2 and MP3 methods. In Chapter 10, an ultra-fast MP3 transcoder is proposed, implemented and tested.

Chapter 8

Tandem Quantization Error

In the previous chapter, we used the cascaded quantization model to describe the audio transcoding process. When the transcoding is sample-synchronized, the outputs of the transcoder synthesis-analysis block were shown to be delayed versions of the inputs. In [83], the effect of transcoding on audio quality was studied, and different combinations of transcoding methods (sample-synchronization vs. non-sample-synchronization, different compression methods, etc.) were tested. It is known that a major problem of transcoding is the potentially significant degradation of audio quality in the transcoded material.

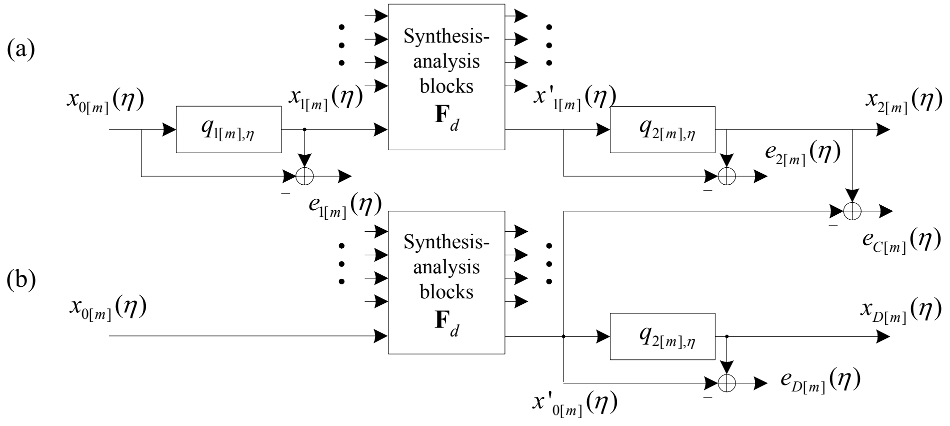
Furthermore, it is known that for the MPEG 1 Layer 2 method, audio which is transcoded sample-synchronously tends to be of higher quality than audio which is transcoded non-sample-synchronously. Thus, a method of 'improved' transcoding ([84]-[85]) is to estimate or calculate the input-to-output delay of a transcoder and then adding to the delay, such that the total delay is a multiple of the number of subbands M.

In this chapter, we analyze the effect of sample-synchronized cascaded quantization on the error of the output relative to the input. Our work is different from previous works, in the sense that our study focuses on the quantizers that are used, and the solutions address the transcoding problem by directly addressing these quantizers. The scope of our work in this chapter covers only uniform quantizers, such as those that are used in MPEG 1 Layer 2. Here, we reduce the audio transcoding problem to the general problem of a cascaded pair of quantizers. Note that our approach to the problem of cascaded quantization, and the methods and results presented, is not necessarily restricted to only audio transcoding.

8.1 Definition of tandem quantization error

8.1.1 Sample-synchronized transcoding

The cascaded quantization model was first defined in Section 7.3, and we repeat Figure 7-5 here for convenience. The variables that are listed in the diagram were explained earlier. In this chapter, we hide the sample-time η in our equations for convenience.



(Repeat of Figure 7-5.)

When the transcoding is sample-synchronized, $x'_{1[m]} = x_{1[m]}$ up to a delay. The outputs $x_{2[m]}$ are related to the inputs $x_{0[m]}$ by 2 successive quantizations:

$$x_{2[m]} = q_{2[m]}\left(q_{1[m]}(x_{0[m]})\right). \tag{8.1}$$

The synthesis-analysis blocks can be removed, and we redraw the diagram as shown in Figure 8-1. The variables in the diagram were explained in Section 7.3.

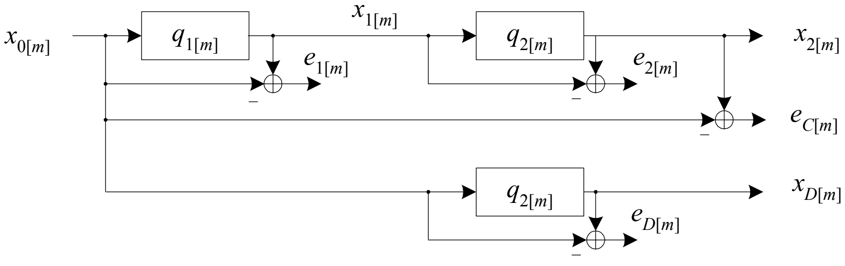
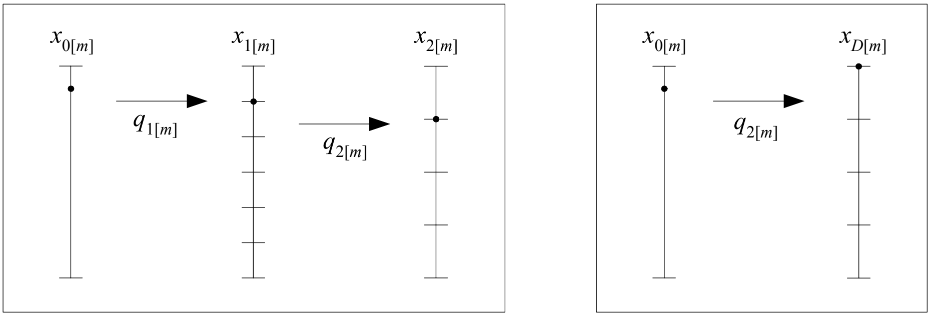


Figure 8-1 Cascaded quantization model for sample-synchronized transcoding.

8.1.2 *Illustration of tandem quantization error effect*

Figure 8-2(a) shows a sample $x_{0[m]}$ which is quantized by a cascade of two quantizers represented by $q_{1[m]}$ and $q_{2[m]}$. Figure 8-2(b) shows the sample $x_{0[m]}$ which is quantized directly by $q_{2[m]}$. In this particular example, $q_{1[m]}$ has 7 quantization steps, $q_{2[m]}$ has 5 quantization steps, and the value of $x_{2[m]} = q_{2[m]}\left[q_{1[m]}(x_{0[m]})\right]$ is not equal to the value of $x_{D[m]} = q_{2[m]}(x_{0[m]})$. For the illustrated value of $x_{0[m]}$, the cascaded quantization case results in a larger error than the direct quantization case.



(a) Cascaded quantization. (b) Direct quantization.

Figure 8-2 Illustration of tandem quantization error effect.

8.1.3 *Some definitions*

Directly quantizing $x_{0[m]}$ with $q_{2[m]}$, the quantization error power is given by $e^2_{D[m]}$. The relative quantization error power of $q_{2[m]}(x_{0[m]})$ is defined as the quantization error power divided by the average squared quantization step-size:

$$e^2_{RD[m]} = \frac{e^2_{D[m]}}{\overline{\Delta}^2_{2[m]}}, \tag{8.2}$$

where $\overline{\Delta}^2_{2[m]}$ is the average squared quantization step-size of $q_{2[m]}$. The average squared quantization step-size can be found by summing the square of the individual quantization step-sizes for the dynamic range of the quantizer, and then dividing by the number of quantization steps. For a uniform quantizer, the quantization step-sizes are constant, and the average squared quantization step-size is thus equal to the square of the individual quantization step-size, $\overline{\Delta}^2_{2[m]} = \Delta^2_{2[m]}$.

Tandem noise power: The tandem noise power $P_{T[m]}$ is defined as the difference between the error power of $x_{2[m]}$ and the error power of $x_{D[m]}$:

$$\begin{aligned} P_{T[m]} &= \left(x_{2[m]} - x_{0[m]}\right)^2 - \left(x_{D[m]} - x_{0[m]}\right)^2 \\ &= e^2_{C[m]} - e^2_{D[m]}. \end{aligned} \tag{8.3}$$

The relative tandem noise power is defined as the tandem noise power divided by the average squared quantization step-size of $q_{2[m]}$:

$$P_{RT[m]} = \frac{P_{T[m]}}{\overline{\Delta}^2_{2[m]}}. \tag{8.4}$$

For the case when the tandem noise power is equal to zero, the error in $x_{2[m]}$ is solely due to $q_{2[m]}$, (i.e. $e^2_{C[m]} = e^2_{D[m]}$). This case is considered an ideal cascaded quantization; the cascaded quantization using $q_{1[m]}$ followed by $q_{2[m]}$, does not change the result as compared to direct quantization using $q_{2[m]}$. As an example, for the case when $q_{1[m]}=q_{2[m]}$, $x_{2[m]}=x_{1[m]}$ and $P_{T[m]}=0$.

Tandem indicator: The 'tandem indicator' is a ternary value, which serves to indicate whether tandem noise power is present, and whether tandem quantization error is positive or negative, for a combination of $q_{1[m]}$, $q_{2[m]}$, and $x_{0[m]}$. The values are defined to be:

$$I_{T[m]} = \begin{cases} 1, & \text{if}\left(e_{C[m]} - e_{D[m]}\right) > 0 \\ 0, & \text{if}\left(e_{C[m]} - e_{D[m]}\right) = 0 \\ -1, & \text{if}\left(e_{C[m]} - e_{D[m]}\right) < 0 \end{cases} . \tag{8.5}$$

When $I_{T[m]}$ is zero, tandem noise power is zero. When $I_{T[m]}$ is non-zero, tandem noise power is positive.

Tandem quantization error: The tandem quantization error $e_{T[m]}$ is defined to be:

$$e_{T[m]} = I_{T[m]}\sqrt{P_{T[m]}} \, . \tag{8.6}$$

8.2 Tandem quantization error regions

The tandem quantization error that is incurred depends on the value of $x_{0[m]}$, and on the relationship between the quantizers $q_{1[m]}$ and $q_{2[m]}$. We define tandem quantization error regions as the range of values of $x_{0[m]}$ for a set of $q_{1[m]}$ and $q_{2[m]}$ that lead to tandem quantization error, i.e. $I_{T[m]} \neq 0$. In this chapter, we analyze tandem quantization error for the MPEG 1 Layer 2 method. It is assumed that the input signal has a minimum value of -1 and a maximum value of 1 ($x_{0[m]} \in [-1,1]$), and that the quantizers are uniform.

For $q_{1[m]}$ and $q_{2[m]}$, let us denote the number of quantization steps by $N_{1[m]}$ and $N_{2[m]}$, and the quantization step-sizes by $\Delta_{1[m]}$ and $\Delta_{2[m]}$ respectively. For the quantizers used in the MPEG 1 Layer 2 method:

$$\Delta_{[m]} = 2 / N_{[m]} \, . \tag{8.7}$$

The number of tandem quantization error regions $N_{T[m]}$ is limited by the number of quantization steps for $q_{2[m]}$:

$$N_{T[m]} \leq N_{2[m]} - 1 \, . \tag{8.8}$$

For illustrative purposes, let us assume that $q_{1[m]}$ has 7 quantization steps and $q_{2[m]}$ has 5 quantization steps.

8.2.1 Analysis for a 7-step/5-step quantizer pair

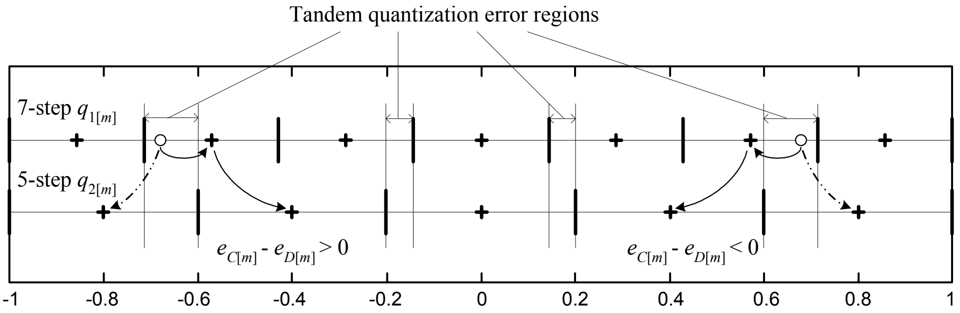

Tandem quantization error regions
7-step $q_{1[m]}$
5-step $q_{2[m]}$
$e_{C[m]} - e_{D[m]} > 0$
$e_{C[m]} - e_{D[m]} < 0$
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

Figure 8-3 Quantizer characteristics and tandem quantization error regions for the 5-step and 7-step quantizers.

Figure 8-3 shows a one-dimensional plot of the quantization characteristics for $q_{1[m]}$ and $q_{2[m]}$. In our figure, the bold vertical bars mark the quantization boundaries and the crosses mark the quantization steps for a quantizer $q_{[m]}$: samples having values which are between 2 adjacent bold vertical bars are quantized to the nearest value which is marked by a cross.

For a pair of $q_{1[m]}$ and $q_{2[m]}$, if all the quantization boundaries of $q_{2[m]}$ fall on the same values as the quantization boundaries of $q_{1[m]}$, then $N_{T[m]}$=0. If some of the quantization boundaries of $q_{2[m]}$ fall on the same values as the quantization boundaries of $q_{1[m]}$, then $N_{T[m]}<N_{2[m]}-1$. If none of the quantization boundaries of $q_{2[m]}$ fall on the same values as the quantization boundaries of $q_{1[m]}$, then $N_{T[m]}=N_{2[m]}-1$.

For the 7-step/5-step quantizer pair shown in Figure 8-3, there exist $N_{T[m]}$=4 tandem quantization error regions. For values of $x_{0[m]}$ which fall within these regions, tandem quantization error is not zero, i.e. $q_{2[m]}\left[q_{1[m]}(x_{0[m]})\right] \neq q_{2[m]}(x_{0[m]})$ and $e_{T[m]} \neq 0$. For values of $x_{0[m]}$ falling outside the tandem quantization error regions, tandem quantization error is zero, i.e. $q_{2[m]}\left[q_{1[m]}(x_{0[m]})\right] = q_{2[m]}(x_{0[m]})$ and $e_{T[m]} = 0$. In the figure, examples are shown where $x_{0[m]}$, which are marked by circles, fall within the tandem quantization error regions. The path taken when $x_{0[m]}$ is quantized by $q_{2[m]}$ is indicated by a dotted arrow, and the path taken when $x_{0[m]}$ is quantized by $q_{1[m]}$ followed by $q_{2[m]}$ is indicated by solid arrows.

If there are zero tandem quantization error regions for a pair of $q_{1[m]}$ and $q_{2[m]}$,(e.g. for the case of $q_{1[m]}=q_{2[m]}$) or alternatively, if all values of $x_{0[m]}$ fall outside the tandem quantization error regions for a pair of $q_{1[m]}$ and $q_{2[m]}$, then the cascaded quantization is ideal. In such a case, $e_{T[m]}$=0 and $P_{T[m]}$=0. In our analysis, we focus on values of $x_{0[m]}$ that fall within the tandem quantization error regions.

Let us denote the set of quantization boundaries of $q_{1[m]}$, $q_{2[m]}$ (marked by bold vertical bars) respectively by $\Phi_{1[m]}$, $\Phi_{2[m]}$ and the set of quantization steps (marked by crosses) by $\Theta_{1[m]}$, $\Theta_{2[m]}$. For example, for the MPEG 1 Layer 2 method, a 7-step $q_{1[m]}$ has the following parameters:

$$\Phi_{1[m]} = \{\text{-0.7143} \quad \text{-0.4286} \quad \text{-0.1429} \quad 0.1429 \quad 0.4286 \quad 0.7143\}\text{, and}$$

$$\Theta_{1[m]} = \{\text{-0.8571} \quad \text{-0.5714} \quad \text{-0.2857} \quad 0 \quad 0.2857 \quad 0.5714 \quad 0.8571\}.$$

A 5-step $q_{2[m]}$ has the following parameters:

$$\Phi_{2[m]} = \{\text{-0.6} \quad \text{-0.2} \quad 0.2 \quad 0.6\}\text{, and}$$

$$\Theta_{2[m]} = \{\text{-0.8} \quad \text{-0.4} \quad 0 \quad 0.4 \quad 0.8\}.$$

The tandem quantization error regions can be determined from the quantization boundaries $\Phi_{1[m]}$ and $\Phi_{2[m]}$. Let us denote the set of tandem quantization error regions by $\mathbf{R}_{[m]}$. A simple rule for finding $\mathbf{R}_{[m]}$ is to take the set of values in $\Phi_{2[m]}$ and pair them with a subset of the values in $\Phi_{1[m]}$, such that the values for each pair have the least absolute difference. In our example, the 4 tandem quantization error regions are (see Figure 8-3):

$$\mathbf{R}_{[m]} = \begin{cases} R_{[m],1} \in [-0.714, -0.6], & R_{[m],2} \in [-0.2, -0.143] \\ R_{[m],3} \in [0.143, 0.2], & R_{[m],4} \in [0.6, 0.714]. \end{cases}$$

The lower and upper bounds of $x_{0[m]}$ for each tandem quantization error region are denoted by $R^L_{[m]}$ and $R^U_{[m]}$ respectively, e.g. $R_{[m],1} \in \left[R^L_{[m],1}, R^U_{[m],1}\right]$.

8.2.2 Tandem indicators

For values of $x_{0[m]}$ falling within $\mathbf{R}_{[m]}$, a plot of the tandem indicators for the 7-step/5-step pair is shown in Figure 8-4.

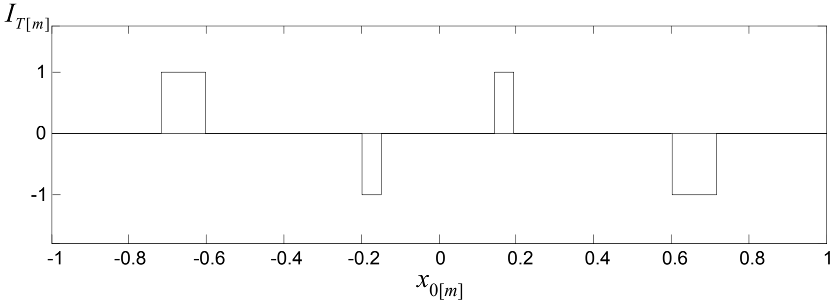


Figure 8-4 Plot of tandem indicators for a 7-step/5-step quantizer pair.

8.2.3 Tandem noise power

For values of $x_{0[m]}$ falling within $\mathbf{R}_{[m]}$, the tandem noise power is:

$$\begin{aligned}P_{T[m]} &= e^2_{C[m]} - e^2_{D[m]} \\ &= \left(q_{2[m]}\left[q_{1[m]}(x_{0[m]})\right]\right)^2 - \left(q_{2[m]}(x_{0[m]})\right)^2 \\ &\qquad +2x_{0[m]}\left(q_{2[m]}(x_{0[m]}) - q_{2[m]}\left[q_{1[m]}(x_{0[m]})\right]\right) \\ &= \alpha + \beta x_{0[m]},\end{aligned} \tag{8.9}$$

where α and β are constant for values of $x_{0[m]}$ falling within each tandem quantization error region:

$$\alpha = \left(q_{2[m]}\left[q_{1[m]}(x_{0[m]})\right]\right)^2 - \left(q_{2[m]}(x_{0[m]})\right)^2, \text{ and} \tag{8.10}$$

$$\beta = 2\left(q_{2[m]}(x_{0[m]}) - q_{2[m]}\left[q_{1[m]}(x_{0[m]})\right]\right). \tag{8.11}$$

From (8.9), we observe that the tandem noise power varies in a linear-piecewise fashion with $x_{0[m]}$. As an illustration, for $N_{1[m]}=7$, $N_{2[m]}=5$, and values of $x_{0[m]}$ falling within $R_{[m],4}$ (i.e. $x_{0[m]} \in R_{[m],4}$), the tandem noise power is given by:

$$\begin{aligned}P_{T[m],x_{0[m]}\in R_{[m],4}} &= (0.4 - x_{0[m]})^2 - (0.8 - x_{0[m]})^2 \\ &= \alpha + \beta x_{0[m]},\end{aligned} \tag{8.12}$$

where α=-0.48 and β=0.8.

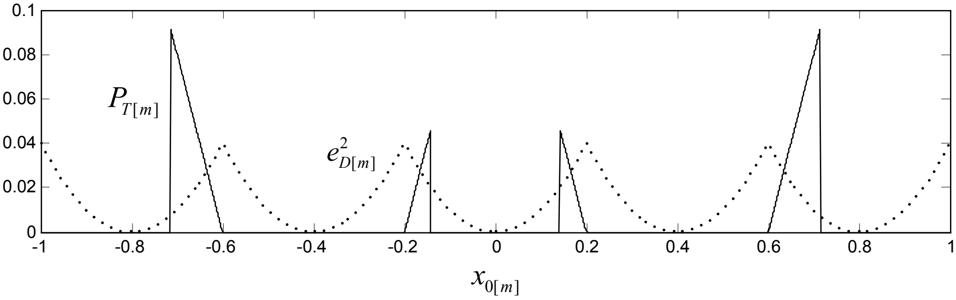


Figure 8-5 Plot of tandem noise power for a 7-step/5-step quantizer pair.

The solid line in Figure 8-5 shows the tandem noise power $P_{T[m]}$ for the range of values $-1 < x_{0[m]} < 1$, and the 7-step/5-step quantizer pair with input $x_{0[m]}$ and output $x_{2[m]}$. The dotted line shows the quantization error power $e^2_{D[m]}$ when $x_{0[m]}$ is directly quantized to $x_{D[m]}$ by the 5-step quantizer. It is clear that for certain values of $x_{0[m]}$, the tandem noise power can be significant when compared to $e^2_{D[m]}$.

8.3 Mean tandem noise power

When audio is encoded directly to bit-rate B_2, we model the process as a direct quantization of the subband signal $x_{0[m]}$ with $q_{2[m]}$ for each q-frame. The quantization error can be calculated and the mean quantization error power for subband m can be found from $E\left[e_{D[m]}^2\right]$.

When audio is encoded to bit-rate B_1, and then transcoded (sample-synchronized) to bit-rate B_2, we model the process as a cascaded quantization of the subband signal $x_{0[m]}$ with $q_{1[m]}$ and $q_{2[m]}$ for each q-frame. The mean quantization error power for subband m can be found from:

$$E\left[e_{C[m]}^2\right] = E\left[e_{D[m]}^2\right] + E\left[P_{T[m]}\right]. \tag{8.13}$$

Both the instantaneous and mean quantization error powers provide an objective indication of the quality of the encoded or transcoded material. Short-term, large spikes in the quantization error power can be highly audible to the human ear. Long-term, mean quantization error power provides a reasonably objective (though not fully perceptually accurate) estimation of the audio quality. For low bit-rates, coarse quantizers are used, which result in a larger mean quantization error power and generally lower audio quality. On the other hand, for high bit-rates, fine quantizers are used, which result in a smaller mean quantization error power and generally higher audio quality.

For the transcoded audio at bit-rate B_2, the instantaneous and mean tandem noise power provide an objective indication to its audio quality. For the ideal case, tandem noise power is zero, and the transcoded audio is equal in audio quality to the directly encoded audio at bit-rate B_2.

8.3.1 *Mean direct quantization error power*

Consider the uniform quantizer represented by $q_{2[m]}$ with N_2 steps. The quantization step-size is $\Delta_{2[m]} = 2 / N_{2[m]}$. If we assume that the inputs to the quantizer are uniformly distributed between -1 and 1, it is well-known that the error can be approximated using a uniform distribution with zero mean and variance $\Delta_{2[m]}^2 / 12$. The error power incurred by the quantization $x_{D[m]} = q_{2[m]}(x_{0[m]})$ is:

$$E\left[e_{D[m]}^2\right] = E\left[\left(x_{2[m]} - x_{0[m]}\right)^2\right] \approx \frac{\Delta_{2[m]}^2}{12}. \tag{8.14}$$

8.3.2 *Signal distributions*

We would like to highlight that for actual audio inputs, $x_{0[m]}$ does not necessarily follow any specific distribution. The actual distribution of $x_{0[m]}$ depends on the nature and content of the audio material, as well as on the encoding method. The encoder may modify the subband samples in a way that affects its distribution, e.g. normalization scaling of the subband samples in the MPEG 1 Layer 2 method. It is observed that in most cases, for the MPEG 1 Layer 2 method, there is a tendency for a larger concentration of values towards the zero value and a smaller concentration of values towards the -1 and +1 extremities.

In this chapter, we would like to address the problem of cascaded quantization for a general scenario, and for a wide range of possible signal distributions. For this purpose, we do not restrict the methods and results presented here to a specific signal distribution. Instead, as an illustrative tool to show the effectiveness of our methods, we simply assume a uniformly distributed $x_{0[m]}$ for the analysis of tandem quantization error. In Section 8.8, we further consider the implications for signal distributions that are more typical for audio transcoding.

8.3.3 *Mean tandem noise power*

For an input signal $x_{0[m]} \in [-1,1]$, the mean tandem noise power can be found if we know its probability density function:

$$E\left[P_{T[m]}\right] = \sum_{a=1}^{N_{T[m]}} \left[\int_{R^L_{[m],a}}^{R^U_{[m],a}} P_{T[m]}(x_{0[m]}).p(x_{0[m]})dx_{0[m]} \right], \tag{8.15}$$

where $p(x_{0[m]})$ is the probability density function of $x_{0[m]}$. As an example, for a uniform distribution of $x_{0[m]} \in [-1,1]$, this equation can be simplified to:

$$E\left[P_{T[m]}\right] = 0.5 \sum_{a=1}^{N_{T[m]}} \left[\alpha\left(R^U_{[m],a} - R^L_{[m],a}\right) + 0.5\beta\left(R^U_{[m],a} - R^L_{[m],a}\right)^2 \right]. \tag{8.16}$$

8.4 Effect of different cascade-pair combinations

Figure 8-6 shows the tandem noise power (given by $P_{T[m]}$, which is shown by a solid line) for various combinations of the cascaded pair $q_{1[m]}$ and 5-step $q_{2[m]}$. The error power for a direct quantization using $q_{2[m]}$ (given by $e^2_{D[m]}$) is shown by a dotted line. It is noted that for the 15-step $q_{1[m]}$ (which we did not show), there is zero tandem quantization error and tandem noise power because the quantization boundaries of the 15-step quantizer are aligned with the quantization boundaries of the 5-step quantizer. It can be observed that as the difference in the number of quantization steps between $q_{1[m]}$ and $q_{2[m]}$ increases, the widths of the tandem quantization error regions generally decrease.

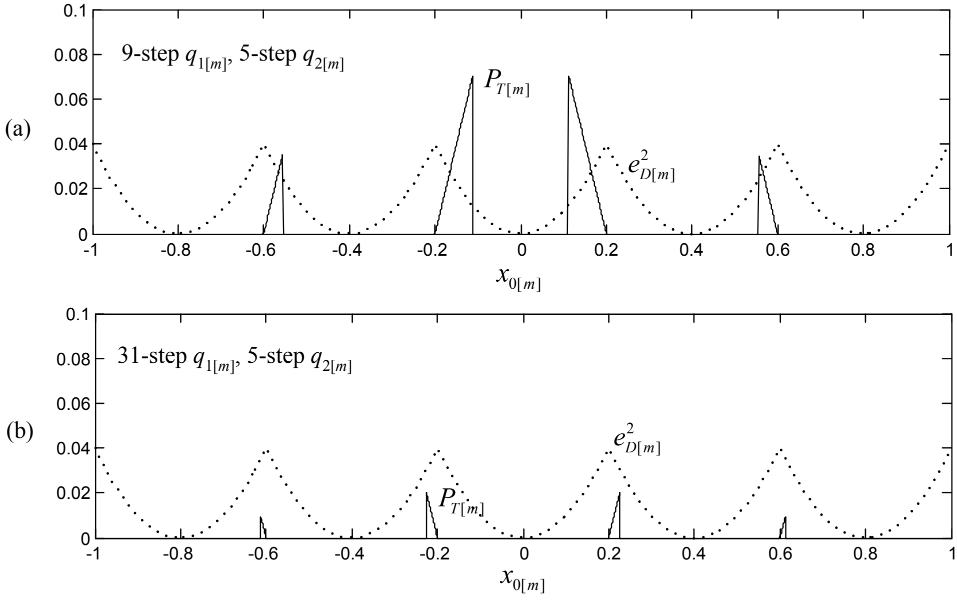


Figure 8-6 Tandem noise power for a 5-step $q_{2[m]}$ and various $q_{1[m]}$.

When the difference in the number of quantization steps between $q_{1[m]}$ and $q_{2[m]}$ is sufficiently large, $P_{T[m]}$ becomes very small compared to $e^2_{D[m]}$. From this result, it can be inferred that when transcoding (sample-synchronized) from bit-rate B_1 to bit-rate B_2, the larger the difference between B_1 and B_2, the smaller the amount of tandem noise power compared to $e^2_{D[m]}$. Consequently, the audio quality of the transcoded material approaches that of the quality of material that is directly encoded from the source to bit-rate B_2.

8.5 Overview of proposed methods to reduce tandem noise power

In the next 2 sections (Sections 8.6-8.7), we consider methods to reduce tandem noise power for a cascaded quantization process, in the context of audio transcoding. Cascaded quantization effects in the time domain PCM signal from a larger wordlength (finer quantization) to a smaller wordlength (coarser quantization) have been well-studied [86]. Usually, a small amount of uncorrelated noise known as dither is added to the signal prior to the second quantization [87]-[88]. Dithering methods are effective for improving the audio quality of PCM signals having long time-segments of similar values. These PCM signals are usually quantized to many quantization steps, e.g. using 16 bits. Since we consider quantizers with very few quantization steps in this chapter, the noise added by dithering becomes very large and easily audible.

In [89], audio transcoding from material encoded using compression method A to material that is encoded using a different compression method B (at a similar bit-rate) is considered. To improve the audio quality of the transcoded material, the second quantizer (compression method B) is matched, in terms of having a similar bit-rate, to the first quantizer (compression method A), i.e. $q_{2[m]} \approx q_{1[m]}$.

For audio transcoding from a higher bit-rate to a lower bit-rate, so that $q_{2[m]}$ has fewer quantization steps than $q_{1[m]}$, the typical approach is to embed additional information in the initial encoding ([93]-[100], also see Section 10.1.1 for a more detailed explanation). Another approach (further explained in Chapter 9) is to enforce sample-synchronization (such as for MPEG 1 Layer 2) or non-sample-synchronization (such as for MP3).

Our approach considers the problem of transcoding, in terms of audio quality, at its fundamental level: the quantization process. In Section 8.6, we consider a method that modifies the quantization characteristics of $q_{1[m]}$. The method assumes an a-priori knowledge of $q_{2[m]}$. Further discussions on the validity of this assumption are provided in Section 8.6.5.

In Section 8.7, we consider a method that selects $q_{2[m]}$ from a range of possible quantizers. Given the available $x_{1[m]}$ (signal entering the transcoder) and $q_{1[m]}$, the method estimates the likely range of values of $x_{0[m]}$ and selects $q_{2[m]}$ that minimizes the resultant tandem noise power.

8.6 Modified quantizer method

8.6.1 Modification of initial quantization

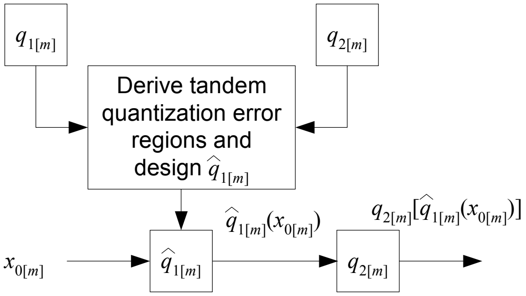

$q_{1[m]}$
$q_{2[m]}$
Derive tandem quantization error regions and design $\hat{q}_{1[m]}$
$\hat{q}_{1[m]}(x_{0[m]})$
$q_{2[m]}[\hat{q}_{1[m]}(x_{0[m]})]$
$x_{0[m]}$
$\hat{q}_{1[m]}$
$q_{2[m]}$

Figure 8-7 Proposed method for reducing tandem noise power.

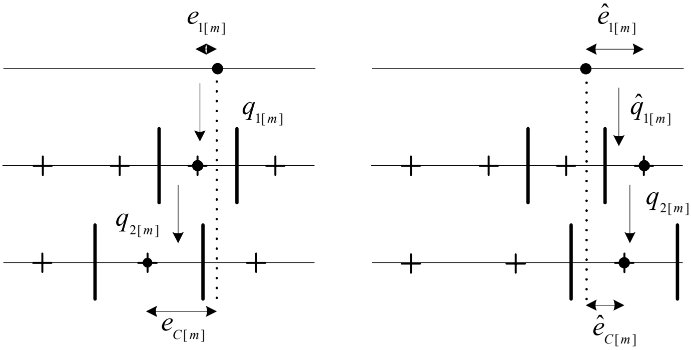
Figure 8-7 shows the proposed method for the reduction of tandem noise power. If we assume that the quantizer pair $q_{1[m]}$ and $q_{2[m]}$ is known (i.e. that we have a-priori knowledge of the quantizer represented by $q_{2[m]}$, which is used in the transcoder), then we are able to derive the tandem quantization error regions, and the corresponding tandem noise power for different values of $x_{0[m]}$. Suppose that for a value of $x_{0[m]}$ falling within the tandem quantization error region, we apply a modified non-uniform quantizer $\hat{q}_{1[m]}$ such that $x_{0[m]}$ is quantized in the opposite direction instead of the usual direction $q_{1[m]}(x_{0[m]})$. Then, the quantization error power introduced in the first quantization is:

$$\hat{e}^2_{1[m]} = \left[\hat{q}_{1[m]}(x_{0[m]}) - x_{0[m]}\right]^2 . \tag{8.17}$$

This value is larger than the quantization error power $e^2_{1[m]}$ incurred in the original quantization $q_{1[m]}(x_{0[m]})$. However, if we consider the second quantization using $q_{2[m]}$, the tandem quantization error is negated ($e_{T[m]}$=0) :

$$q_{2[m]}\left[\hat{q}_{1[m]}(x_{0[m]})\right] = q_{2[m]}(x_{0[m]}) . \tag{8.18}$$

An example of this modified quantization method for an illustrative value of $x_{0[m]}$ falling within the tandem quantization error region is shown in Figure 8-8(b). Compare this with the normal quantization method shown in Figure 8-8(a). The trade-off in using a modified $\hat{q}_{1[m]}$ is a decreased cascaded quantization error power ($\hat{e}^2_{C[m]} < e^2_{C[m]}$), at the expense of an increased quantization error power at the output of $q_{1[m]}$ ($\hat{e}^2_{1[m]} > e^2_{1[m]}$). The increased quantization error power at the output of $\hat{q}_{1[m]}$ (i.e. the quality of the audio material encoded at bit-rate B_1 is not as good as it would be when encoded normally) might or might not be acceptable, depending on the application.



(a) Normal quantization (b) Modified quantization

Figure 8-8 Illustration of a modified quantization method, where $q_{1[m]}$ is replaced with $\hat{q}_{1[m]}$.

Another possible implementation of $\hat{q}_{1[m]}$ is to divide the tandem quantization error region into two sub-regions. Values of $x_{0[m]}$ falling into the first region are quantized normally and those falling in the second region are quantized in the opposite direction. Figure 8-9 shows the division of the tandem quantization error region into 2 equal halves. Values of $x_{0[m]}$ that fall within the shaded tandem quantization error regions are quantized in the opposite direction using $\hat{q}_{1[m]}$. On the other hand, values of $x_{0[m]}$ that fall within the non-shaded tandem quantization error regions or outside of the tandem quantization error regions are quantized normally using $q_{1[m]}$.

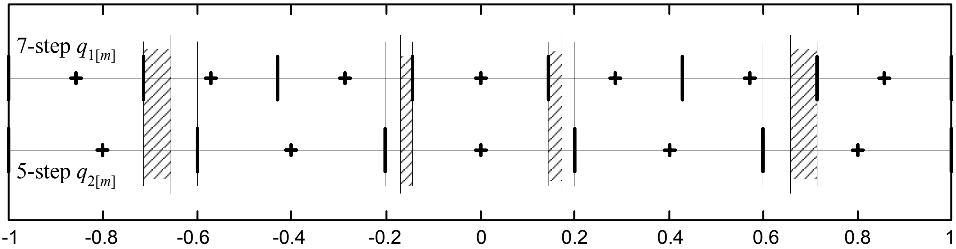


Figure 8-9 Dividing each of the tandem quantization error regions into 2 half-parts.

The implementation of $\hat{q}_{1[m]}$ can be possibly varied by defining other split-points for the division of the tandem quantization error regions. The modified quantizer $\hat{q}_{1[m]}$ might also be varied over time and subband *m*, depending on psychoacoustic conditions.

In Section 8.6.5, we further discuss the validity of the assumption of a-priori knowledge of $q_{2[m]}$, and we also propose a nearest neighbours method when this knowledge is unavailable.

8.6.2 *Dividing the tandem quantization error regions*

The tandem quantization error regions can be divided to varying proportions (instead of simply into halves), to offer a tradeoff between the quantization error power at the output of the first quantizer and the cascaded quantization error power at the output of the second quantizer.

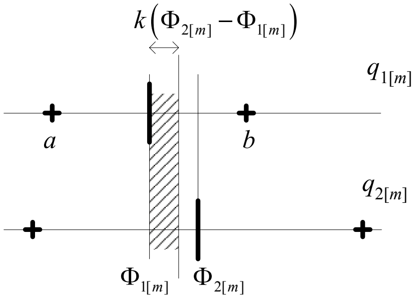


Figure 8-10 Diagram illustrating method to divide tandem quantization error region.

With reference to Figure 8-10, let a sample $x_{0[m]}$ that falls within the shaded region between $\Phi_{1[m]}$ and $k\left(\Phi_{2[m]}-\Phi_{1[m]}\right)$, $0 \leq k \leq 1$, be quantized to the quantization step '*a*' using $q_{1[m]}$. Let a sample $x_{0[m]}$ that falls within the non-shaded region between $k\left(\Phi_{2[m]}-\Phi_{1[m]}\right)$ and $\Phi_{2[m]}$ be quantized to the quantization step '*b*' using $q_{1[m]}$.

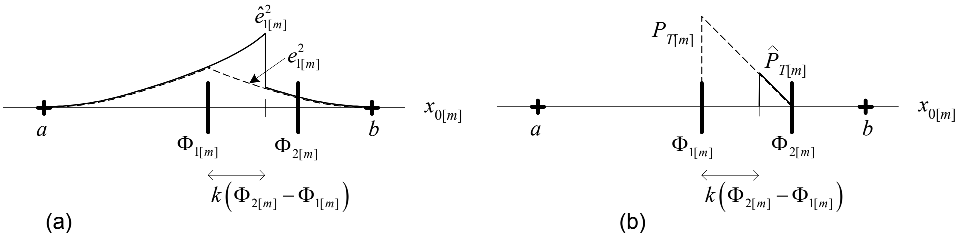


Figure 8-11 Effect of dividing the tandem quantization error region on the error powers.

In Figure 8-11(a), the dotted line shows the quantization error power for different values of $x_{0[m]}$ when quantized with $q_{1[m]}$. The solid line shows the quantization error power when $x_{0[m]}$ falling within the shaded region are quantized in the opposite direction (using $\hat{q}_{1[m]}$). In Figure 8-11(b), the dotted line shows the tandem noise power for different values of $x_{0[m]}$ when quantized with the cascade $q_{1[m]}$ and $q_{2[m]}$. The solid line shows the tandem noise power when $x_{0[m]}$ falling within the shaded region are quantized using $\hat{q}_{1[m]}$.

Assuming uniform distribution of $x_{0[m]}$, for $x_{0[m]} \in \left[\Phi_{1[m]}, \Phi_{2[m]}\right]$, we can calculate (see Figure 8-11(b)):

$$\begin{aligned} E\left[\hat{P}_{T[m]}\right] &= \frac{1}{\Phi_{2[m]}-\Phi_{1[m]}}\left[\frac{1}{2}(1-k)^2\left(\Phi_{2[m]}-\Phi_{1[m]}\right)P_{T[m],\max}\right] \\ &= \frac{1}{2}(1-k)^2 P_{T[m],\max} \quad , \end{aligned} \tag{8.19}$$

where the equation in square brackets on the right-hand side is the area of the solid triangle, and $P_{T[m],\max}$ is found by taking the value of $P_{T[m]}$ when $x_{0[m]} = \Phi_{1[m]}$.

Furthermore:

$$\begin{aligned} E\left[\hat{e}^2_{1[m]}\right] &= \frac{1}{\Phi_{2[m]}-\Phi_{1[m]}}\int_0^{k\left(\Phi_{2[m]}-\Phi_{1[m]}\right)}\left(x_{0[m]}+\frac{\Delta_{1[m]}}{2}\right)^2 dx_{0[m]} \\ &+ \frac{1}{\Phi_{2[m]}-\Phi_{1[m]}}\int_{k\left(\Phi_{2[m]}-\Phi_{1[m]}\right)}^{\Phi_{2[m]}-\Phi_{1[m]}}\left(x_{0[m]}-\frac{\Delta_{1[m]}}{2}\right)^2 dx_{0[m]} \\ &= C + k^2\Delta_{1[m]}\left(\Phi_{2[m]}-\Phi_{1[m]}\right), \end{aligned} \tag{8.20}$$

where $\Delta_{1[m]}$ is the quantization step size of $q_{1[m]}$, and:

$$C = \frac{\left(\Phi_{2[m]}-\Phi_{1[m]}\right)^2}{3} - \frac{\Delta_{1[m]}\left(\Phi_{2[m]}-\Phi_{1[m]}\right)}{2} + \frac{\Delta^2_{1[m]}}{4}. \tag{8.21}$$

Let us consider the relationships between k, $E\left[\hat{e}^2_{1[m]}\right]$ and $E\left[\hat{P}_{T[m]}\right]$. Define the ratio:

$$R = E\left[\hat{P}_{T[m]}\right] \Big/ E\left[\hat{e}^2_{1[m]}\right], \tag{8.22}$$

which can be expanded by substituting from (8.19) and (8.20). Note that $0 \le k \le 1$, and $0 \le R \le P_{T[m],\max} / 2C$.

It can be seen that as k increases, $E\left[\hat{e}^2_{1[m]}\right]$ increases, $E\left[\hat{P}_{T[m]}\right]$ decreases, and hence R decreases. Working backwards, we can also obtain k from R, by way of the second-order equation:

$$k^2\left(\frac{P_{T[m],\max}}{2} - \Delta_{1[m]}R\left(\Phi_{2[m]} - \Phi_{1[m]}\right)\right) - kP_{T[m],\max} + \left(\frac{P_{T[m],\max}}{2} - RC\right) = 0, \tag{8.23}$$

which can be easily solved.

In a typical transcoding scenario, it is conceivable that we might want to make the tandem noise power after transcoding as small as possible, subject to the condition that the quantization error power after the first encoding does not exceed the amount E_0, i.e. minimize $E\left[\hat{P}_{T[m]}\right]$ such that $E\left[\hat{e}^2_{1[m]}\right] \le E_0$. Since $E\left[\hat{e}^2_{1[m]}\right]$ increases monotonically with k, and $E\left[\hat{P}_{T[m]}\right]$ decreases monotonically with k, the solution is straightforward. We can solve for k by substituting $E\left[\hat{e}^2_{1[m]}\right] = E_0$ into (8.20).

Conversely, we might want to make the quantization error power after the first encoding as small as possible, subject to the condition that the tandem noise power after transcoding does not exceed the amount P_0, i.e. minimize $E\left[\hat{e}^2_{1[m]}\right]$ such that $E\left[\hat{P}_{T[m]}\right] \le P_0$. The solution for k can be obtained by substituting $E\left[\hat{P}_{T[m]}\right] = P_0$ into (8.19).

8.6.3 Results for a pair of modified quantizers

In this sub-section, the proposed modified quantization method is tested in a simplified scenario. In our experiment, we generate a large number (>10,000) of input samples $x_{0[m]}$ with a uniform distribution between -1 and 1. In Case 1, $x_{0[m]}$ is quantized normally with a 7-step $q_{1[m]}$, followed by a 5-step $q_{2[m]}$.

In Case 2, $x_{0[m]}$ is quantized with the modified 7-step $\hat{q}_{1[m]}$, such that all values of $x_{0[m]}$ that fall within the tandem quantization error region are quantized in the opposite direction. The tandem quantization error regions are located at:

$$\mathbf{R}_{[m]} = \begin{cases} R_{[m],1} \in [-0.714, -0.6], & R_{[m],2} \in [-0.2, -0.143] \\ R_{[m],3} \in [0.143, 0.2], & R_{[m],4} \in [0.6, 0.714]. \end{cases}$$

In Case 3, we divide each of the tandem quantization error regions into 2 half-parts as was shown in Figure 8-9, and modified quantization is applied to the values of $x_{0[m]}$ that fall within the shaded region. The modified regions (shaded area) are located at:

$$\hat{\mathbf{R}}_{[m]} = \begin{cases} \hat{R}_{1[m]} \in [-0.714, -0.657], \ \hat{R}_{2[m]} \in [-0.172, -0.143] \\ \hat{R}_{3[m]} \in [0.143, 0.172], \ \hat{R}_{4[m]} \in [0.657, 0.714]. \end{cases}$$

The results are tabulated in Table 8-1. Here: $E[e_{1[m]}{}^2]$, $E[e_{Dm]}{}^2]$ and $E[e_{C[m]}{}^2]$ refer to the average power of the quantization errors of $q_{1[m]}(x_{0[m]})$, $q_{2[m]}(x_{0[m]})$ and the cascaded quantization $q_{2[m]}[q_{1[m]}(x_{0[m]})]$ respectively. Max inst. $[e_{1[m]}{}^2]$ and max inst. $[e_{C[m]}{}^2]$ refer to the largest instantaneous quantization error power that is incurred for the test.

	Case 1	Case 2	Case 3
$E[e_{1[m]}{}^2]$	0.0068	0.0115	0.0080
$E[e_{D[m]}{}^2]$	0.0133	0.0133	0.0133
$E[e_{C[m]}{}^2]$	0.0199	0.0133	0.0150
Max inst. $[e_{1[m]}{}^2]$	0.0204	0.0660	0.0400
Max inst. $[e_{C[m]}{}^2]$	0.0988	0.0400	0.0660

Table 8-1 Comparison of quantization error power for different cascaded quantization methods.

Figure 8-12 plots the instantaneous cascaded quantization error power for $q_{2[m]}[q_{1[m]}(x_{0[m]})]$ against the values of $x_{0[m]}$, for the 3 cases.

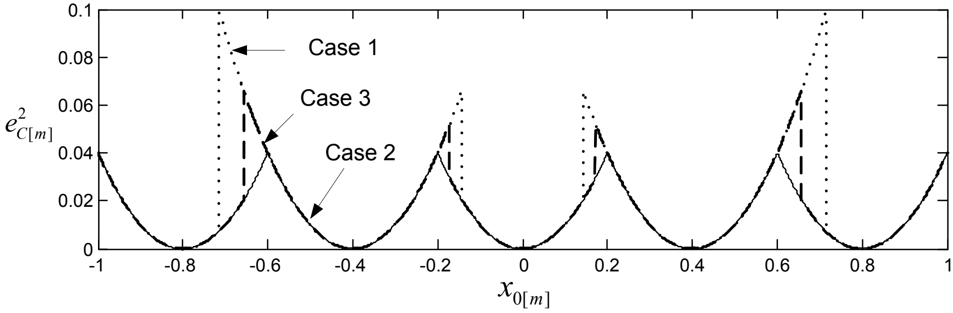


Figure 8-12 Plot of quantization error power vs. input sample value for different cascaded quantization methods.

For Case 1, tandem noise power plays a significant role; the average cascaded quantization error power, and the maximum instantaneous cascaded quantization error power of $q_{2[m]}[q_{1[m]}(x_{0[m]})]$, are large. For Case 2, tandem noise power is eliminated and the average cascaded quantization error power of $q_{2[m]}[q_{1[m]}(x_{0[m]})]$ is equal to the average cascaded quantization error power of $q_{2[m]}(x_{0[m]})$. However, $q_{1[m]}(x_{0[m]})$ is affected, as reflected by the increased average and max instantaneous quantization error power at the output of the first quantizer. Case 3 offers a compromise between the first 2 cases.

8.6.4 Possible future work using modified quantization in transcoding

We provided a theoretical groundwork for the reduction of tandem noise power, based on the assumption that the input signal $x_{0[m]}$ is uniformly distributed between -1 and 1. A practical implementation requires considerably more factors to consider, such as programming issues, compliance with the MPEG 1 Layer 2 specifications, and further tweaking and testing of the algorithm. Since our focus for Part II is on efficient transcoder implementation, we leave the implementation aspects of the reduction of tandem noise power for future work. We propose a possible basic guideline for future work below.

A possible use of the modified quantization method in audio transcoding is as follows. During the first encoding process to bit-rate B_1, the subband samples $\mathbf{x}_1$ are quantized using $\mathbf{q}_1$, which are selected based on the psychoacoustic analysis. During this quantization, the noise-to-mask ratio (NMR) generated in each subband is calculated. In subbands with low NMR, there is a potential to increase the quantization error power without significantly affecting the perceptual quality. Assuming that $\mathbf{q}_2$ is known (if not, a nearest neighbours method may be used; this is explained in Section 8.6.5), the samples $x_{0[m]}$ in these low NMR subbands that lead to large tandem noise power when cascaded with $q_{2[m]}$ are then quantized in the opposite direction with $\hat{q}_{1[m]}$. Those $x_{0[m]}$ that would lead to zero or small tandem noise power, or are in subbands where the NMR's are high, are quantized in the normal direction with $q_{1[m]}$.

Definitions of some terms used here: i) The mask-to-signal ratio is defined as the masking threshold value divided by the signal power; ii) The noise-to-mask ratio (NMR) is defined as the noise power due to the quantization error divided by the masking threshold value; iii) The noise-to-signal ratio is defined as the noise power due to the quantization error divided by the signal power.

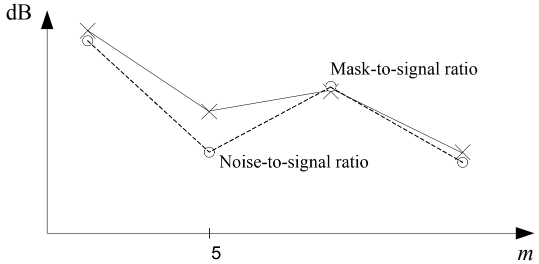


Figure 8-13 Sample illustration of noise-to-signal ratio and mask-to-signal ratio for reduction of tandem noise power.

Figure 8-13 shows a sample plot of the noise-to-signal ratio generated by the quantization of the subband signals and the mask-to-signal ratio calculated by the psychoacoustic model. Note that the NMR is equal to the noise-to-signal ratio minus the mask-to-signal ratio, if the values are given in dB. For noise-to-signal ratios that are smaller than the mask-to-signal ratios, the noise generated by the quantization is inaudible. In this example, the quantization $q_{1[5]}(x_{0[5]})$ has a noise-to-signal ratio that is well below the mask-to-signal ratio. Thus, $x_{0[5]}$

can be subjected to our modified quantizer $\hat{q}_{1[5]}$, where $\hat{q}_{1[5]}$ can be designed using the proposed method.

Though, as the result of the modified quantization method, the quantization error power in the audio compressed at bit-rate B_1 is slightly increased, it may not be perceptually significant when compared to the normal quantization method. However, when transcoded to bit-rate B_2, both the instantaneous and mean tandem noise power can be reduced.

8.6.5 Further discussions: On the assumption of a-priori knowledge of the second quantizer

On the subject of using modified quantizers during transcoding, we realize that several 'modification' options are available. The approach that was discussed involved a modification in the quantization boundaries of $q_{1[m]}$, which requires a-priori knowledge of $q_{2[m]}$.

In some applications, a-priori knowledge of $q_{2[m]}$ may be available and hence exploited. For example, in a music distribution system ([96]-[97]), music is encoded and stored at a high bit-rate on the server. Prior to distribution, the music can be transcoded to a lower bit-rate to increase transmission speed, or alternatively, watermarked ([100]) and transcoded to a lower bit-rate. Since the transcoding is also performed on the server, $q_{2[m]}$ can be made known at the time of the initial encoding.

An alternative approach when $q_{2[m]}$ is not known a-priori, might be to modify $\hat{q}_{1[m]}$ for a range of $q_{2[m]}$ that are its nearest neighbours. This is based on the reasoning that as the difference in the number of quantization steps between $q_{1[m]}$ and $q_{2[m]}$ increases, the tandem noise power tends to decrease. For example, for a 7-step $q_{1[m]}$, we might modify $\hat{q}_{1[m]}$ for a 5-step $q_{2[m]}$. Naturally, in the event that a different $q_{2[m]}$ is actually used, such as a 3-step $q_{2[m]}$, the quantization error power $e^2_{C[m]}$ is likely to be increased compared to if a non-modified $q_{1[m]}$ was used.

To illustrate this approach, consider the following example. The Case 3 $\hat{q}_{1[m]}$ (with the tandem quantization region divided into 2 half-parts) in Section 8.6.3 was designed with a 5-step $q_{2[m]}$ in mind. In Table 8-2, we use this $\hat{q}_{1[m]}$ to show the quantization error power for different conditions. We assume that $x_{0[m]}$ is uniformly distributed between -1 and 1.

When $x_{0[m]}$ is quantized once with the 7-step, 5-step and 3-step quantizers $q_{1[m]}$, the mean quantization error powers are 0.00680, 0.0133 and 0.0370 respectively. In the first and second rows of Table 8-2, the mean quantization error powers when the 7-step modified $\hat{q}_{1[m]}$ and normal $q_{1[m]}$ (respectively) were used are tabulated. In this example, it can be observed that when the 7-step modified $\hat{q}_{1[m]}$ is used, $E\left[e^2_{1[m]}\right]$ is increased by 17%, $E\left[e^2_{C[m]}\right]$ is decreased by 25% for a 5-step $q_{2[m]}$, $E\left[e^2_{C[m]}\right]$ shows no change for a 3-step $q_{2[m]}$, when compared to using the normal 7-step $q_{1[m]}$.

	$E\left[e^2_{1[m]}\right]$	$E\left[e^2_{C[m]}\right]$ for 5-step $q_{2[m]}$	$E\left[e^2_{C[m]}\right]$ for 3-step $q_{2[m]}$
Using a 7-step modified $\hat{q}_{1[m]}$	0.00797	0.0150	0.0431
Using a normal 7-step $q_{1[m]}$	0.00680	0.0199	0.0431

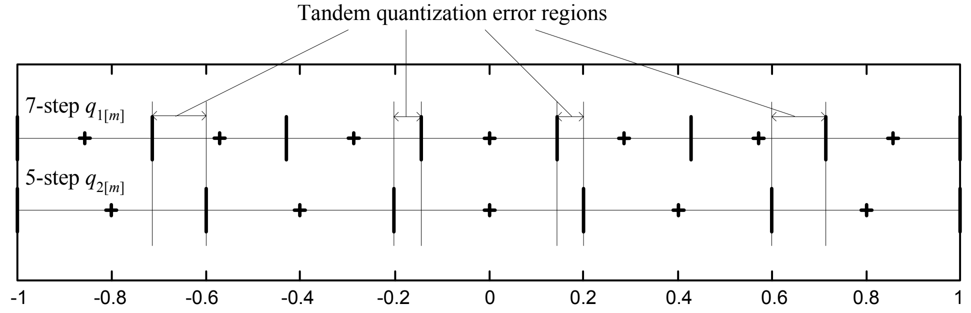
Table 8-2 Mean quantization error powers, using the modified quantizer method when $q_{2[m]}$ is not known a-priori, for the example when a 7-step $q_{1[m]}$ is used.

8.6.6 Further discussions: Modification of second quantization

Previously, we proposed a method that modifies the initial quantizer $q_{1[m]}$ to reduce the tandem noise power. In this sub-section, we consider an alternative method that modify the values (i.e. positions) of the quantization steps of the second quantizer.

Modification of the quantization steps of the second quantizer $q_{2[m]}$ presents a problem in that it requires the use of a custom decoder. To clarify, the values of the quantization steps are determined by the rescaling performed in the decoder. Since we typically do not have control over the decoder (e.g. in music distribution), the quantization steps are fixed for a certain compression method. Modifying the quantization steps of $q_{2[m]}$ would be feasible only if the transcoded bitstream is for personal use, i.e. usage of a customized bitstream decoder that is matched specifically to the transcoder. For interest, we briefly touch on the concept of a possible method.

Note that at the point of decoding the transcoded material, we typically have no knowledge of $x_{0[m]}$, $q_{0[m]}$ or $x_{1[m]}$. Thus, we require knowledge of $q_{1[m]}$, or at least some information to guide the 'modification of $q_{2[m]}$', which will therefore require additional information to be embedded into the transcoded bitstream. We use the cascaded 7-step/5step quantizer pair for illustration, and their characteristics (Figure 8-3) are included here for easy reference.



(Repeat of Figure 8-3, with unnecessary markings removed.)

Since the number of quantization steps of $q_{2[m]}$ is $N_{2[m]}$, we have $N_{2[m]}$ possible values of $x_{2[m]}$. Since $N_{2[m]} < N_{1[m]}$, each value of $x_{2[m]}$ could have originated from more than 1 value of $x_{1[m]}$. For each value of $x_{2[m]}$, we define its 'region of influence' as the range of values of $x_{0[m]}$ that eventually end up at the value of $x_{2[m]}$ after cascaded quantization with $q_{1[m]}$ and $q_{2[m]}$.

In this example, the regions of influence are:

$x_{2[m]}$	Region of influence
-0.8	[-1. -0.7143]
-0.4	[-0.7143, -0.1429]
0.0	[-0.1429, 0.1429]
0.4	[0.1429, 0.7143]
0.8	[0.7143, 1]

Table 8-3 Regions of influence for a 7-step/5-step cascaded pair.

A straightforward method to determine the quantization steps of the modified $\hat{q}_{2[m]}$ would then be to place them at the centre of each region of influence. In this example, they would be given by $\hat{x}_{2[m]} \in \{-0.857, -0.4286, 0, 0.4286, 0.857\}$. More advanced methods might take into account the distribution of $x_{1[m]}$ (which would then require some more information to be embedded into the transcoded bitstream).

8.7 Quantizer selection method

A 'quantizer selection' method that does not modify either $q_{1[m]}$ or $q_{2[m]}$ is proposed next. A-priori knowledge of $q_{2[m]}$ is also not required. Consider the transcoding from an MPEG 1 Layer 2 audio encoded at bit-rate B_1 to an MPEG 1 Layer 2 audio encoded at bit-rate B_2. Without the availability of the PCM source, $x_{0[m]}$ is not known. Furthermore, in this context, $q_{1[m]}$ is known since the information on $q_{1[m]}$ is included in the bitstream. In this case, contrary to the modified quantizer method, $x_{1[m]}$ is already fixed and we do not have the option to modify the initial encoding process.

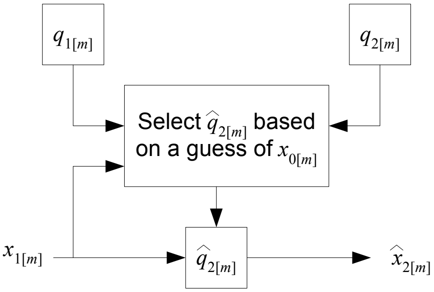


Figure 8-14 Block diagram for quantizer selection method.

The block diagram for the proposed method is shown in Figure 8-14. Assume that the transcoding is performed as per the normal practice, e.g. cascaded decoder-encoder. However, instead of quantizing $x_{1[m]}$ with the $q_{2[m]}$ that was chosen by the psychoacoustic model (assuming the chosen $q_{2[m]}$ has fewer quantization steps than $q_{1[m]}$), another quantizer represented by $\hat{q}_{2[m]}$ is used.

The decision which $\hat{q}_{2[m]}$ to use is based on a guess of likely values of $x_{0[m]}$, given the input $x_{1[m]}$, and the $q_{1[m]}$ that was used to quantize $x_{0[m]}$ to $x_{1[m]}$. If these likely values of $x_{0[m]}$ give rise to significant tandem noise power when $q_{2[m]}$ is used, then we attempt to use a $\hat{q}_{2[m]}$, where $\hat{q}_{2[m]} \neq q_{2[m]}$, so that $\hat{x}_{2[m]}$ has a lower tandem noise power than $x_{2[m]} = q_{2[m]}(x_{1[m]})$.

In the simple method that follows (Section 8.7.2), we allow only 2 possible options for $\hat{q}_{2[m]} \in \{q_{1[m]}, q_{2[m]}\}$. Thus, when the likely values of $x_{0[m]}$ give rise to significant tandem noise power when $q_{2[m]}$ is used, we use $\hat{q}_{2[m]} = q_{1[m]}$. The output $\hat{x}_{2[m]}$ is then equal to $x_{1[m]}$, i.e. no change to $x_{1[m]}$. The output $\hat{x}_{2[m]}$ therefore takes on 1 of 2 possible values, $\hat{x}_{2[m]} \in \{x_{1[m]}, x_{2[m]}\}$. The overall tandem noise power is reduced when compared to normal transcoding, because values of $x_{2[m]}$ that are estimated to have large tandem noise power, are replaced with $x_{1[m]}$ which have zero tandem noise power. The drawback is a larger final bit-rate, when compared to normal transcoding.

8.7.1 Selection of quantizer

In this section, we consider the issues involved in the selection of $\hat{q}_{2[m]}$. The number of quantization steps of $\hat{q}_{2[m]}$ is $\hat{N}_{2[m]}$, and $N_{2[m]} \le \hat{N}_{2[m]} \le N_{1[m]}$, where $N_{1[m]}$ and $N_{2[m]}$ are the number of quantization steps of $q_{1[m]}$ and $q_{2[m]}$ (chosen by the psychoacoustic model) respectively. Values of $\hat{N}_{2[m]}$ outside of this range are meaningless; if $\hat{N}_{2[m]} > N_{1[m]}$ then the bit-rate would be increased, if $\hat{N}_{2[m]} < N_{2[m]}$ then the resultant error power would likely be greater than allowed (by the psychoacoustic model).

A number of consecutive samples $x_{1[m]}$ which are within the same q-frame were previously quantized (during initial encoding) by the same $q_{1[m]}$. For MPEG 1 Layer 2, the number of consecutive samples $x_{1[m]}$ in a q-frame is 12. For each value of $x_{1[m]}$, we make the assumption that its true value, i.e. $x_{0[m]}$, has an equal probability of being any value within the range defined by the 2 quantization boundaries of $q_{1[m]}$ which quantizes to $x_{1[m]}$.

When $q_{1[m]}$ is cascaded with $q_{2[m]}$, $x_{2[m]} = q_{2[m]}(x_{1[m]})$. Clearly:

$$E\left[e_{C[m]}^2\right] = \int_{\Phi_{1[m]}^L}^{\Phi_{1[m]}^U} \left(x_{0[m]} - x_{2[m]}\right)^2 p(x_{0[m]})\, dx_{0[m]}, \tag{8.24}$$

where $\Phi_{1[m]}^L$ and $\Phi_{1[m]}^U$ are the values of the quantization boundaries of $q_{1[m]}$ which quantize to $x_{1[m]}$, $\Phi_{1[m]}^U$ being the larger of the two; and $p(x_{0[m]})$ is the probability of $x_{0[m]}$.

If $x_{0[m]}$ is uniformly distributed between $\Phi_{1[m]}^L$ and $\Phi_{1[m]}^U$:

$$E\left[e_{C[m]}^2\right] = \frac{1}{\Phi_{1[m]}^U - \Phi_{1[m]}^L} \int_{\Phi_{1[m]}^L}^{\Phi_{1[m]}^U} \left(x_{0[m]} - x_{2[m]}\right)^2 dx_{0[m]} . \tag{8.25}$$

8.7.2 *Simple method*

In the simple method that we propose here, we use a $\hat{q}_{2[m]}$ such that:

$$\hat{q}_{2[m]} = q_{2[m]} \text{, if } \frac{\overline{E\left[P_{T[m]}\right]}}{\overline{E\left[e_{D[m]}^2\right]}} \le \sigma , \tag{8.26}$$

or:

$$\hat{q}_{2[m]} = q_{1[m]} \text{, if } \frac{\overline{E\left[P_{T[m]}\right]}}{\overline{E\left[e_{D[m]}^2\right]}} > \sigma , \tag{8.27}$$

where σ is a tolerance value that is set by the user.

Assume that each q-frame has 12 consecutive samples $x_{1[m]}$. For each q-frame, $\overline{E\left[P_{T[m]}\right]}$ is calculated as the average of the 12 values of $E\left[P_{T[m]}\right]$, where $E\left[P_{T[m]}\right]$ is the likely tandem noise power for 1 sample of $x_{1[m]}$. Similarly, $\overline{E\left[e_{D[m]}^2\right]}$ is calculated as the average of the 12 values of $E\left[e_{D[m]}^2\right]$, where $E\left[e_{D[m]}^2\right]$ is the likely quantization error power $\left[q_{2[m]}(x_{0[m]}) - x_{0[m]}\right]^2$ for 1 sample of $x_{1[m]}$.

The value of $E\left[e_{D[m]}^2\right] = E\left[\left(q_{2[m]}(x_{0[m]}) - x_{0[m]}\right)^2\right]$ can be calculated for each $x_{1[m]}$, where $x_{0[m]}$ is assumed to be uniformly distributed between the 2 quantization boundaries of $q_{1[m]}$ that are nearest to $x_{1[m]}$:

$$\begin{aligned} E\left[e_{D[m]}^2\right] &= \int_{\Phi_{1[m]}^L}^{\Phi_{1[m]}^U} \left(x_{0[m]} - x_{D[m]}\right)^2 p(x_{0[m]})\, dx_{0[m]} \\ &= \frac{1}{\Phi_{1[m]}^U - \Phi_{1[m]}^L} \int_{\Phi_{1[m]}^L}^{\Phi_{1[m]}^U} \left(x_{0[m]} - x_{D[m]}\right)^2 dx_{0[m]} , \end{aligned} \tag{8.28}$$

where $x_{D[m]} = q_{2[m]}(x_{0[m]})$.

The likely tandem noise power can be calculated from:

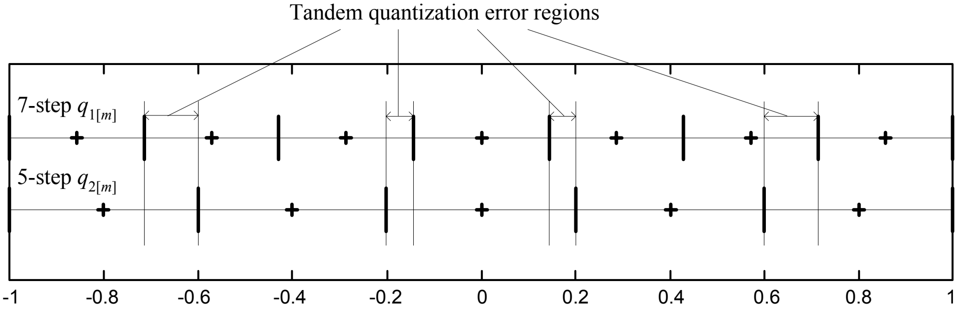
$$E\left[P_{T[m]}\right] = E\left[e_{C[m]}^2\right] - E\left[e_{D[m]}^2\right] . \tag{8.29}$$

8.7.3 Example and results for a 7-step initial quantization

In this sub-section, we illustrate the use of the quantizer selection method, by using a 7-step $q_{1[m]}$ as an example. Consider a uniformly distributed sample $x_{0[m]}$ which is quantized to $x_{1[m]}$ using a 7-step $q_{1[m]}$. The possible values that $x_{1[m]}$ can take are the quantization steps of $q_{1[m]}$, which are:

$$\Theta_{1[m]} = \{\text{-0.8571} \quad \text{-0.5714} \quad \text{-0.2857} \quad 0 \quad 0.2857 \quad 0.5714 \quad 0.8571\}.$$

Assume that a 5-step $q_{2[m]}$ was chosen by the transcoder. We repeat the plot of the tandem quantization error regions for a 7-step/5-step pair (Figure 8-3) here for easy reference.



(Repeat of Figure 8-3, with unnecessary markings removed.)

$\underline{x_{1[m]} = 0}$: There are no tandem quantization error regions within the 2 nearest quantization boundaries. Therefore, $E\left[P_{T[m]}\right] = 0$ and $E\left[e_{D[m]}^2\right] = 0.00681$, where:

$$E\left[e_{D[m]}^2\right] = \frac{1}{0.2857}\int_{-0.1429}^{0.1429}\left(x_{0[m]} - 0\right)^2 dx_{0[m]}.$$

$\underline{x_{1[m]} = 0.2857}$: We calculated $E\left[P_{T[m]}\right] = 0.0046$ and $E\left[e_{D[m]}^2\right] = 0.0153$, where:

$$E\left[e_{C[m]}^2\right] = \frac{1}{0.2857}\int_{0.1429}^{0.4286}\left(x_{0[m]} - 0.4\right)^2 dx_{0[m]},$$

$$E\left[e_{D[m]}^2\right] = \frac{1}{0.2857}\int_{0.1429}^{0.2}\left(x_{0[m]} - 0\right)^2 dx_{0[m]} + \frac{1}{0.2857}\int_{0.2}^{0.4286}\left(x_{0[m]} - 0.4\right)^2 dx_{0[m]}.$$

$\underline{x_{1[m]} = 0.5714}$: We calculated $E\left[P_{T[m]}\right] = 0.0183$ and $E\left[e_{D[m]}^2\right] = 0.0179$, where:

$$E\left[e_{C[m]}^2\right] = \frac{1}{0.2857}\int_{0.4286}^{0.7143}\left(x_{0[m]} - 0.4\right)^2 dx_{0[m]},$$

$$E\left[e_{D[m]}^2\right] = \frac{1}{0.2857}\int_{0.4286}^{0.6}\left(x_{0[m]} - 0.4\right)^2 dx_{0[m]} + \frac{1}{0.2857}\int_{0.6}^{0.7143}\left(x_{0[m]} - 0.8\right)^2 dx_{0[m]}.$$

$x_{1[m]} = 0.8571$: There are no tandem quantization error regions within the 2 nearest quantization boundaries. Therefore, $E\left[P_{T[m]}\right] = 0$ and $E\left[e_{D[m]}^2\right] = 0.0101$, where:

$$E\left[e_{D[m]}^2\right] = \frac{1}{0.2857}\int_{0.7143}^{1}\left(x_{0[m]} - 0.8\right)^2 dx_{0[m]}.$$

Thus, for a 7-step/5-step cascaded pair, we tabulate the values of $E\left[P_{T[m]}\right]$ and $E\left[e_{D[m]}^2\right]$ in Table 8-4.

	$x_{1[m]} = 0$	$x_{1[m]} = 0.2857$, or -0.2857	$x_{1[m]} = 0.5714$, or -0.5714	$x_{1[m]} = 0.8571$, or -0.8571
$E\left[P_{T[m]}\right]$	0.0000	0.0046	0.0183	0.0000
$E\left[e_{D[m]}^2\right]$	0.0068	0.0153	0.0179	0.0101

Table 8-4 Reference table for a 7-step/5-step cascaded pair.

We can see from Table 8-4 that for values of $x_{1[m]} \in \{0, \pm 0.8571\}$, no tandem noise power is incurred when a 7-step $q_{1[m]}$ is cascaded with a 5-step $q_{2[m]}$. For values of $x_{1[m]} \in \{\pm 0.2857\}$, about 23% of $E\left[e_{C[m]}^2\right]$ is due to tandem noise power, and for values of $x_{1[m]} \in \{\pm 0.5714\}$, about 51% of $E\left[e_{C[m]}^2\right]$ is due to tandem noise power.

For a q-frame with 12 samples $x_{1[m]}$ which were quantized using a 7-step $q_{1[m]}$, the ratio $\overline{E\left[P_{T[m]}\right]}/\overline{E\left[e_{D[m]}^2\right]}$ ranges from a minimum value of 0 (for the case when $x_{1[m]} \in \{0, \pm 0.8571\}$) to a maximum value of 1.02 (for the case when $x_{1[m]} \in \{\pm 0.5714\}$). Therefore, we can set a tolerance value $0 \le \sigma \le 1.02$, such that for $\overline{E\left[P_{T[m]}\right]}/\overline{E\left[e_{D[m]}^2\right]}$ that are larger than σ, we choose $\hat{q}_{2[m]} = q_{1[m]}$. For $\overline{E\left[P_{T[m]}\right]}/\overline{E\left[e_{D[m]}^2\right]}$ that are smaller than or equal to σ, we choose $\hat{q}_{2[m]} = q_{2[m]}$.

In a simulation to test our method for different values of σ, we used a sample size of 100,000 q-frames. In each q-frame, a set of 12 values of uniformly distributed $x_{1[m]}$ was generated.

Figure 8-15 shows a plot of the ratio of the number of q-frames that are unchanged (i.e. $\hat{q}_{2[m]} = q_{1[m]}$) to the total number of q-frames, against the tolerance σ. A larger ratio indicates a smaller reduction in the overall bit-rate after transcoding (i.e. having a poor bit-rate efficiency), and vice-versa.

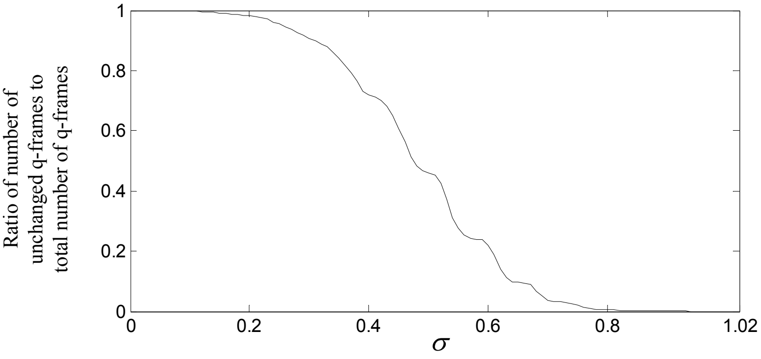


Figure 8-15 Plot of the ratio of the number of unchanged q-frames to the total number of q-frames, against the tolerance σ.

Figure 8-16 shows a plot of the mean quantization error power $E\left[e_{C[m]}^2\right]$ against the tolerance σ. It can be clearly seen from the results that a smaller tolerance value of σ results in a smaller mean quantization error power $E\left[e_{C[m]}^2\right]$, at the cost of a larger bit-rate. Conversely, a larger tolerance value of σ results in a larger mean quantization error power $E\left[e_{C[m]}^2\right]$, and a smaller bit-rate.

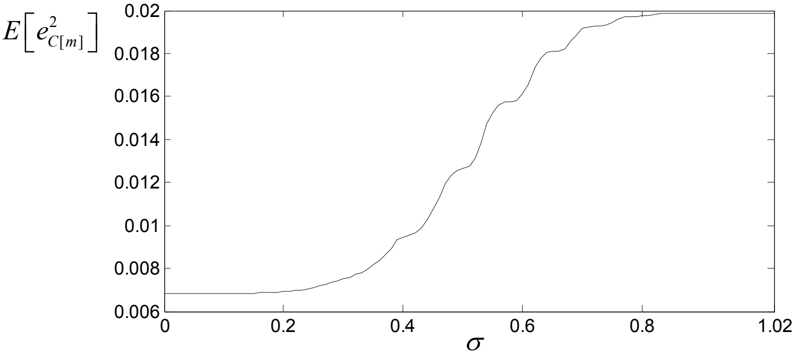


Figure 8-16 Plot of the mean quantization error power $E\left[e_{C[m]}^2\right]$ against the tolerance σ.

8.7.4 *Remarks on quantizer selection method*

The drawback of the quantizer selection method is that the final bit-rate using the quantizer selection method will be larger than bit-rate B_2, assuming that bit-rate B_2 was initially fed as a requirement into the transcoder. This is because some of the quantizers that are selected to be the finer $q_{1[m]}$ would, under normal transcoding, have been selected to be the coarser $q_{2[m]}$. If bit-rate B_2 is required, we can define a bit-rate B_3 (perhaps by using an iterative guess) that is smaller than B_2, and the target bit-rate of B_3 is then fed as a requirement into the transcoder.

The simple method that was proposed considers only a quantizer $\hat{q}_{2[m]}$ that is either equal to $q_{1[m]}$ or $q_{2[m]}$, and can be further improved by allowing $\hat{q}_{2[m]}$ to be a quantizer with a number of quantization steps between the range of $N_{1[m]}$ and $N_{2[m]}$. Similarly, as for the modified quantizer method, we provided some groundwork for the quantizer selection method and based our results on the statistical assumption that the input signal $x_{0[m]}$ is uniformly distributed between -1 and 1. The implementation aspects are left for future work. The simple method proposed in this section assumes that $\hat{q}_{2[m]}$ takes on one of two possible options, $\hat{q}_{2[m]} = q_{1[m]}$ or $\hat{q}_{2[m]} = q_{2[m]}$. The investigation of using a $\hat{q}_{2[m]}$ that takes on one of a greater selection of possible options can also be room for future work.

8.8 Comments for other distributions

We illustrated the effect of tandem quantization error by using input signals that are uniformly distributed. In actual applications, this may not be the case. It was noted that for MPEG 1 Layer 2, the distribution of the subband samples (prior to quantization) is typically more concentrated towards the zero value and less concentrated towards the +1 and -1 values. Let us consider the implications of this distribution on tandem quantization error. There are 2 cases of interest here.

Case 1: $\left|\Phi_{1[m]}\right| < \left|\Phi_{2[m]}\right|$

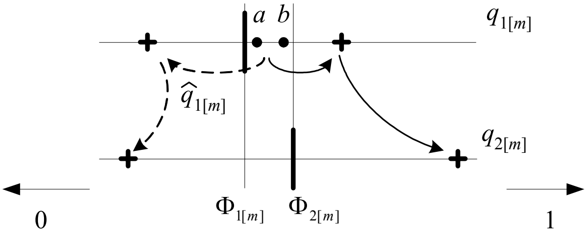


Figure 8-17 Illustration of tandem quantization error for an input signal with non-uniform distribution - Case 1.

In the first case, the value of $\Phi_{1[m]}$ is nearer to zero, than the value of $\Phi_{2[m]}$ is. This is illustrated in Figure 8-17, where the zero point on the quantizer axis lies towards the left, and the +1 point on the quantizer axis lies towards the right. Samples of $x_{0[m]}$ falling between $\Phi_{1[m]}$ and $\Phi_{2[m]}$, such as at points '*a*' and '*b*', follow the quantization path as indicated by the solid arrows. Due to the distribution of $x_{0[m]}$, there is a higher probability of $x_{0[m]}$ at '*a*' then at '*b*'. It can clearly be seen that the mean tandem noise power $E\left[P_{T[m]}\right]$ would be larger in this case, than for the case of a uniformly distributed $x_{0[m]}$.

The dashed arrows show the quantization path when the modified quantizer $\hat{q}_{1[m]}$ is used. Since there is a higher probability of $x_{0[m]}$ at '*a*' then at '*b*', the mean quantization error power

$E\left[\hat{e}^2_{1[m]}\right]$ due to using $\hat{q}_{1[m]}$ is smaller in this case, than for the case of a uniformly distributed $x_{0[m]}$.

Case 2: $\left|\Phi_{1[m]}\right| > \left|\Phi_{2[m]}\right|$

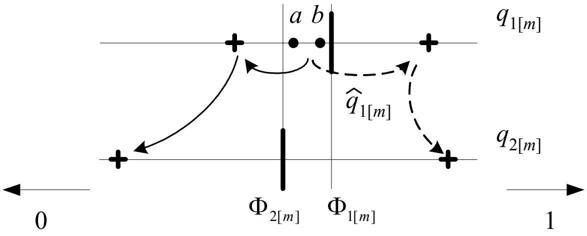


Figure 8-18 Illustration of tandem quantization error for an input signal with non-uniform distribution - Case 2.

In the second case, the value of $\Phi_{2[m]}$ is nearer to zero, than the value of $\Phi_{1[m]}$ is. This is illustrated in Figure 8-18. Since there is a higher probability of $x_{0[m]}$ at '*a*' then at '*b*', $E\left[P_{T[m]}\right]$ is smaller (for the solid arrow path) and $E\left[\hat{e}^2_{1[m]}\right]$ is larger (for the dashed arrow path) in this case, than for the case of a uniformly distributed $x_{0[m]}$.

8.9 Summary

During sample-synchronized transcoding, which we represented using a cascaded quantization model, we isolated an error component called tandem quantization error. We investigated the nature of tandem quantization error, with reference to the quantizers used in the MPEG 1 Layer 2 compression method. Tandem quantization error is a characteristic that depends on the nature of the cascaded quantizer pair, $q_{1[m]}$ and $q_{2[m]}$.

Several methods for reducing the total quantization error power of the signal at the output of the transcoder were proposed. In the modified quantizer method, characteristics of either $q_{1[m]}$ or $q_{2[m]}$ are modified. One proposed method was to modify the quantization boundaries of $q_{1[m]}$, making use of a-priori knowledge of $q_{2[m]}$ (or in the event that $q_{2[m]}$ is not known, a nearest-neighbours approach is used). Another method, in which the values of the quantization steps of $q_{2[m]}$ are modified, was also highlighted briefly. This method requires the use of a custom decoder that is matched to the transcoder, thus necessitating the embedding of additional information in the transcoded bitstream (such as information on the $q_{1[m]}$ that was used).

The quantizer selection method takes an approach that does not modify the quantizers. It uses the values of $x_{1[m]}$ in each q-frame, and calculates the likely tandem noise power when $q_{1[m]}$ is cascaded with the $q_{2[m]}$ chosen by the psychoacoustic model in the transcoder. If the likely tandem noise power is beyond a specified tolerance, then another quantizer is selected in place of $q_{2[m]}$.

Chapter 9

Analysis of the Impact of Sample-synchronized and Non-sample-synchronized Audio Transcoding for MPEG 1

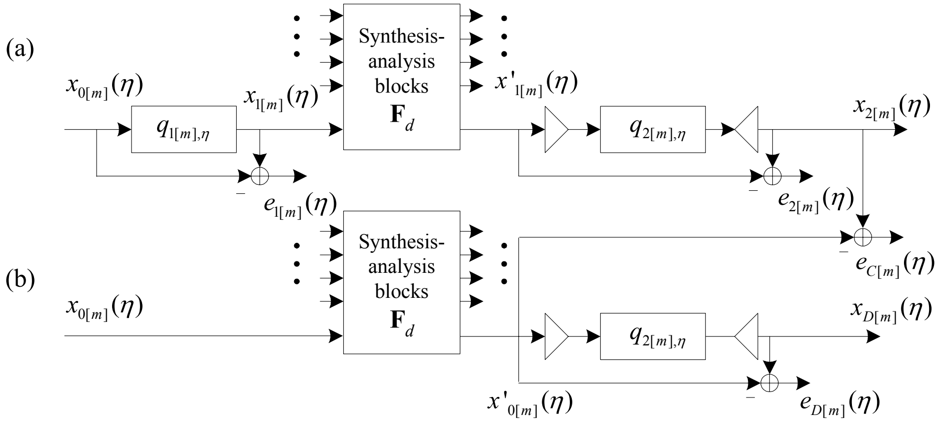
It is known that sample-synchronization (defined in Section 7.4.3) can have different effects on transcoded audio material for different compression methods [83]. For example, sample-synchronized transcoding is preferred for the MPEG 1 Layer 2 method, in terms of output audio quality. On the contrary, non-sample-synchronized transcoding is preferred for the MP3 method. Sample-synchronization can be trivially realized by the insertion of delay elements in the transcoder, such that its input-to-output delay is a multiple of the decimation factor (equal to the number of subbands for critically-decimated subband samples). This was explained in Section 7.4.3 and further details can be found in Appendix D.

Note that the additional delay required is known, if the entire transcoding process is performed at once (i.e. decoding is followed by encoding immediately). The original delay of the transcoder can then be easily established; for example, by first encoding from and then decoding to a PCM signal, the transcoder delay is then measured as the delay of the PCM output relative to the PCM input. The required delay is computed such that the sum of the required delay and the original delay of the transcoder is a multiple of the decimation factor. In this thesis, we only consider transcoding processes where decoding and encoding are performed at once.

Consider as a counter-example to the above, a compressed MP3 bitstream which is decoded to an intermediate PCM signal, and is then stored to an audio CD. The CD is distributed, and the end-user decides to encode the PCM signal back to MP3. In this case, the delay of the decoding stage is not known at the encoding stage. To address this problem, methods to estimate the delay of the decoding stage based on the intermediate PCM signal were proposed in [84]-[85].

In this chapter, we investigate the impact of sample-synchronization and non-sample-synchronization on audio transcoding. The MPEG 1 Layer 2 compression method is first discussed, followed by MP3.

9.1 Overview



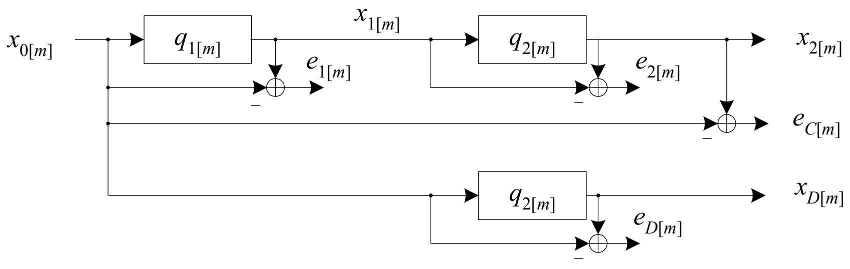
(Repeat of Figure 7-5, with additional blocks showing normalization scaling.)

In this chapter, we hide the sample-time variable η for the sake of convenience. The cascaded quantization model (Figure 7-5), which was used to model audio transcoding, is repeated above. A detailed explanation of the figure can be found in Section 7.3. Note that we added normalization scaling blocks in this figure, where the right-pointing triangle represents normalization scaling and the left-pointing triangle represents inverse normalization scaling, which is applicable to the case of MPEG 1 Layer 2.

9.2 Analysis for MPEG 1 Layer 2

9.2.1 Overview

An explanation of the MPEG 1 Layer 2 method was provided in Chapter 7 (more information is available in [71]). It is known ([83]-[85]) that for this method, sample-synchronized transcoding is preferred over non-sample-synchronized transcoding, in relation to the audio quality of the transcoded material.



(Repeat of Figure 8-1.)

In Chapter 8, we provided a detailed analysis of sample-synchronized transcoding for MPEG 1 Layer 2. Figure 8-1, which shows the reduced block diagram of the cascaded quantization model for sample-synchronized transcoding, is repeated above. We defined the tandem noise power of the transcoded signal $x_{2[m]}$ as $P_{T[m]}$, where:

$$P_{T[m]} = e^2_{C[m]} - e^2_{D[m]}. \tag{9.1}$$

The characteristics of the tandem noise power were shown to be a function of the input signal $x_{0[m]}$ and the quantizers represented by $q_{1[m]}$ and $q_{2[m]}$.

9.2.2 Non-sample-synchronized transcoding

When the transcoding is non-sample-synchronized, the samples $x_{1[m]}$ (for $0 \le m \le M-1$) are transformed into $x'_{1[m]}$ by the set of transform functions $f_{[m',m]}$, as a result of the filter bank block processing. The values of $x'_{1[m]}$ vary for different values of delay d (measured from the input to the output of the filter bank block). Details on $f_{[m',m]}$, the effect of different values of d on $x'_{1[m]}$ and the transform operation from $x_{1[m]}$ to $x'_{1[m]}$ are found in Appendix D.

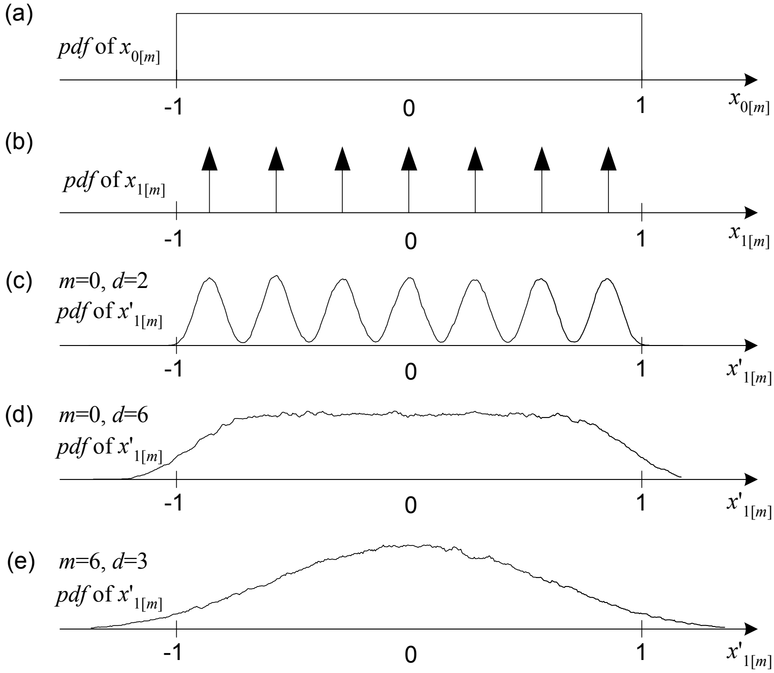


Figure 9-1 Effect of non-sample-synchronization on the distribution of $x'_{1[m]}$, for a uniform distribution of $x_{0[m]}$.

Assuming that $x_{0[m]}$ is uniformly distributed between -1 and 1 (Figure 9-1(a)), the quantization of $x_{0[m]}$ with a 7-step $q_{1[m]}$ gives $x_{1[m]}$ with the distribution shown in Figure 9-1(b). Figure 9-1(c)-(e) shows experimentally-obtained distributions of $x'_{1[m]}$ for different values of m and d, where $x'_{1[m]}$ can be found from $x_{1[m]}$ by the transformation using $f_{[m',m]}$ (refer to Appendix D, Section D.4). A simple way to view the effect of the transform operation on $x_{1[m]}$ is to visualize the values $x'_{1[m]}$ as being 'dispersed' with respect to the discretely distributed values $x_{1[m]}$.

For a non-uniform distribution of $x_{0[m]}$, the same outcome is observed, i.e. dispersal of the discretely distributed values $x_{1[m]}$ for the case of non-sample-synchronization. A typical distribution of $x_{0[m]}$ for MPEG 1 Layer 2 has a larger concentration of values towards zero, and a smaller concentration of values towards +1 and -1. In this case, it is intuitive that the corresponding distribution curves would also show a larger concentration towards zero, and a smaller concentration towards +1 and -1, such as shown in Figure 9-2.

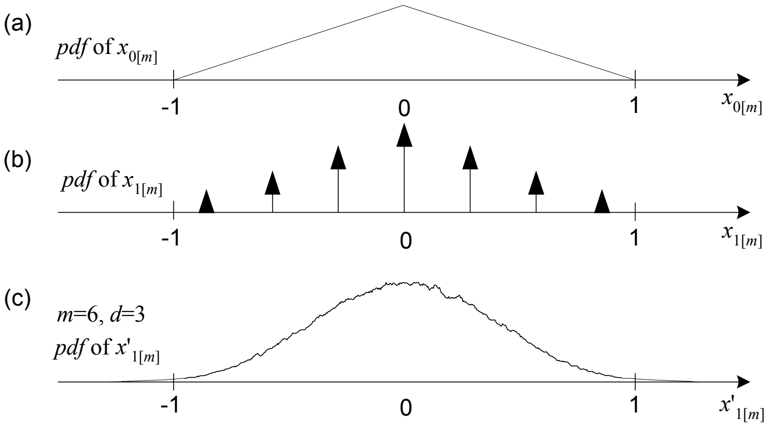
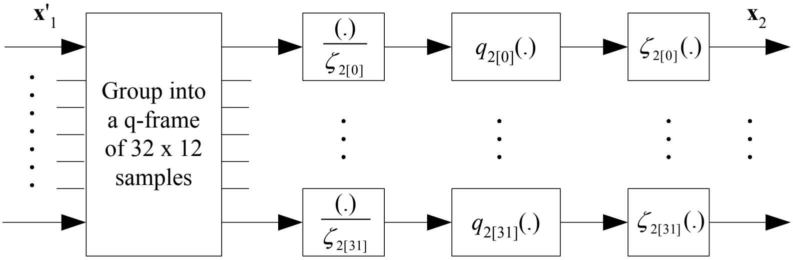


Figure 9-2 Effect of non-sample-synchronization on the distribution of $x'_{1[m]}$, for a non-uniform distribution of $x_{0[m]}$.

It is known that for MPEG 1 Layer 2, sample-synchronization is preferred for transcoding, in terms of audio quality. To explain this, we introduce a more advanced cascaded quantization model for MPEG 1 Layer 2, that takes into account the scaling of $x'_{1[m]}$ to the normalized range of -1 and +1.

9.2.3 Effect of normalization scaling on cascaded quantization


$\mathbf{x}'_1$
$\mathbf{x}_2$
Group into a q-frame of 32 x 12 samples
$\frac{(.)}{\zeta_{2[0]}}$
$q_{2[0]}(.)$
$\zeta_{2[0]}(.)$
$\frac{(.)}{\zeta_{2[31]}}$
$q_{2[31]}(.)$
$\zeta_{2[31]}(.)$

(Repeat of Figure 7-8.)

For MPEG 1 Layer 2 (refer to Section 7.5.1), the subband samples $x'_{1[m]}$ are grouped into a q-frame, and the values of $x'_{1[m]}$ are normalized by a scaling factor $\zeta_{2[m]}$ to the range of -1 and +1 prior to quantization by $q_{2[m]}$ (see Figure 7-8 and Figure 7-9 for an illustration of normalization scaling). This is necessary so that large values of $x'_{1[m]}$ do not saturate the quantizer, and small values of $x'_{1[m]}$ acquire sufficient dynamic resolution when quantized. The value of $\zeta_{2[m]}$ for each q-frame is selected to be the smallest value from a pre-defined discrete set (see [71]); such that the largest absolute value of $x'_{1[m]}$ in the same q-frame, when divided by $\zeta_{2[m]}$, has a magnitude less than unity.

From Figure 9-1 and Figure 9-2, we notice that the distribution of $x'_{1[m]}$ is 'dispersed' into a range of values which may exceed the range of -1 and +1, depending on the values of m and d. In this case, $\zeta_{2[m]}$ takes on a value which is larger than unity. Here, we show that the normalization scaling is an important consideration when estimating the mean quantization error power $E\left[e_{C[m]}^2\right]$ during non-sample-synchronized transcoding.

Let us define the effective scaling factor $\zeta_{E[m]}$ as the root-mean-squared value of $\zeta_{2[m]}$:

$$\zeta_{E[m]} = \sqrt{E\left[\zeta_{2[m]}^2\right]}. \tag{9.2}$$

Taking into account the normalization scaling, the signal that is fed as input to $q_{2[m]}$ becomes $x'_{1[m]}/\zeta_{2[m]}$. Therefore, $x_{2[m]}$ can be expressed as:

$$x_{2[m]} = \zeta_{2[m]} q_{2[m]}\left(x'_{1[m]}/\zeta_{2[m]}\right), \tag{9.3}$$

and the error $e_{2[m]} = x_{2[m]} - x'_{1[m]}$ can be expressed as:

$$e_{2[m]} = \zeta_{2[m]} e_{q2[m]}, \tag{9.4}$$

where $e_{q2[m]}$ is the quantization error that is locally visible to $q_{2[m]}$:

$$e_{q2[m]} = q_{2[m]}\left(x'_{1[m]}/\zeta_{2[m]}\right) - x'_{1[m]}/\zeta_{2[m]}. \tag{9.5}$$

Therefore:

$$E\left[e_{C[m]}^2\right] \approx E\left[e_{1[m]}^2\right] + \zeta_{E[m]}^2 E\left[e_{q2[m]}^2\right]. \tag{9.6}$$

The effective scaling factor $\zeta_{E[m]}$ is typically larger than unity when transcoding non-sample-synchronously (due to the 'dispersal' effect). Therefore, $E\left[e_{C[m]}^2\right]$ is typically larger when transcoding non-sample-synchronously than sample-synchronously. (Note that $E\left[e_{C[m]}^2\right]$ in the sample-synchronized case can be approximated by setting $\zeta_{E[m]}$ to unity).

To verify the validity of our assumptions and results, we performed several experiments and show an example where $N_{1[m]}$=127, $N_{2[m]}$=63 and $x_{0[m]}$ (for $0 \le m \le M-1$) is uniformly distributed between -1 and 1. Assuming that the mean quantization error power due to the quantization of a uniformly distributed sample by a uniform quantizer with step-size $\Delta_{[m]}$ is $E\left[e_{[m]}^2\right] \approx \Delta_{[m]}^2/12$ (a well-known result), we estimate $E\left[e_{1[m]}^2\right] \approx 0.0000207$ and $E\left[e_{2[m]}^2\right] \approx 0.0000840$. The solid lines in Figure 9-3 show the measured values of $E\left[e_{C[m]}^2\right]$ (for $0 \le m \le M-1$, and for d=1 and d=6) when we used generated values of $x_{0[m]}$. The dotted lines show the predicted values of $E\left[e_{C[m]}^2\right]$ using (9.6). These values correspond closely to the measured values of $E\left[e_{C[m]}^2\right]$.

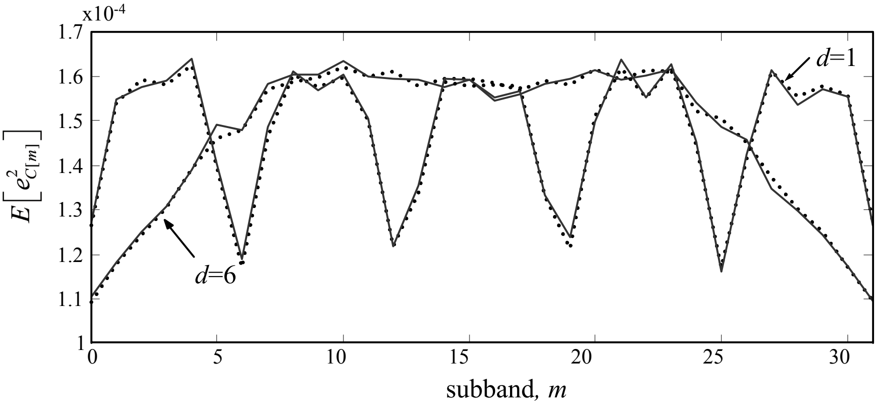


Figure 9-3 Comparison of predicted and measured values of $E\left[e_{C[m]}^2\right]$ for different m, and for d=1 and d=6.

We further averaged the values of $\zeta_{E[m]}$ across all m, for each value of d ($0 \le d \le 31$), where:

$$\overline{\zeta_{E[m]}} = \frac{1}{32}\sum_{m=0}^{31}\zeta_{E[m]} . \tag{9.7}$$

It was found that $\overline{\zeta_{E[m]}} \approx 0.99$ when d=0. This result agrees with our prior assumption that $\overline{\zeta_{E[m]}}$ is approximately unity when the transcoding is sample-synchronized. For values of d between 1 and 31, we found that $\overline{\zeta_{E[m]}}$ was approximately 1.23 for each value of d (also see note below). Therefore, we expect $\overline{E\left[e_{C[m]}^2\right]}$ to exhibit a 48% $\left((1.23-1)^2\right)$ increase when the transcoding is non-sample-synchronous (as compared to sample-synchronous), where:

$$\overline{E\left[e_{C[m]}^2\right]} = \frac{1}{32}\sum_{m=0}^{31}E\left[e_{C[m]}^2\right] . \tag{9.8}$$

This also corresponds with our measured values which were:

Sample-synchronized transcoding : $\overline{E\left[e_{C[m]}^{2}\right]} \approx 0.000105$

Non-sample-synchronized transcoding : $\overline{E\left[e_{C[m]}^{2}\right]} \approx 0.000148$

Note: Although it is known that $\zeta_{E[m]}$ varies with different values of *m* and *d*, we found that $\overline{\zeta_{E[m]}}$ (which is averaged across all *m*) does not vary significantly with *d*. For a fixed *d*, say *d*=1, some values of $\zeta_{E[m]}$ are larger than 1.23 (e.g. for *m*=30), and some are smaller than 1.23 (e.g. for *m*=0). The mean value $\overline{\zeta_{E[m]}}$, is approximately 1.23. For a different *d*, say *d*=5, although a different set of larger and smaller values (relative to 1.23) of $\zeta_{E[m]}$ exists, it turns out that the mean value $\overline{\zeta_{E[m]}}$ remains approximately 1.23.

9.2.4 Remarks

The scaling factor $\zeta_{E[m]}$ can be used to provide an indication of the increase in $E\left[e_{C[m]}^{2}\right]$, for different *m* and *d*. Since $E\left[e_{C[m]}^{2}\right]$ is increased (as a result of the 'dispersion' of $x_{1[m]}$) when the transcoding is non-sample-synchronous (as compared to sample-synchronous), we can draw the conclusion that sample-synchronized transcoded material generally has a better audio quality than non-sample-synchronized transcoded material, for MPEG 1 Layer 2 audio.

We realize that it is possible to calculate $\zeta_{E[m]}$ for a variety of $x_{0[m]}$ distributions, *d*, *m*, and $q_{1[m]}$; and then use these values of $\zeta_{E[m]}$ to estimate $E\left[e_{C[m]}^{2}\right]$. However, such a method is perhaps too painstaking. Instead, it might be more reasonable to make a few simplified conclusions for MPEG 1 Layer 2 audio:

a. Sample-synchronized transcoding ($\overline{\zeta_{E[m]}} \approx 1$) is better than non-sample-synchronized transcoding in terms of objective audio quality.

b. Assuming a uniform distribution of $x_{0[m]}$, the mean cascaded quantization error $E\left[e_{C[m]}^{2}\right]$ can be estimated from $\zeta_{E[m]}$, where we can find $\zeta_{E[m]}$ for different *m* and *d* by performing a prior experiment. An even bigger simplification would be to assume $\overline{\zeta_{E[m]}} = 1.23$.

c. For non-uniform $x_{0[m]}$ distributions with a larger concentration of values towards the zero value than the -1 and +1 values, $\overline{\zeta_{E[m]}}$ should take on a value between 1 and 1.23. This is evident, for example, by considering a triangularly distributed $x_{0[m]}$ (see Figure 9-2(a)). The distribution of $x_{1[m]}$ is then shown in Figure 9-2(b). Since the probability of large values of $|x_{1[m]}|$ is smaller for the triangularly distributed $x_{0[m]}$ case than for the uniformly distributed $x_{0[m]}$ case, we also expect the probability of $|x'_{1[m]}|>1$ to be smaller for the triangularly distributed $x_{0[m]}$ case (and hence $\overline{\zeta_{E[m]}} < 1.23$).

9.3 Analysis for MP3

9.3.1 Overview

A short description of the MP3 compression method is available in Appendix C. Quantization in MP3 is performed very differently from quantization in MPEG 1 Layer 2, and in Section 7.5, this was explained in greater detail. For convenience, we repeat the block diagram (Figure 7-11) showing the MP3 quantization scheme (Figure 9-4). Note that in this diagram, we changed the symbols to appropriately reflect that these signals and processing blocks are used in the second quantization (instead of the first quantization, as was the case for Figure 7-11). For example, $q_{1\{s\}}$ in Figure 7-11 is changed to $q_{2\{s\}}$ in this diagram.

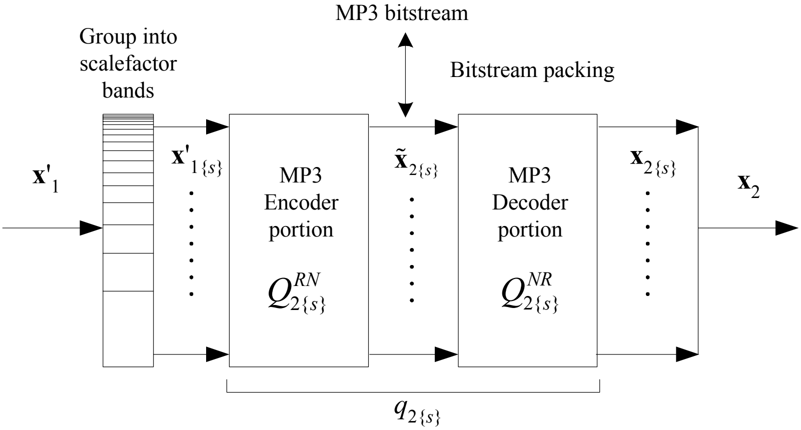


Figure 9-4 (Repeat of Figure 7-11.)

In MP3, a q-frame consists of 576 samples of $x'_{1[m]}$, which are then grouped into its scalefactor bands. We denote the samples which are grouped into a scalefactor band using the vector $\mathbf{x'}_{1\{s\}}$ (s being the index of the scalefactor band). The samples $x'_{1[m]}$ which belong to $\mathbf{x'}_{1\{s\}}$ are quantized by $q_{2\{s\}}$, where the implementation of $q_{2\{s\}}$ is shown in Figure 9-4 and was explained in Section 7.5.2.

Note: When we quantize a sample $x_{0[m]}$ which belongs to the vector $\mathbf{x}_{0\{s\}}$ using $q_{1\{s\}}$, we express this mathematically as $x_{1[m]} = q_{1\{s\}}(x_{0[m]})$. Although it is not explicitly stated in the equation that $x_{0[m]}$ belongs to the vector $\mathbf{x}_{0\{s\}}$, we implicitly assume this for equations of this form.

It is known [83] that for MP3, non-sample-synchronized transcoding is preferred over sample-synchronized transcoding, in relation to the audio quality of the transcoded material.

Note that this is in contradistinction to MPEG 1 Layer 2 audio. In our work, we attempt to explain this by using a measure which we call tandem gain.

9.3.2 Sample-synchronization: Problems posed

The initial quantization in the first encoding process using $q_{1\{s\}}$ (defined by the lumped quantization parameter $\mathbb{Z}_{1\{s\}}$) is described by $x_{1[m]} = q_{1\{s\}}(x_{0[m]})$; or alternatively by $\tilde{x}_{1[m]} = Q^{RN}_{1\{s\}}\left(x_{0[m]}, \mathbb{Z}_{1\{s\}}\right)$ and $x_{1[m]} = Q^{NR}_{1\{s\}}\left(\tilde{x}_{1[m]}, \mathbb{Z}_{1\{s\}}\right)$, where $x_{0[m]}$ and $x_{1[m]}$ are real-valued and $\tilde{x}_{1[m]}$ is integer-valued. The functions $Q^{RN}_{1\{s\}}$ and $Q^{NR}_{1\{s\}}$ were defined in Section 7.5.2.

Similarly, the second quantization in the transcoding process using $q_{2\{s\}}$ (defined by the lumped quantization parameter $\mathbb{Z}_{2\{s\}}$) can be described by $x_{2[m]} = q_{2\{s\}}(x'_{1[m]})$; or alternatively by $\tilde{x}_{2[m]} = Q^{RN}_{2\{s\}}\left(x'_{1[m]}, \mathbb{Z}_{2\{s\}}\right)$ and $x_{2[m]} = Q^{NR}_{2\{s\}}\left(\tilde{x}'_{2[m]}, \mathbb{Z}_{2\{s\}}\right)$, where $x'_{1m]}$ and $x_{2[m]}$ are real-valued and $\tilde{x}_{2[m]}$ is integer-valued. The functions $Q^{RN}_{2\{s\}}$ and $Q^{NR}_{2\{s\}}$ take the same mathematical form as $Q^{RN}_{1\{s\}}$ and $Q^{NR}_{1\{s\}}$.

Since $x'_{1[m]} = x_{1[m]}$ for sample-synchronized transcoding, we can relate $x_{2[m]}$ directly to $x_{1[m]}$ by $x_{2[m]} = q_{2\{s\}}(x_{1[m]})$.

Furthermore, $\tilde{x}_{2[m]}$ can also be directly related to $\tilde{x}_{1[m]}$ by:

$$\tilde{x}_{2[m]} = \pm\Re\left[\,|\,\tilde{x}_{1[m]}\,|\left(2^{-(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})}\right)^{3/4}\right], \tag{9.9}$$

where the sign of $\tilde{x}_{2[m]}$ is taken equal to the sign of $\tilde{x}_{1[m]}$.

The difference in the lumped quantization parameters, $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$, generally determines the amount of bit-rate reduction when transcoded. The larger the difference, the greater is the bit-rate reduction, as the values of $\tilde{x}_{2[m]}$ tend to be smaller and are hence more easily compressible.

The tandem quantization error analysis for MPEG 1 Layer 2 can be extended to MP3. The tandem quantization error regions can be similarly derived for pairs of $q_{1[m]}$ and $q_{2[m]}$, if we take into account the non-uniform quantization step-sizes for MP3. However, in this thesis, we will not go into specific details on tandem quantization error for the case of non-uniform quantizers.

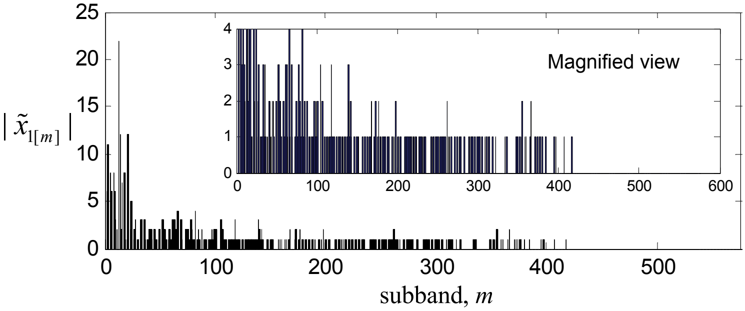
More importantly, apart from tandem quantization error, a serious problem that arises during sample-synchronized cascaded quantization is "tandem gain". We define tandem gain as the ratio of the power of $x_{2[m]}$ to $x_{1[m]}$, when $x_{1[m]}$ is sample-synchronously transcoded to $x_{2[m]}$:

$$P_{R[m]} = x_{2[m]}^2 / x_{1[m]}^2 . \tag{9.10}$$

We also define the regional tandem gain as:

$$P_{R[m_1:m_2]} = \frac{\sum_{m=m_1}^{m_2} x_{2[m]}^2}{\sum_{m=m_1}^{m_2} x_{1[m]}^2}, \tag{9.11}$$

where m_1 and m_2 are the lower and upper bounds of the subband indices that fall within the arbitrarily defined frequency region. Regional tandem gain can be perceived as increased volume of the audio within the defined frequency region during playback.



(Repeat of Figure 7-12.)

Tandem gains are most noticeable for coarsely quantized values, especially in the mid-to-high frequency regions (roughly m>134). As an illustration, take the case when $\tilde{x}_{1[m]} = \pm 1$, which often occurs in the mid-to-high frequency region (roughly m>134). This can be observed in Figure 7-12 (which we repeated above), where we plotted values of $|\tilde{x}_{1[m]}|$ taken from a typical MP3 q-frame. For $q_{1[m]}$ and $q_{2[m]}$ with lumped quantization parameters $\mathbb{Z}_{1\{s\}}$ and $\mathbb{Z}_{2\{s\}}$ respectively, we consider the cases when $0 \le \mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}} \le 4/3$ and when $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}} > 4/3$:

$\underline{0 \le \mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}} \le 4/3}$: Calculating from (9.9), we get $\tilde{x}_{2[m]} = \tilde{x}_{1[m]}$ when $\tilde{x}_{1[m]} = \pm 1$. Since $x_{2[m]} = \pm 2^{-\mathbb{Z}_{2\{s\}}} \left(|\tilde{x}_{2[m]}|\right)^{4/3}$ and $x_{1[m]} = \pm 2^{-\mathbb{Z}_{1\{s\}}} \left(|\tilde{x}_{1[m]}|\right)^{4/3}$, the ratio of $x_{2[m]}^2$ to $x_{1[m]}^2$ is $P_{R[m]} = 2^{2(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})}$. As the difference $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$ increases within the stated range, $P_{R[m]}$ increases.

$\underline{\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}} > 4/3}$: Calculating from (9.9), we get $\tilde{x}_{2[m]} = 0$ when $\tilde{x}_{1[m]} = \pm 1$. The tandem gain is equal to zero.

Note that the event $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}} < 0$ is not particularly interesting in the context of transcoding, because $|\tilde{x}_{2[m]}|$ will tend to be larger than $|\tilde{x}_{1[m]}|$ when this is the case, thus increasing the average bit-rate.

Taking the same values of $\tilde{\mathbf{x}}_1$ as were used in Figure 7-12, we show the effect of different values of $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$ on the values of $\tilde{\mathbf{x}}_2$, in Figure 9-5. Even though it appears in Figure 9-5(b) ($\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}} = 1$) that most of the content in the mid-to-high frequency regions is preserved, note that regional tandem gain is inherent since $\mathbb{Z}_{1\{s\}} \neq \mathbb{Z}_{2\{s\}}$. In Figure 9-5(c) ($\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}} = 1.5$), we observe that most of the spectral content in the mid-to-high frequency regions is lost.

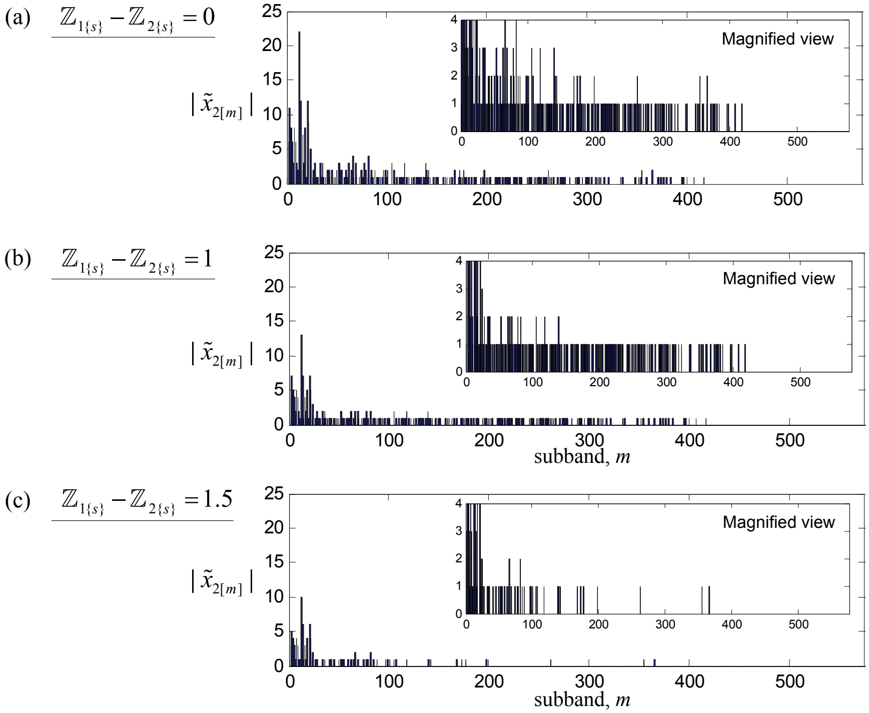


Figure 9-5 Plot of $|\tilde{\mathbf{x}}_2|$ for different values of $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$.

Furthermore, we expect the difference $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$ to be time-varying across consecutive q-frames, since $\mathbb{Z}_{2\{s\}}$ is selected by the algorithm used in transcoding (typically by using a psychoacoustic model analysis), which does not take into consideration the value of $\mathbb{Z}_{1\{s\}}$. Thus, when transcoding sample-synchronously for MP3, we are faced with a problem of fluctuating regional tandem gain. Due to the non-constant regional tandem gain across consecutive q-frames, the power of $x_{2[m]}$ (for values of m which belong to the defined frequency region) fluctuates disproportionately relative to the power of $x_{1[m]}$. Also, when the tandem gain is zero, $x_{2[m]}$ has zero power. We found that fluctuating tandem gain creates an audible and irregular 'loud-soft-loud-soft' sensation (in the affected frequencies). A repeatable experiment can be conducted whereby $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$ is set to 1 and 1.5 for alternating q-frames. For frequencies where $\tilde{x}_{1[m]} = \pm 1$, $\tilde{x}_{2[m]}$ alternates between 0 and ±1 for consecutive q-frames. The result is a highly audible 'on-off-on-off' effect.

We informally tested the audible effect of fluctuating tandem gain on transcoding. Figure 9-6 illustrates the setup that we used. In Case 1, we held the tandem gain constant in the mid-to-high frequency region, over the duration of transcoding from the set of samples from the n-th q-frame, $\mathbf{x}_1(n)$, to $\mathbf{x}_2(n)$ (such as by using the fixed mapping transcoder, which is described in Section 10.4.4). The transcoding parameters were set to give a reasonably large tandem gain in the mid-to-high frequency region, so as to accentuate the effect of the fluctuation. In Case 2, we fluctuated the tandem gain in the mid-to-high frequency region by alternating between 2 transcoding states for consecutive q-frames; the first state (from $\mathbf{x}_1(n)$ to $\mathbf{x}_2(n)$) is defined by using the same transcoder parameters as were used in Case 1 (and hence the same tandem gain as in Case 1), and the second state (from $\mathbf{x}_1(n)$ to $\mathbf{x}_1(n)$, i.e. no change) is defined by not using the transcoder (and hence zero tandem gain). We expect transcoded audio material in Case 2 to have a smaller average error power than in Case 1, since every other frame consists of subband samples taken from the 192 kbps bitstream (not transcoded). Therefore, objectively, transcoded audio material in Case 2 should have an audio quality that is not worse (if not better) than in Case 1. Even so, transcoded audio material in Case 2 generally showed an audio quality that is worse than in Case 1.

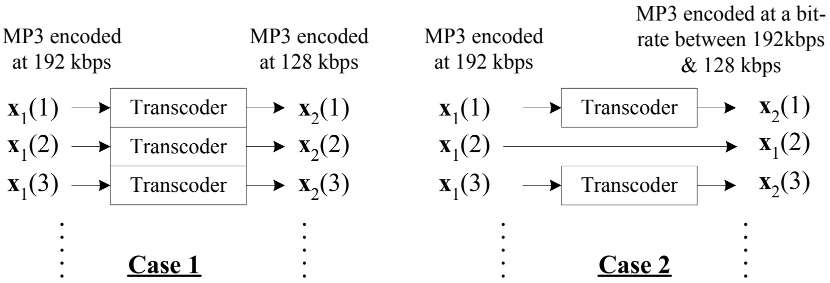


Figure 9-6 Illustration of informal test on fluctuating tandem gain.

For some music material (such as rock music and singing), the difference in audio quality was not easily discernible. For other material (such as orchestral, instrumental, and highly tonal audio material), the difference in audio quality was clearly in favor of Case 1. This result is further substantiated in our formal listening tests in Appendix E (which were conducted to evaluate the transcoders proposed in Chapter 10). As an example (see Figure E-5), for audio material such as the highly tonal harpsichord (Item S-8) and the orchestral music (Item S-9), the fixed mapping transcoder (constant tandem gain) performed better than the conventional transcoder (fluctuating tandem gain).

9.3.3 Sample-synchronization: Possible solutions

A direct solution to the problem of fluctuating regional tandem gain is to avoid sample-synchronized transcoding (by trivially inserting delays, see Section 7.4.3) in the case of

MP3, as the severity of the fluctuating tandem gain problem is greatly reduced for non-sample-synchronized transcoding. This is further explained in the next sub-section.

A possible alternative method to reduce the fluctuating regional tandem gain is to exert a degree of influence on the selection of $\mathbb{Z}_{2\{s\}}$, perhaps in conjunction with the psychoacoustic model. In Section 10.4.4, we propose a method of transcoding that totally bypasses the psychoacoustic model, by fixing $\mathbb{Z}_{2\{s\}}$ for the entire duration of the audio. We also show that a reasonable audio quality (for the transcoded material) can be achieved by using this method, for certain selections of $\mathbb{Z}_{2\{s\}}$.

9.3.4 Non-sample-synchronization

Following the analysis for the MPEG 1 Layer 2 non-sample-synchronized transcoder (Section 9.2.2), we extend the same principles to the analysis here. Again, let us assume a uniform distribution of $x_{0[m]}$ between -1 and 1, for $0 \le m \le 575$. By way of illustration, we assume a 3-step $q_{1\{s\}}$ (by choosing the parameter $\mathbb{Z}_{1\{s\}} = 0$).

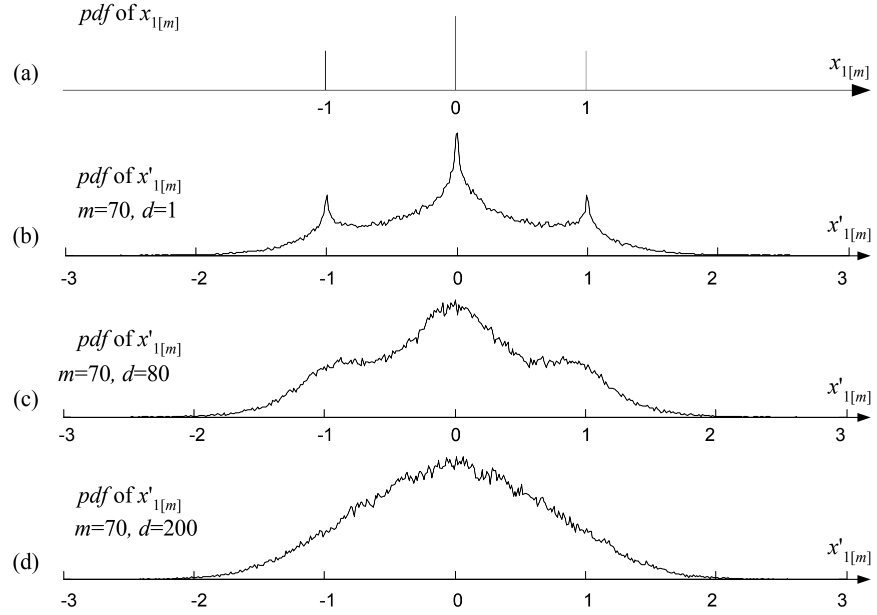


Figure 9-7 Distributions of $x_{1[m]}$ and $x'_{1[m]}$ for different values of delay d.

The distribution of $x_{1[m]}$ is shown in Figure 9-7(a). Note that the probabilities of $x_{1[m]}$ are not constant because $q_{1\{s\}}$ is non-uniform. When the transcoding is non-sample-synchronized, we expect the distribution of $x'_{1[m]}$ to be 'dispersed' in a similar fashion as for the MPEG 1 Layer 2 method. We show the distributions of $x'_{1[m]}$ for different values of d in Figure 9-7 (b)-(d).

There are some advantages to be gained by 'dispersing' the distribution of $x'_{1[m]}$:

1. Firstly, the tandem gain problem is greatly reduced. We illustrate this with an example using $x_{1[m]}$ with distribution shown in Figure 9-7(a), and the 'dispersed' $x'_{1[m]}$ with distribution shown in Figure 9-7(d). For $\mathbb{Z}_{1\{s\}}=0$, when:

   i. $\underline{|x'_{1[m]}|\le 0.5^{4/3}2^{-\mathbb{Z}_{2\{s\}}}}$: $|\tilde{x}_{2[m]}|=0$, and $|x_{2[m]}|=0$;

   ii. $\underline{0.5^{4/3}2^{-\mathbb{Z}_{2\{s\}}}<|x'_{1[m]}|\le 1.5^{4/3}2^{-\mathbb{Z}_{2\{s\}}}}$: $|\tilde{x}_{2[m]}|=1$, and $|x_{2[m]}|=2^{-\mathbb{Z}_{2\{s\}}}$;

   and in general,

   iii. $\underline{\left(|k|-0.5\right)^{4/3}2^{-\mathbb{Z}_{2\{s\}}}<|x'_{1[m]}|\le\left(|k|+0.5\right)^{4/3}2^{-\mathbb{Z}_{2\{s\}}}}$: $|\tilde{x}_{2[m]}|=|k|$, and $|x_{2[m]}|=2^{-\mathbb{Z}_{2\{s\}}}|\tilde{x}_{2[m]}|$, where k is an integer.

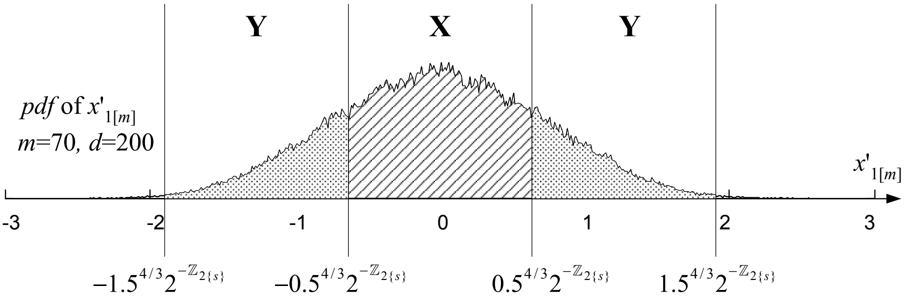
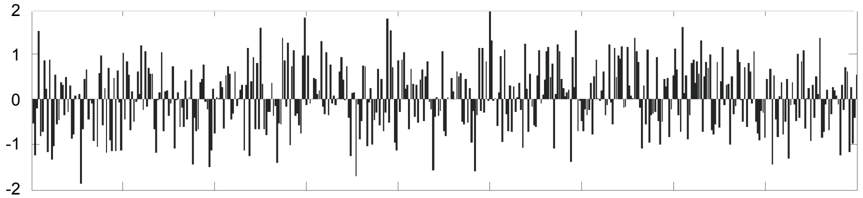


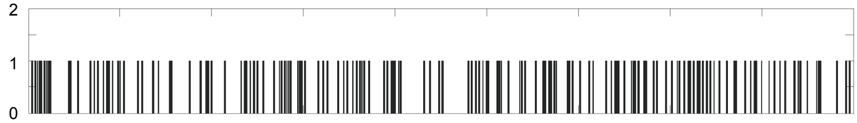
Figure 9-8 Quantization of dispersed $x'_{1[m]}$.

Only values of $x'_{1[m]}$ falling in the shaded region (labeled by '**X**') with probability $p\left(|x'_{1[m]}|\le 0.5^{4/3}2^{-\mathbb{Z}_{2\{s\}}}\right)$ (see Figure 9-8) would be quantized to 0. Values of $x'_{1[m]}$ falling in the shaded regions labeled by '**Y**' would be quantized to 1, etc. Changing the value of $\mathbb{Z}_{2\{s\}}$ changes the probability that $x'_{1[m]}$ is quantized to 0 or 1. Contrast this with the sample-synchronized case, where the probability $p\left(|x_{1[m]}|\le 0.5^{4/3}2^{-\mathbb{Z}_{2\{s\}}}\right)$ is equal to either 0 (when $\mathbb{Z}_{2\{s\}}<-4/3$), or 1 (when $-4/3\le\mathbb{Z}_{2\{s\}}\le 0$).

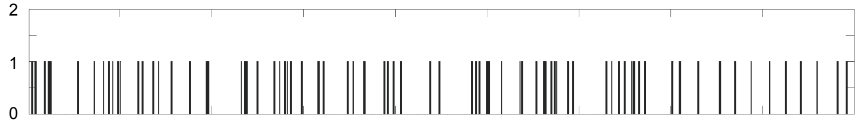
For the mid-to-high frequency region (say $134\le m\le 418$), the 'on-off-on-off' effect is eliminated, since $p\left(|x_{2[m]}|=0\right)$ changes gradually with $\mathbb{Z}_{2\{s\}}$. This is illustrated in Figure 9-9, where we show a statistically generated plot of $x'_{1[m]}$ with $\mathbb{Z}_{1\{s\}}=0$, and the subsequent quantized values of $|\tilde{x}_{2[m]}|$ for different $\mathbb{Z}_{2\{s\}}$. The values of $x'_{1[m]}$ are obtained by first generating $x_{0[m]}$ (uniformly distributed between -1 and 1), which are then quantized using $q_{1\{s\}}$ (with parameter $\mathbb{Z}_{1\{s\}}=0$), and then dispersed by processing with the transcoder filter bank blocks.



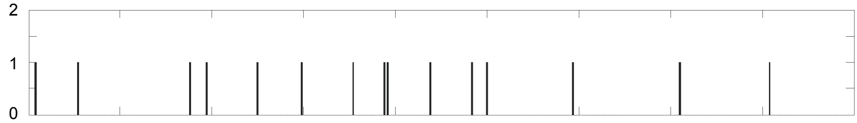
(a) Plot of $x'_{1[m]}$



(b) Plot of $\left|\tilde{x}_{2[m]}\right|$, $\mathbb{Z}_{2\{s\}} = -0.5$

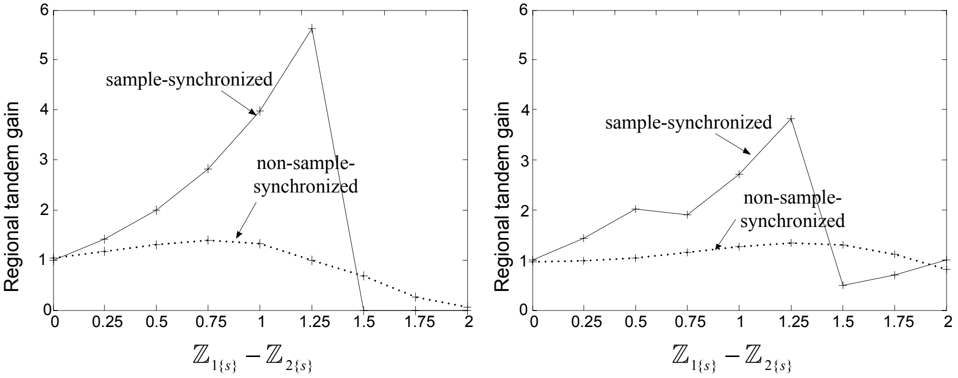


(c) Plot of $\left|\tilde{x}_{2[m]}\right|$, $\mathbb{Z}_{2\{s\}} = -1$



(d) Plot of $\left|\tilde{x}_{2[m]}\right|$, $\mathbb{Z}_{2\{s\}} = -1.5$

Figure 9-9 Effect of different $\mathbb{Z}_{2\{s\}}$, with $\mathbb{Z}_{1\{s\}} = 0$, on $|\tilde{x}_{2[m]}|$ for non-sample-synchronized transcoding.



(a) $\mathbb{Z}_{1\{s\}} = 0$

(b) $\mathbb{Z}_{1\{s\}} = 1$

Figure 9-10 Estimated regional tandem gain for different values of $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$.

The fluctuating regional tandem gain (for the frequency region $134 \le m \le 418$) is also less severe when compared to the sample-synchronized case. In Figure 9-10, we plot the estimated regional tandem gain (obtained by means of simulations) $P_{R[134:418]}$ for the case when $\mathbb{Z}_{1\{s\}} = 0$ (i.e. $|\tilde{x}_{1[m]}| \in \{0, 1\}$) and $\mathbb{Z}_{1\{s\}} = 1$ (i.e. $|\tilde{x}_{1[m]}| \in \{0, 1, 2\}$). It can be seen that $P_{R[134:418]}$ in the non-sample-synchronized case is much more stable with respect to changes in the values of $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$.

2. Secondly, as $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$ is gradually increased, we expect the quantization using $q_{2\{s\}}$ to gradually decrease the mean value of $|\tilde{x}_{2[m]}|$, as demonstrated in Figure 9-9. This leads to a gradual bit-rate reduction in the mid-to-high frequency region when $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$ is gradually increased. On the other hand, for the sample-synchronized case, the change in bit-rate in the mid-to-high frequency region is expected to be abrupt due to the abrupt change in the mean value of $|\tilde{x}_{2[m]}|$ with $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$, as demonstrated in Figure 9-5.

9.4 Remarks

We discussed bit-rate transcoding for the case of MPEG 1 Layer 2 to MPEG 1 Layer 2, and for the case of MP3 to MP3, and the effect of sample-synchronization on their audio quality. The reason why we chose to analyze the MP3 to MP3 case was partly because of our work in Chapter 10 on MP3 transcoding, but also very importantly, because same-format sample-synchronized transcoding is known to exhibit a lower audio quality than can be explained by the perceptually-weighted quantization error power alone. The MPEG 1 Layer 2 case was presented as a counterpoint to the MP3 case, as an example when sample-synchronized transcoding tends to lead to a better audio quality compared to non-sample-synchronized transcoding.

For same-format bit-rate transcoding of other compressed audio (e.g. from AAC to AAC), the same issues that are discussed here may also be used to explain the effect of sample-synchronization on audio quality. For AAC, as an example, the quantization issues are similar to that for MP3, in that mid-to-high frequency subband samples are very coarsely quantized. Sample-synchronization is known to lead to a lower transcoded audio quality for AAC, and this can be explained by the fluctuating tandem gain effect as observed for the case of MP3. However, as we did not look into transcoding for these other compression methods, there may also be other important factors causing poor audio quality that we are unaware of.

Our analyses on tandem quantization error and sample-synchronization in Chapters 8 and 9 assume that the synthesis block of the decoder and the analysis block of the encoder form a PR or NPR pair. This is the case for same-format bit-rate transcoding. For cross-format transcoding, such as from MP3 to AAC, this is usually not true. Hence, the sample-synchronization condition does not exist for these cases, and our analyses do not apply.

9.5 Summary

The impact of sample-synchronization on audio transcoding was investigated for the MPEG 1 Layer 2 and MP3 compression methods.

For MPEG 1 Layer 2, we proposed a measure termed the effective scaling factor $\zeta_{E[m]}$, which can be used to estimate the amount of quantization error incurred during transcoding. The effective scaling factor was related to the normalization scaling of the subband signals prior to quantization. Assuming the input $x_{0[m]}$ is uniformly distributed between -1 and 1, the effective scaling factor was found to be approximately 1.23. For non-uniform input with distributions that have a higher concentration of values towards zero (such as triangular distribution), the effective scaling factor lies between unity and 1.23. The increased value of $\zeta_{E[m]}$ leads to a larger error power during non-sample-synchronized transcoding and hence a lower objective quality of the transcoded audio, when compared to sample-synchronized transcoding.

For MP3, we found several problems associated with audio that was transcoded sample-synchronously. MP3 subband samples in the mid-to-high frequency region are usually quantized using very coarse quantizers. In this frequency region, the subband components suffer from a condition which we call fluctuating tandem gain, due to a time-varying $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$, where $\mathbb{Z}_{1\{s\}}$ and $\mathbb{Z}_{2\{s\}}$ are the lumped quantization parameters for the quantizers $q_{1\{s\}}$ and $q_{2\{s\}}$ respectively. Fluctuating tandem gain can give rise to an irregular and audible 'loud-soft-loud-soft' sensation during audio playback.

On the other hand, for MP3 audio material which are transcoded non-sample-synchronously, the 'dispersal' of the distribution of the subband samples was found to be favourable, as it allows for a smoother degradation of audio quality and bit-rate reduction with respect to $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$. The regional tandem gain was also found to be more stable in the mid-to-high frequency range with respect to changes in $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$.

Chapter 10

An Ultra-fast Bit-rate Transcoder for MP3

In conventional audio transcoding (from a higher bit-rate to a lower bit-rate), a cascaded decoder-encoder is usually applied. In Chapter 9, we examined the effect of different input-to-output transcoder delays on the quality of the transcoded material. For some audio compression methods (such as MPEG 1 Layer 2), a sample-synchronized transcoder is preferred. For other methods (such as MP3), a non-sample-synchronized transcoder is preferred.

The conventional cascaded decoder-encoder is very inefficient in terms of number of operations required and the total execution time. In this chapter, we propose an ultra-fast bit-rate transcoder for MP3. It assumes that the transcoding is frame-synchronous (as defined in Section 7.4.4), and does not require a psychoacoustic model. Although it was shown (Section 9.3) that MP3 audio material which is transcoded sample-synchronously tends to have a lower audio quality than material that is transcoded non-sample-synchronously, we impose certain controls to address the problem of tandem gain (Section 9.3.2). Results show that the proposed transcoder compares favorably with existing transcoders in terms of audio quality, while affording significant reductions in the operating complexity and total execution time.

The organization of this chapter is as follows: First, an introduction to the current methods of bit-rate transcoding is given. We proceed to describe the proposed method of bit-rate transcoding. The results of this new method are then presented, and finally possibilities for further developments are suggested. The transcoder was implemented using the C programming language, and its execution time and audio quality were tested. The new method was described in a patent application [104].

10.1 Audio bit-rate transcoding

10.1.1 State-of-the-art

Audio Layered Transcoder for fast transcoding

Some prior work on audio transcoding is of interest to this chapter. In [99], an 'Audio Layered Transcoder', based on the MPEG 1 Layer 2 compression method, was proposed for fast bit-rate transcoding (mainly directed towards internet-distributed audio applications), by restructuring the way in which data is stored in the bitstream.

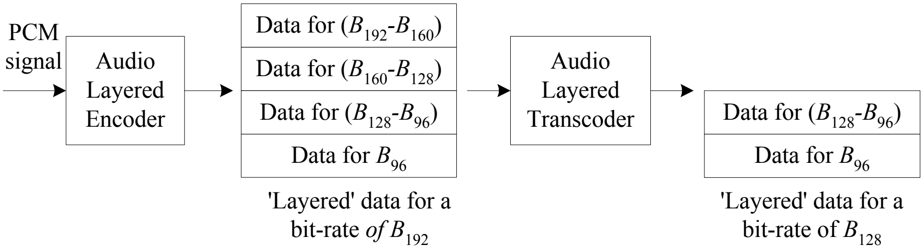

PCM signal
Audio Layered Encoder
Data for (B_{192}-B_{160})
Data for (B_{160}-B_{128})
Data for (B_{128}-B_{96})
Data for B_{96}
'Layered' data for a bit-rate of B_{192}
Audio Layered Transcoder
Data for (B_{128}-B_{96})
Data for B_{96}
'Layered' data for a bit-rate of B_{128}

Figure 10-1 Data truncation using audio layered transcoding.

The operation of the Audio Layered Transcoder can be explained using Figure 10-1. The PCM input is first encoded using the Audio Layered Encoder, which comprises several standard MPEG 1 Layer 2 encoder processing blocks (such as filter banks, psychoacoustic model, etc.). However, in order to structure the data in a layered form, proprietary quantizers are used. The data is arranged such that each additional layer stores extra information which provides an increasingly higher precision. Transcoding of the 'layered data' can then be performed quickly by the simple procedure of removing unwanted 'layers'. The diagram shows the transcoding from a bit-rate of B_{192} (192 kbps) to B_{128} (128 kbps).

The above description can be visualized by considering an analogous example of the 'layered' quantized number 0.111. The first layer contains the value 0.1, the second layer contains 0.01 and the third layer contains 0.001. To 'transcode' from 3 layers to 2 layers, we simply have to remove the 3rd layer. The result is then 0.11. The drawback of this method is that modified decoders are required to decode the layered data.

Use of ancillary data for better audio quality

Some 'ancillary' data bits are allocated within the MPEG (including Layer 2 and MP3) bitstream for custom usage (refer to the specifications [71]). The number of ancillary data bits can be arbitrarily defined by the user. In normal encoding and decoding, these bits are not utilized. The ancillary data provide the flexibility for custom encoders and decoders to insert additional information into the bitstream, at the cost of a higher bit-rate.

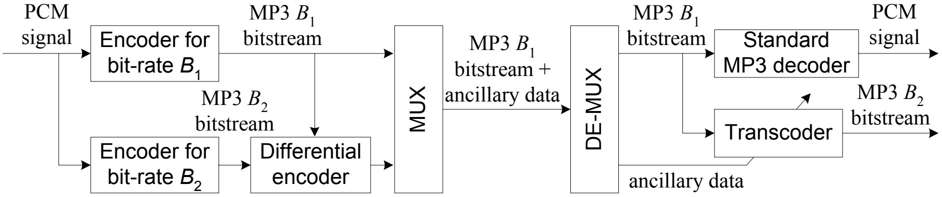


Figure 10-2 Using ancillary data for transcoding.

In [100]-[101], methods were proposed that use the ancillary data space to store additional information during the initial encoding (using a custom encoder). This data is then used to guide future transcoding, with the objective of improving the output audio quality. Figure 10-2 shows a simple overview of the described methods. A-priori knowledge of both the higher and lower target bit-rates (bit-rates B_1 and B_2 respectively) is assumed. In the initial stage, encoding of both B_1 and B_2 bitstreams take place. The differences between these bitstreams are encoded, and multiplexed as ancillary data with the B_1 bitstream. When transcoding the multiplexed B_1 bitstream to a bit-rate of B_2, the ancillary data is used as helper information to guide the transcoding (using a custom transcoder). Alternatively, the multiplexed bitstream can be read by a standard decoder, by ignoring the ancillary data.

This method has a few drawbacks however. Extra bits are required to store the ancillary data, the PCM source must be available, and both the bit-rates B_1 and B_2 must be known a-priori. A trade-off between the increase in the bit-rate of the multiplexed B_1 bitstream and the audio quality of the transcoded B_2 bitstream exists. The method is useful in situations where the prospect of future transcoding is certain, and transcoded audio quality is vital, such as in music distribution [96]. Furthermore, in music distribution applications, the conditions that the PCM source is available, B_1 and B_2 are known a-priori, are satisfied.

Efficient transcoding by a direct requantization of data

The method of transcoding (using a 'bit-rate scaling in coded data domain') proposed in [102] resembles the 'ultra-fast bit-rate transcoding method' that is discussed later on in this chapter. Figure 10-3 shows the block diagram for the bit-rate scaler (compare with the ultra-fast transcoder in Figure 10-7). Computation-intensive processing such as the filter bank blocks were removed. The bit-rate scaler extracts the values of $x_{1[m]}$ from the bitstream (by Huffman decoding), quantizes $x_{1[m]}$ to $x_{2[m]}$ by using $q_{2[m]}$, and then processes $x_{2[m]}$ into the output bitstream (by Huffman encoding). This process of transcoding is sample-synchronous. The bit-rate scaler was applied successfully in the quoted literature for MPEG 1 Layer 2 transcoding.

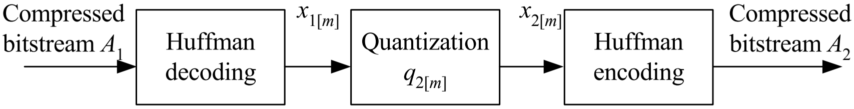


Figure 10-3 Block diagram for bit-rate scaling in coded data domain.

The method of quantizing $x_{1[m]}$ to $x_{2[m]}$ (i.e. by choosing $q_{2[m]}$) that is proposed in this paper is straightforward. Starting from the largest *m* (highest frequency) down to the smallest *m* (lowest frequency), $q_{2[m]}$ is chosen to have a wordlength of 1 bit less than $q_{1[m]}$, where $q_{1[m]}$ was used in the initial encoding process to quantize the source $x_{0[m]}$ to $x_{1[m]}$. This quantizer wordlength reduction is iteratively performed until the output bitstream satisfies the target bit-rate requirement.

Note that the bit-rate scaler benefits from the fact that sample-synchronized transcoding is favorable for the case of MPEG 1 Layer 2. For MP3, successful sample-synchronized transcoding requires the consideration of more factors (see Section 9.3 for a more complete analysis). Transcoded material typically exhibits poor audio quality when the wordlengths of all the quantizers $q_{1[m]}$ are reduced by an equal amount (see also Section 10.4.3, which describes some experiments to verify this statement). For MP3, the mid-to-high frequency components are deemed to be more important than the low frequency components, due to the very coarse quantizers that are used in this region. The bit-rate scaler method, on the other hand, tends to reduce the resolution of high frequency components prior to the low frequency components. Furthermore, since the high frequency components for MP3 usually comprise only 1 or 2 quantization levels, the bit-rate scaler tends to completely remove these components.

Our ultra-fast transcoder focuses on MP3. For the quantization process, we impose a form of control both within the same q-frame (by placing greater importance on the mid-to-high frequency components), and across consecutive q-frames (by controlling fluctuating tandem gain). We also propose a lookup table method for the quantization process, which further reduces the computation time. We also demonstrate that the ultra-fast transcoder performs reasonably well for a variety of MP3 audio material.

Other methods

Several proprietary compression methods that evolve from the MP3 compression method, including [103], make use of additional audio descriptors, such as 'music beat', and encode them together with the primary bitstream. The descriptors can be utilized by compatible transcoders to improve audio quality. However, these methods are often incompatible with standard encoders and decoders.

10.1.2 Objectives of the transcoder

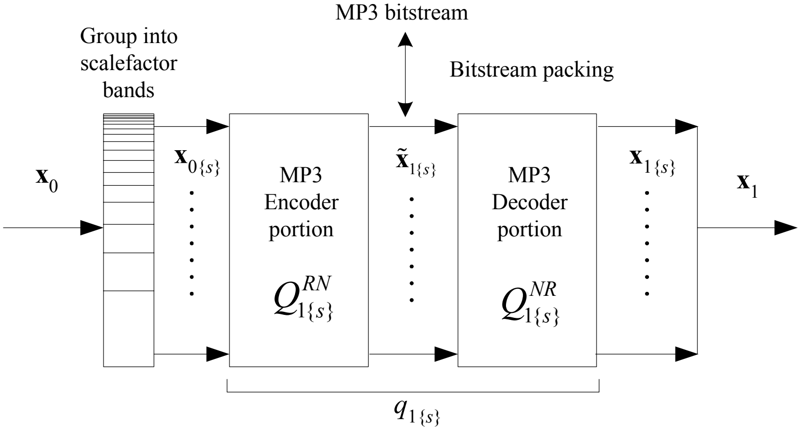
The main design objectives of the transcoder are:

> (i). Targeted for the MP3 compression method. Since there are many compression methods available in the current market which are incompatible with each other, we need to select one for our implementation. MP3 is currently the most popular, and is viable for market-oriented development.
>
> (ii). Low implementation complexity. A more complex design requires a larger memory footprint in the case of a software implementation and a larger chip area and power consumption in the case of a hardware implementation.
>
> (iii). High execution speed. The design should be streamlined as much as possible to optimize the transcoding speed.
>
> (iv). Preservation of audio quality. The transcoded material should maintain a reasonable reproduction quality during play back.

10.1.3 MP3 encoding

Both commercial and freeware MP3 encoder and decoder software are widely available. In our work, we focus on the highly popular LAME 3.95 software [108], MADPLAY [109] (which we found to be a very fast decoder), and the Philips Fast MP3 encoder [110] (a very fast MP3 encoder). In Appendix C, we highlighted important aspects of the MP3 method which are relevant to our work. A full description of the MP3 method is available in [71]. In

Section 7.5.2, we further explained the notations used for the MP3 quantization. For convenience, we briefly highlight the important details below.



(Repeat of Figure 7-11.)

In Figure 7-11, the subband components $x_{0[m]}$ (shown in the diagram as the vector $\mathbf{x}_0$, comprising $x_{0[m]}$ for $0 \le m \le 575$) are grouped into scalefactor bands. In the 'group into scalefactor bands' block, the horizontal lines pictorially show the allocated widths of the scalefactor bands (refer to Appendix C, Figure C-2 for more information). Those $x_{0[m]}$ (shown in the diagram as the vector $\mathbf{x}_{0\{s\}}$) which fall within the scalefactor band denoted by s, are then quantized by $q_{1\{s\}}$ to $\mathrm{x}_{1[m]}$ (shown in the diagram as the vector $\mathbf{x}_{1\{s\}}$). The selection of $q_{1\{s\}}$ typically depends on the psychoacoustic model analysis (not shown in diagram).

The quantizer represented by $q_{1\{s\}}$ is defined by the parameter $\mathbb{Z}_{1\{s\}}$:

$$\mathbb{Z}_{1\{s\}} = \mathbb{Z}_{g1} / 4 - \mathbb{Z}_{\alpha 1}\mathbb{Z}_{f1\{s\}} + \mathbb{Z}_{\phi 1}, \tag{10.1}$$

where the parameters $\mathbb{Z}_{g1}$, $\mathbb{Z}_{\alpha 1}$, $\mathbb{Z}_{f1\{s\}}$ and $\mathbb{Z}_{\phi 1}$ are the values of the global gain, scalefactor multiplier, scalefactor and other encoder-specific variables respectively (see [71] for more information).

The quantization is given by:

$$x_{1[m]} = \pm 2^{-\mathbb{Z}_{1\{s\}}} \left[\Re\left(\left(2^{\mathbb{Z}_{1\{s\}}} \, | x_{0[m]} | \right)^{3/4} \right) \right]^{4/3}, \tag{10.2}$$

where $\Re(.)$ is the rounding operator, and the sign of $x_{1[m]}$ is taken equal to the sign of $x_{0[m]}$.

Furthermore:

$$x_{1[m]} = \pm 2^{-\mathbb{Z}_{1\{s\}}} \left(| \tilde{x}_{1[m]} | \right)^{4/3}, \tag{10.3}$$

where $\tilde{x}_{1[m]}$ is the integer part of $x_{1[m]}$ which is stored in the MP3 bitstream, and the sign of $\tilde{x}_{1[m]}$ is taken equal to the sign of $x_{1[m]}$.

Just as in Chapters 8 and 9, we hide the time variables n and η in our equations. Also note that for MP3, a q-frame consists of 576 subband samples, and an MP3 frame consists of 2 q-frames (1152 subband samples).

10.1.4 Conventional method of bit-rate transcoding

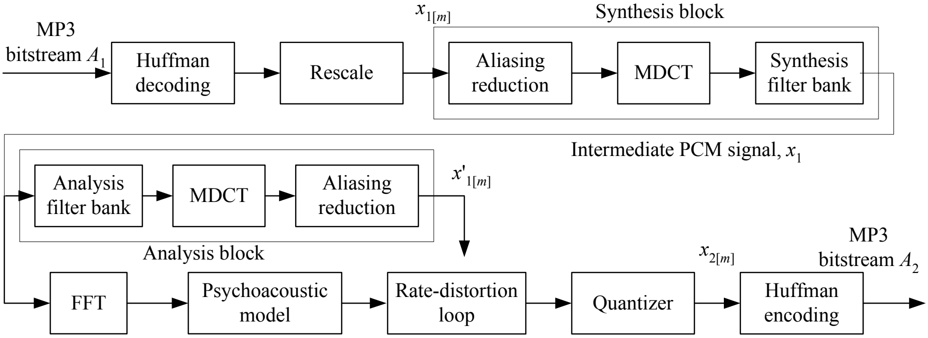


Figure 10-4 Conventional implementation of an MP3 transcoder.

Figure 10-4 shows the block diagram of the conventional method of transcoding using a cascaded decoder-encoder. Descriptions of the individual blocks can be found in Appendix C. This method is straightforward, and its advantage is that it can be implemented easily by using readily available software, such as the LAME software package (by first decoding from MP3 to a PCM signal, and then re-encoding to MP3). This method is computationally intensive. Implementations using this method typically focus on reducing the complexity of the individual blocks, such as the filter bank blocks [105].

Using the Fast MP3 encoder and decoder programs as basis, a rough indication of the relative complexities of the different processing blocks is as follows:

Filter bank blocks	20-30%
Huffman encoding and decoding	20-30%
Psychoacoustic model + rate-distortion loop + quantizer	50%

Note that these relative complexities only provide crude estimates. The actual complexities for different software implementations and different audio material (having different bit-rates, for example) can vary significantly. As an example, for the LAME encoder and decoder, the psychoacoustic model + rate-distortion loop + quantizer can account for more than 80% of the total complexity.

10.2 Complexity reduction I: Frame-synchronization

Frame-synchronization was defined in Section 7.4.4. For the MP3 method, frame-synchronized transcoding can be realized by delaying the intermediate PCM signal (refer to Figure 10-5), so that the total input-to-output delay is a multiple of 1152 (the size of an MP3 frame). Since this number is also a multiple of 576 (the size of a q-frame, also the number of subbands, and the decimation factor) sample-synchronization is realized by frame-synchronized transcoding.

If $x'_{1[m]} = x_{1[m]}$, we can directly route $x_{1[m]}$ to the rate-distortion loop block as shown in Figure 10-5. The intermediate PCM signal is used as input for the FFT and psychoacoustic model. The analysis block can be removed. The resultant complexity, compared to the conventional transcoder, is approximately reduced by a factor of 1.15 (based on using the Fast MP3 programs).

For MP3, the analysis and synthesis blocks form NPR pairs. Therefore, $x'_{1[m]} \approx x_{1[m]}$ to a small degree of error. This error is much smaller than the quantization step-size of $q_{1[m]}$ and $q_{2[m]}$. Realistically, this means that $q_{2[m]}\left(x'_{1[m]}\right) = q_{2[m]}\left(x_{1[m]}\right)$ for almost all values of $x_{1[m]}$. This complexity reduction is therefore also applicable to NPR analysis-synthesis blocks, in the context of audio transcoding.

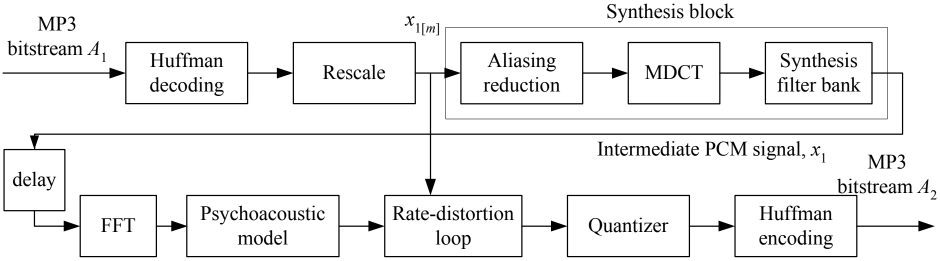


Figure 10-5 Complexity reduction I: Frame-synchronization.

10.3 Complexity reduction II: Modified psychoacoustic model

The original psychoacoustic model uses the 1024-point complex-valued FFT of the PCM signal to analyze the audio content. By using a modified psychoacoustic model which can directly make use of the 576 real-valued subband samples $x_{1[m]}$, the complexity of the frame-synchronized transcoder can be further reduced. Such modified psychoacoustic models have been proposed for the MPEG 1 Layer 2 and MP3 methods [106], and for the MPEG 2 AAC method [107].

Figure 10-6 shows the block diagram for a frame-synchronized transcoder using a modified psychoacoustic model. The synthesis block has been further removed (compared to Figure 10-5), and $x_{1[m]}$ can be directly fed to both the modified psychoacoustic model and the rate-distortion loop. The resultant complexity, compared to the conventional transcoder, is approximately reduced by a factor of 1.3 to 1.5 (based on using the Fast MP3 programs).

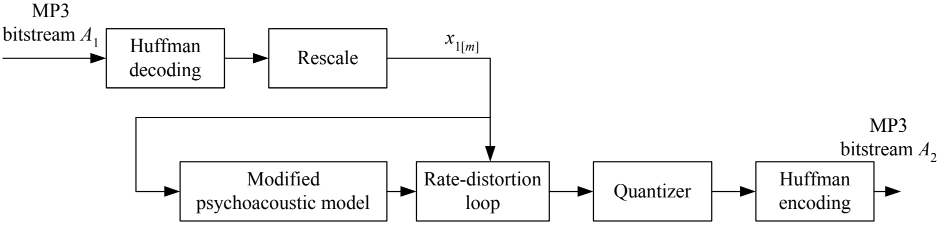


Figure 10-6 Complexity reduction II: Modified psychoacoustic model.

10.4 Complexity reduction III: Direct mapping

In conventional transcoding methods, some form of psychoacoustic analysis is typically used to drive the quantization process. We call a transcoding process in which psychoacoustic models are used to direct quantizer selection a psychoacoustic-centric transcoding process. Conversely, a non-psychoacoustic-centric transcoding process does not rely on psychoacoustic models to direct the quantizer selection. The proposed method of transcoding differs from conventional methods by using a non-psychoacoustic-centric transcoding process.

10.4.1 Overview of method

From our analysis in Section 9.3 on sample-synchronized transcoding, simply selecting the quantizers using a psychoacoustic-centric process is not sufficient to ensure good reproduction quality. The fluctuation of tandem gain across consecutive MP3 frames should also be considered. Furthermore, since the audio has already been compressed using a psychoacoustic model in the initial encoding process (by the selection of $q_{1\{s\}}$), we question whether it is necessary to reapply the psychoacoustic model in the transcoder. Instead, we propose that the quantizer $q_{2\{s\}}$ should be selected based on the quantizer $q_{1\{s\}}$ that was selected, using a defined set of rules. Then, $\tilde{\mathbf{x}}_1$ can be mapped directly to $\tilde{\mathbf{x}}_2$ based on the defined set of rules, using only information which is available in the MP3 bitstream A_1.

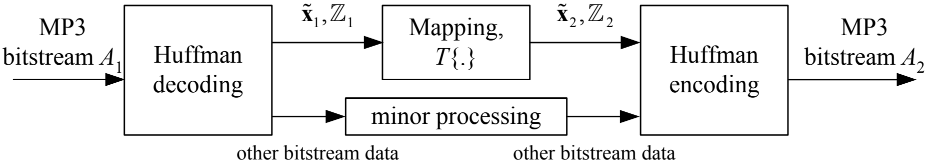


Figure 10-7 Complexity reduction III: Direct mapping.

Note: The following notational conventions are used: $\mathbf{x}$ represents a vector of samples $x_{[m]}$ (for $0 \le m \le 575$); $\tilde{x}_{[m]}$ is the integer component of $x_{[m]}$ (refer to (10.4)); $\mathbb{Z}_{\{s\}}$ is a parameter defining the quantizer $q_{\{s\}}$(for $0 \le s \le 21$).

Supposing that the psychoacoustic model can be taken out, the transcoder can then be reduced to the 3-step operation which is shown in Figure 10-7. The MP3 bitstream A_1 is first Huffman-decoded. The integer vector $\tilde{\mathbf{x}}_1$ and the quantizer parameters $\mathbb{Z}_1 = \left\{\mathbb{Z}_{1\{s\}}, \text{ for } 0 \le s \le 21\right\}$ are then extracted, and mapped to $\tilde{\mathbf{x}}_2$ and $\mathbb{Z}_2$. We denote the mapping by the symbol $T\{.\}$. Finally, $\tilde{\mathbf{x}}_2$, $\mathbb{Z}_2$ and other bitstream data are Huffman-encoded to the MP3 bitstream A_2. Other bitstream data refers to the MP3 frame header and side info (Appendix C). Minor processing is required, such as changing the header information to reflect the new bit-rate.

The complexity of the resultant transcoder is very low since:

(i). Most of the computation-intensive blocks have been removed, e.g. psychoacoustic model, filter bank blocks, and rate-distortion loop.

(ii). The mapping $T\{.\}$ can be efficiently performed in the integer domain. Integer-to-floating point conversions, floating point-to-integer conversions, and floating point operations can be avoided. This is further explained below.

10.4.2 Simplification of quantization operation

Note: In an equation involving both m and s, e.g. $x_{1[m]} = q_{1\{s\}}(x_{0[m]})$, we assume $m_s \le m < m_{s+1}$, where m_s refers to the index of the subband corresponding to the first element of the scalefactor band s (refer to (C.2)); i.e. although it is not explicitly stated in the equation that $x_{0[m]}$ belongs to the vector $\mathbf{x}_{0\{s\}}$, we implicitly assume this for equations of this form.

For the MP3 method, $x_{1[m]}$ is first obtained from $\tilde{x}_{1[m]}$:

$$x_{1[m]} = \pm\left|\tilde{x}_{1[m]}\right|^{4/3} \cdot 2^{-\mathbb{Z}_{1\{s\}}}, \tag{10.4}$$

where $\mathbb{Z}_{1\{s\}} = -\mathbb{Z}_{g1}/4 + \mathbb{Z}_{\alpha 1}\mathbb{Z}_{f1\{s\}} - \mathbb{Z}_{\phi 1}$ and the sign of $x_{1[m]}$ is taken equal to the sign of $\tilde{x}_{1[m]}$.

Then, $x_{1[m]}$ is quantized to $x_{2[m]}$, having the integer component $\tilde{x}_{2[m]}$:

$$\tilde{x}_{2[m]} = \pm\Re\left[\left(2^{\mathbb{Z}_{2\{s\}}}\,|\,x_{1[m]}\,|\right)^{3/4}\right], \tag{10.5}$$

where $\mathbb{Z}_{2\{s\}} = -\mathbb{Z}_{g2}/4 + \mathbb{Z}_{\alpha 2}\mathbb{Z}_{f2\{s\}} - \mathbb{Z}_{\phi 2}$, $\Re(.)$ is the rounding operator, and the sign of $\tilde{x}_{2[m]}$ is taken equal to the sign of $x_{1[m]}$.

Therefore, we can relate $\tilde{x}_{2[m]}$ directly to $\tilde{x}_{1[m]}$ by:

$$\tilde{x}_{\mathbf{2}[m]} = \Re\left\{\tilde{x}_{1[m]}\left(2^{-(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})}\right)^{3/4}\right\}. \tag{10.6}$$

Let us set $\mathbb{Z}_{\alpha 2} = \mathbb{Z}_{\alpha 1}$ and $\mathbb{Z}_{\phi 2} = \mathbb{Z}_{\phi 1}$, so that the scalefactor multiplier and the fine-tuning variables are unchanged by the transcoder. Then:

$$\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}} = -(\mathbb{Z}_{g1} - \mathbb{Z}_{g2})/4 + \mathbb{Z}_{\alpha 1}(\mathbb{Z}_{f1\{s\}} - \mathbb{Z}_{f2\{s\}}) \ , \tag{10.7}$$

where $\mathbb{Z}_{\alpha 1} = 0.5$ or 1 (defined by the MP3 specifications).

Define:

$$\delta_{f\{s\}} = \mathbb{Z}_{f2\{s\}} - \mathbb{Z}_{f1\{s\}}, \tag{10.8}$$

and:

$$\delta_g = \mathbb{Z}_{g2} - \mathbb{Z}_{g1}. \tag{10.9}$$

From the MP3 specifications, the global gains ($\mathbb{Z}_{g1}$ and $\mathbb{Z}_{g2}$) and scalefactors ($\mathbb{Z}_{f1\{s\}}$ and $\mathbb{Z}_{f2\{s\}}$) take on a discrete range of values. The global gains have discrete steps of 0.25 ($0 \le (\mathbb{Z}_{g1}, \mathbb{Z}_{g2}) \le 255$) and the scalefactors have discrete steps of 0.5 ($0 \le (\mathbb{Z}_{f1\{s\}}, \mathbb{Z}_{f2\{s\}}) \le 15$). Therefore, $(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})$ takes on the set of possible values from {-78.75, ..., -0.5, -0.25, 0, 0.25, 0.5, 0.75, ...,78.75}. Hence, the mapping $T\{.\}$ is limited to a finite space.

Note that the Huffman-encoder used in the MP3 encoder allocates statistically fewer bits to store smaller integer values of $|\tilde{x}_{1[m]}|$ and $|\tilde{x}_{2[m]}|$, and statistically more bits to store larger integer values of $|\tilde{x}_{1[m]}|$ and $|\tilde{x}_{2[m]}|$. If $(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}) < 0$, then $\left|\tilde{x}_{2[m]}\right| \ge \left|\tilde{x}_{1[m]}\right|$ and thus $|\tilde{x}_{2[m]}|$ is likely to require more bits than $|\tilde{x}_{1[m]}|$. Since we are interested in transcoding from a higher bit-rate to a lower bit-rate, $(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})$ is usually larger than 0.

On the other hand, if $(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})$ becomes too large, (i.e. if $q_{2\{s\}}$ is a very coarse quantizer) then the quantization error in $x_{2[m]}$ may become unacceptable. In fact, the set of meaningful values of $(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})$ typically consists of only about 10 to 15 values in the range of approximately 0 to 2 (for transcoding from about 192 kbps to 128 kbps) or 0 to 3 (for a larger difference in bit-rate), in steps of 0.25.

We define the direct mapping transcoder using the map $T\{.\}$, consisting of the maps $T_{\{s\}}\{.\}$ for $0 \le s \le 21$. The map $T_{\{s\}}\{.\}$ is defined by the parameter $(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})$, which can be varied by changing δ_g and $\delta_{f\{s\}}$. The resultant map from the input integer vector $\tilde{\mathbf{x}}_{1\{s\}}$ to the output integer vector $\tilde{\mathbf{x}}_{2\{s\}}$ is $\tilde{\mathbf{x}}_{2\{s\}} = T_{\{s\}}\left\{\tilde{\mathbf{x}}_{1\{s\}}, \mathbb{Z}_{1\{s\}}, \mathbb{Z}_{2\{s\}})\right\}$. Thus, the objective now is to design the map $T_{\{s\}}\{.\}$ by selecting δ_g and $\delta_{f\{s\}}$.

10.4.3 *A basic map*

A simple method to achieve an immediate bit-rate reduction is to increase the value of the global gain, i.e. to set a positive value of δ_g. To further reduce the bit-rate, we can limit the cutoff frequency of the audio by removing subband components in the higher frequencies, i.e. by setting $x_{2[m]}$ to zero.

Let us define a basic map, by using the following rules:

(i). Set $\delta_g = (\mathbb{Z}_{g2} - \mathbb{Z}_{g1})$ to be a positive integer constant.

(ii). Set $\delta_{f\{s\}} = (\mathbb{Z}_{f2\{s\}} - \mathbb{Z}_{f1\{s\}}) = 0$, for all s.

(iii). Set $\tilde{x}_{2[m]} = 0$, for $m > m_{cutoff}$, where m_{cutoff} relates to the cutoff frequency.

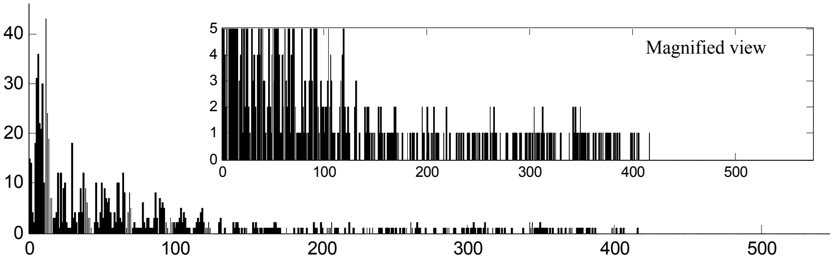
The value of m_{cutoff} offers a tradeoff between the cutoff frequency and the bit-rate for the transcoded material. If m_{cutoff} is very small, then the lack of high frequency components may become too apparent. If m_{cutoff} is very large, the increase in the bit-rate may be excessive while having little quality gain, as the human ear has a low sensitivity at high frequencies. For a target bit-rate of 128 kbps, a reasonable value of m_{cutoff} is experimentally determined to be about 342. We also chose 342 for convenience because this value corresponds with the edge of a scalefactor band. The value of m_{cutoff} can be changed to make minor adjustments to the final bit-rate. For example, m_{cutoff} can be reduced by 1 to reduce the number of bits required for each q-frame by approximately 1.

Thus:

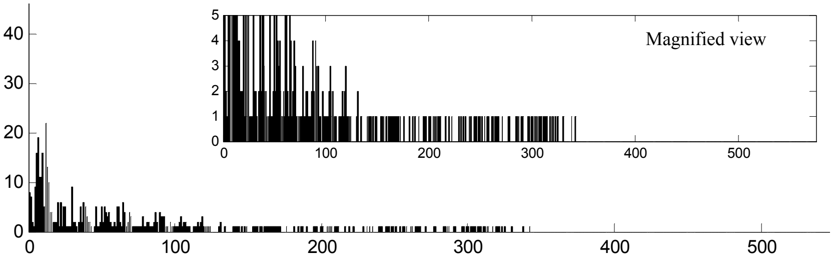
$$\tilde{x}_{2[m]} = \begin{cases} \Re\left\{\tilde{x}_{1[m]}\left[2^{-(\mathbb{Z}_{g2}-\mathbb{Z}_{g1})/4}\right]^{3/4}\right\} & ,\text{for } m \le m_{cutoff} \\ 0 & ,\text{for } m > m_{cutoff}. \end{cases} \tag{10.10}$$

The amount of bit-rate reduction can be controlled by the parameters δ_g and m_{cutoff}. When δ_g is increased (i.e. a coarser $q_{2\{s\}}$ is selected), the values of $\tilde{\mathbf{x}}_2$ tend to be decreased, and are thus expected to require fewer bits to store. The quantization error power is expected to be increased. When m_{cutoff} is decreased, the range of m for which $\tilde{x}_{2[m]}$ are stored is decreased, and hence fewer bits are expected to store $\tilde{\mathbf{x}}_2$. However, this leads to a loss of higher-frequency components.

Figure 10-8 shows the plot of $|\tilde{\mathbf{x}}_1|$ and $|\tilde{\mathbf{x}}_2|$ for $\delta_g = 5$ and $m_{cutoff} = 342$. The vector $\tilde{\mathbf{x}}_1$ was taken from an orchestral music item encoded using MP3 at 192 kbps.



(a) Plot of $|\tilde{\mathbf{x}}_1|$



(b) Plot of $|\tilde{\mathbf{x}}_2|$

Figure 10-8 Plot of $|\tilde{\mathbf{x}}_1|$ and $|\tilde{\mathbf{x}}_2|$ for $\delta_g = 5$ and $m_{cutoff} = 342$.

The basic map is straightforward, and effective in decreasing the bit-rate. However, we found it unsuitable for transcoding MP3. Experiments using different values of δ_g and m_{cutoff} (for typical bit-rates, e.g. from 192 kbps to 128 kbps), show that the quality of the transcoded audio is likely to be considered unacceptable by listeners.

Subband samples in the mid-to-high frequency regions (e.g. m>134; which corresponds to $s \geq 14$) are typically coarsely quantized. This can be observed in our example in Figure 10-8(a). For m>134, $|\tilde{x}_{1[m]}| \in \{0,1,2\}$. These subband samples do not provide much allowance for further quantization. On the other hand, subband samples in the lower frequencies are more pliable to further quantization, due to the larger values of $|\tilde{x}_{1[m]}|$.

To verify these claims, we transcoded a variety of music material (e.g. singing, orchestral, instrumental, etc.). The basic mapping method was used, and a frequency region was defined in which the subband samples were not altered, i.e. $x_{1[m]}$ is mapped to $x_{2[m]}$ for all m, except for m falling within the frequency region where $x_{1[m]}$ is not to be altered ($x_{2[m]} = x_{1[m]}$). We tested for different frequency regions and for different values of δ_g. In each case, the transcoded material was compared to the control, in which no frequency region was defined, i.e. $x_{1[m]}$ is mapped to $x_{2[m]}$ for all m. Generally, the most significant improvement in audio quality was attained when the frequency region was defined to correspond to the scalefactor bands $15 \leq s \leq 18$, or subbands $134 \leq m \leq 287$.

10.4.4 A fixed map

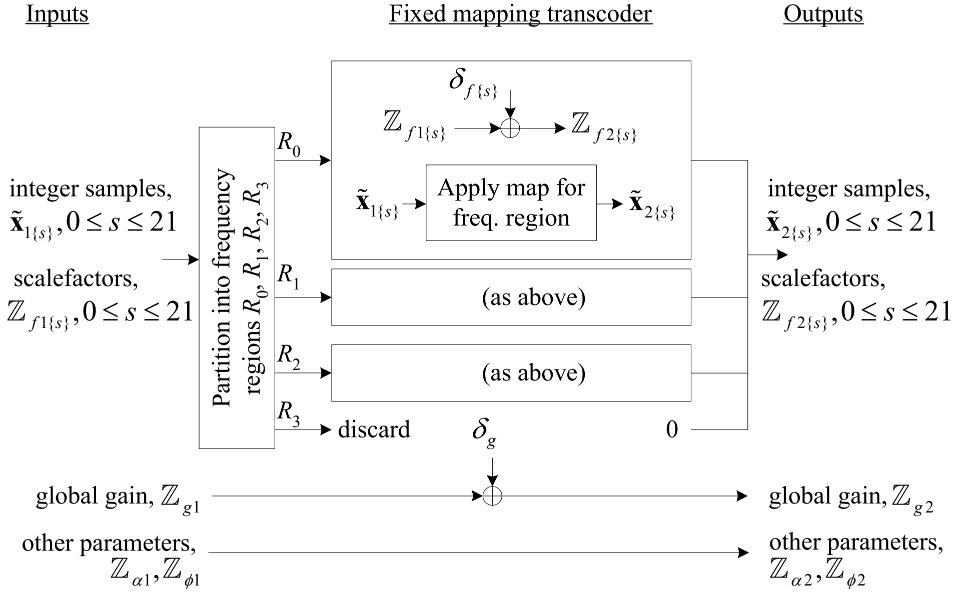

Inputs
Fixed mapping transcoder
Outputs
integer samples, $\tilde{\mathbf{x}}_{1\{s\}}, 0 \le s \le 21$
scalefactors, $\mathbb{Z}_{f1\{s\}}, 0 \le s \le 21$
Partition into frequency regions R_0, R_1, R_2, R_3
R_0
R_1
R_2
R_3
$\delta_{f\{s\}}$
$\mathbb{Z}_{f1\{s\}}$
$\mathbb{Z}_{f2\{s\}}$
$\tilde{\mathbf{x}}_{1\{s\}}$
Apply map for freq. region
$\tilde{\mathbf{x}}_{2\{s\}}$
(as above)
(as above)
discard
δ_g
0
integer samples, $\tilde{\mathbf{x}}_{2\{s\}}, 0 \le s \le 21$
scalefactors, $\mathbb{Z}_{f2\{s\}}, 0 \le s \le 21$
global gain, $\mathbb{Z}_{g1}$
global gain, $\mathbb{Z}_{g2}$
other parameters, $\mathbb{Z}_{\alpha 1}, \mathbb{Z}_{\phi 1}$
other parameters, $\mathbb{Z}_{\alpha 2}, \mathbb{Z}_{\phi 2}$

Figure 10-9 Block diagram for a fixed mapping transcoder.

Note: This method is 'fixed' in the sense that the map $T\{.\}$ is held constant for all MP3 frames. The fixed mapping method is basically a modification of the basic mapping method of Section 10.4.3. The map for each frequency region can be applied by using the mathematical function in (10.6).

The block diagram for the proposed fixed mapping transcoder is shown in Figure 10-9. For each q-frame, the vector of input samples $\tilde{\mathbf{x}}_1$ is divided into a number of frequency regions (we proposed 4 in this example), with the boundaries of each frequency region coinciding with the boundaries of certain scalefactor bands.

We define 4 frequency regions labeled by R_0, R_1, R_2 and R_3, and the input samples are grouped into these regions. In each frequency region, a different value of $\delta_{f\{s\}} = \mathbb{Z}_{f2\{s\}} - \mathbb{Z}_{f1\{s\}}$ is applied. In frequency regions where large values of $\delta_{f\{s\}}$ are applied, a relatively coarser quantization is applied. These frequency regions have 'low emphasis'. In frequency regions where small values of $\delta_{f\{s\}}$ are applied, a relatively finer quantization is applied. These frequency regions have 'high emphasis'.

Figure 10-10 shows the proposed division of the 22 scalefactor bands (576 subband samples) into frequency regions R_0, R_1, R_2 and R_3. The proposed division was chosen based on experimental observations (described earlier in Section 10.4.3). Since the frequency region R_1 corresponding to $15 \le s \le 18$ was deemed to provide the most significant impact on audio quality relative to the other frequency regions, we designate R_1 as 'high emphasis', and the rest as 'low emphasis'. Since the change in global gain δ_g is applied to the entire q-frame, the emphasis is varied by applying different values of $\delta_{f\{s\}}$ to each frequency region.

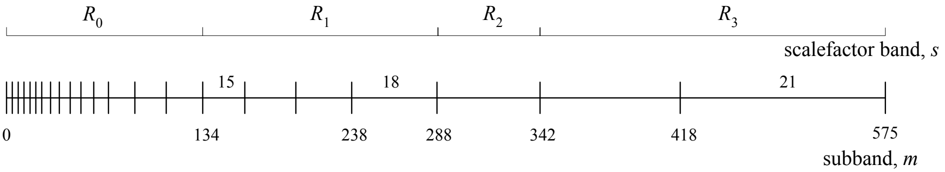


Figure 10-10 Division of scalefactor bands into frequency regions for fixed mapping transcoder.

An effective map that worked well for a variety of audio material encoded at 192 kbps was found to be:

(i). Set δ_g to be a positive integer constant.

(ii). Set $\delta_{f\{s\}} = \begin{cases} 0, & \text{for } R_0 : 0 \le s \le 14 \\ 1, & \text{for } R_1 : 15 \le s \le 18 \\ 0, & \text{for } R_2 : s \ge 19. \end{cases}$

(iii). Set $\tilde{x}_{2[m]} = 0$, for $m > m_{cutoff}$, where $m_{cutoff}=342$.

The average bit-rate achieved using this map was found to be about 55-75% of the initial bit-rate. For an initial bit-rate of 192 kbps, the bit-rate of the transcoded material works out to be around 128 kbps. Various options are possible for the definition of the map $T\{.\}$. For a different source and target bit-rate, a different $T\{.\}$ would have to be defined. The given parameters can also be adjusted to tweak the final bit-rate along with audio quality.

Since δ_g and $\delta_{f\{s\}}$ are 'fixed' for the duration of the transcoding, the tandem gain (refer to Section 9.3.2) is constant for the duration of the transcoded material. Thus, the fixed mapping transcoder has the advantage over conventional sample-synchronized transcoders in that the tandem gain of transcoded material is controlled (does not fluctuate). The effect of fluctuating tandem gain was discussed earlier (Section 9.3.2).

Results of formal listening tests, using audio material that was transcoded by the fixed mapping method are documented in Section 10.5 (Figure 10-13) and Appendix E. The fixed mapping transcoder (*LT-5*) performs reasonably well, when compared to the conventional frame-synchronized transcoder (*LT-3*). The mean opinion score for *LT-5* is slightly higher than for *LT-3*, and the confidence intervals overlap significantly. The favourable results of the listening test demonstrates the feasibility of a non-psychoacoustic-centric transcoder.

10.4.5 An adaptive map

A fixed map may not be optimized for the different conditions that are faced when transcoding. Due to the highly time-varying nature of audio, and the different methods that are used to compress the audio in the initial encoding stage, some form of adaptation is desirable. For example, i) the source bit-rate might not be constant, such as for variable bit-rate audio; and ii) certain scalefactor bands might be more or less compressible than others.

In this sub-section, we consider a transcoder using a non-psychoacoustic-centric adaptive map. Compared to the conventional transcoder, the adaptive mapping transcoder is very efficient.

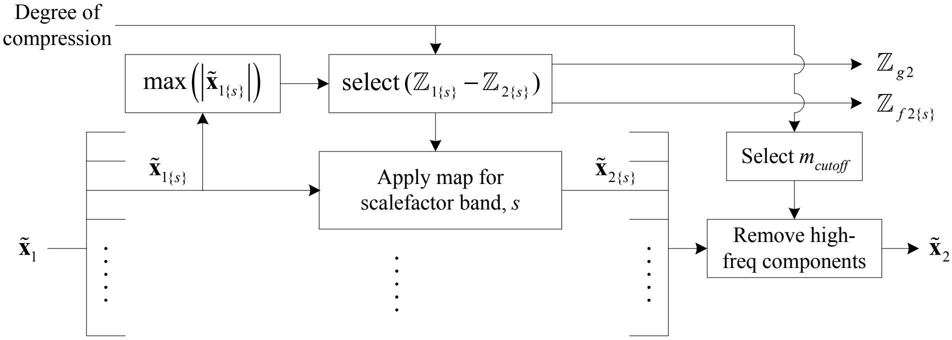

Degree of compression
$\max\left(\left|\tilde{\mathbf{x}}_{1\{s\}}\right|\right)$
select $(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})$
$\mathbb{Z}_{g2}$
$\mathbb{Z}_{f2\{s\}}$
Select m_{cutoff}
$\tilde{\mathbf{x}}_{1\{s\}}$
Apply map for scalefactor band, s
$\tilde{\mathbf{x}}_{2\{s\}}$
$\tilde{\mathbf{x}}_1$
Remove high-freq components
$\tilde{\mathbf{x}}_2$

Figure 10-11 Block diagram for adaptive mapping transcoder.

Figure 10-11 shows the block diagram of the proposed adaptive mapping transcoder. For each q-frame, the compression strength and the value of the largest element in the vector $\left|\tilde{\mathbf{x}}_{1\{s\}}\right|$ are considered in the selection of $(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})$. The proposed transcoder can be thought of as a trimmer; it trims the values of $\tilde{\mathbf{x}}_{1\{s\}}$ (using quantization) by a variable amount which is deemed as surplus, where the surplus is estimated by $\max\left(|\tilde{\mathbf{x}}_{1\{s\}}|\right)$. The 'degree of compression' is used to control the trade-off between the reduction in bit-rate and the audio quality of the transcoded material. A higher degree of compression increases the bit-rate reduction at the cost of decreased audio quality.

The algorithm is described as follows:

a. Select $(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})$:

We adopt a case-select approach for the selection of $(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})$, based on the degree of compression and the value of $\max\left(|\tilde{\mathbf{x}}_{1\{s\}}|\right)$. An internal table of $(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})$ for a 'normal' degree of compression is pre-generated (see Table 10-1). A 'normal' degree of compression is deemed to result in transcoding to a bit-rate of approximately 128 kbps for MP3. If the value of $\max\left(|\tilde{\mathbf{x}}_{1\{s\}}|\right)$ falls within the indicated range, then the corresponding value of $(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})$ is selected, e.g. if $\max\left(|\tilde{\mathbf{x}}_{1\{s=8\}}|\right)\in[15,20]$, then $(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})$ is selected to be 2.

Select	Value of $\max\left(\lvert\tilde{\mathbf{x}}_{1\{s\}}\rvert\right)$, where:				
$(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})$	$0 \le s \le 5$	$6 \le s \le 11$	$12 \le s \le 14$	$15 \le s \le 17$	$18 \le s \le 20$
2.5	>63	>20			
2	[32, 63]	[15,20]	>12	>10	>4
1.5	[16,31]	[10,14]	[8,12]	[6,10]	[4]
1	[8,15]	[5,9]	[4,7]	[2,5]	[2,3]
0	[0,7]	[0,4]	[0,3]	[0,1]	[0,1]

Table 10-1 Allocation table for $(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})$,
for a 'normal' degree of compression.

The tabulated values in Table 10-1 were calculated empirically. The spectrum was divided into 5 frequency regions, and separate rules were applied to each frequency region. The proposed table reflects the tendency that values of $\tilde{\mathbf{x}}_{1\{s\}}$ tend to have larger magnitude at the lower frequencies and smaller magnitude at the higher frequencies. The determination of the table is not rigid, and we merely proposed one that was tested to work well.

b. Obtain $\mathbb{Z}_{g2}$ and $\mathbb{Z}_{f2\{s\}}$:

Recall from (10.7)-(10.9) that $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$ is defined to be equal to $0.25\delta_g - \mathbb{Z}_{\alpha 1}\delta_{f\{s\}}$, where the global gain δ_g is a value that is common to all scalefactor bands. The decomposition of $\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}$ to δ_g and $\delta_{f\{s\}}$ is not unique, since we can obtain:

$$\delta_{f\{s\}} = \left[0.25\delta_g - (\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})\right] / \mathbb{Z}_{\alpha 1}. \tag{10.11}$$

Note that δ_g and $\delta_{f\{s\}}$ are integers, and the parameter $\mathbb{Z}_{\alpha 1}$ is retrieved from the initially-encoded bitstream. For transcoding from 192 kbps to 128 kbps, δ_g can normally be set to a value of 6. We can then obtain $\mathbb{Z}_{g2}$ and $\mathbb{Z}_{f2\{s\}}$ from $\mathbb{Z}_{g1} + \delta_g$ and $\mathbb{Z}_{f1\{s\}} + \delta_{f\{s\}}$ respectively.

c. Map $\tilde{\mathbf{x}}_{1\{s\}}$ to $\tilde{\mathbf{x}}_{2\{s\}}$ using (10.5).

d. Remove high-frequency components:

Set $x_{2[m]} = 0$ for $m > m_{cutoff}$. Decreasing the value of m_{cutoff} reduces bit-rate and audio quality due to the loss of high-frequency content. We use a default of $m_{cutoff} = 342$ for a 'normal' compression strength, which is reasonable at a target bit-rate of 128 kbps. Note that we maintain a constant value of m_{cutoff} throughout the transcoding of a piece of audio, because it is known that fluctuating values of m_{cutoff} have an 'on-off-on-off' effect which is highly audible during play back (explained in Section 9.3.2).

Results of formal listening tests, using audio material that was transcoded by the adaptive mapping method are documented in Section 10.5 (Figure 10-14) and Appendix E. It can be seen that the mean opinion score for the adaptive mapping transcoder (*LT-6*) is slightly higher than for the fixed mapping transcoder (*LT-5*). Furthermore, its upper confidence interval range overlaps with the lower confidence interval range of the non-frame-synchronized transcoder (*LT-4*). Contrast this with the fixed mapping transcoder (*LT-5*), which does not overlap with *LT-4*. The adaptive mapping transcoder has a slightly better audio quality, at the cost of increased computations. In our implementation, the adaptive mapping transcoder typically shows a 10% - 20% increase in the amount of processing time, compared to the fixed mapping transcoder. Note, however, that this result is only indicative, as the efficiency of the transcoder is highly implementation-dependent, and our implementation is not fully optimized.

10.4.6 Using a lookup table

The proposed transcoders are non-psychoacoustic-centric, and highly streamlined in terms of operations required. Computations normally required by the filter bank blocks, psychoacoustic modeling, rate-distortion loop and FFT are eliminated (compare Figure 10-4 and Figure 10-7). The transcoding is reduced to only 3 steps: Huffman decoding, mapping and Huffman encoding.

We recall that the quantized subband samples $\tilde{x}_{1[m]}$ and $\tilde{x}_{2[m]}$ are integer-valued. The map from $\tilde{x}_{1[m]}$ to $\tilde{x}_{2[m]}$ is:

$$\tilde{x}_{2[m]} = \Re\left\{ \tilde{x}_{1[m]} \left(2^{-(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})} \right)^{3/4} \right\}. \tag{10.12}$$

It can be easily seen that some computationally-intensive mathematical operators are involved, notably the multiplication, division, and exponential operators. We can further streamline the transcoding by using an integer-to-integer map, via a lookup table search, that avoids these computationally-intensive mathematical operations. For software implementations, costly floating-point operations are eliminated. For hardware implementations, we can see benefits in complexity reduction, lower power consumption and smaller chip area.

The integer-to-integer map, using the proposed lookup table method, is illustrated in Figure 10-12. For each value of $(\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}})$, we pre-generate a lookup table that is represented by a vector $\boldsymbol{\chi}_{\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}}$. The map from $\tilde{x}_{1[m]}$ to $\tilde{x}_{2[m]}$ can then be performed by reading the $\tilde{x}_{1[m]}$-th element of the vector $\boldsymbol{\chi}_{\mathbb{Z}_{1\{s\}} - \mathbb{Z}_{2\{s\}}}$.

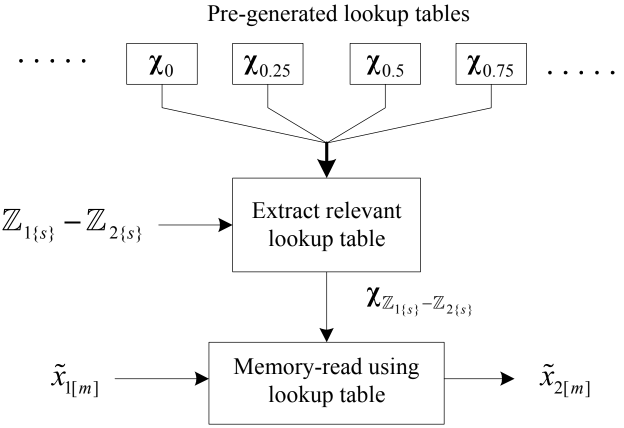

Pre-generated lookup tables
. . . . .
χ0
χ0.25
χ0.5
χ0.75
. . . . .
ℤ1{s} − ℤ2{s}
Extract relevant lookup table
χℤ1{s}−ℤ2{s}
x̃1[m]
Memory-read using lookup table
x̃2[m]

Figure 10-12 Lookup table method for ultra-fast integer-to-integer mapping.

Using the lookup table, the mapping becomes as fast as a single memory-read operation. We realize that an effective lookup table should also be reasonably sized in order to reduce memory storage requirements. At a first glance though, the mapping range is very large, if we assume $(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})$ is arbitrary and the range of $|\tilde{x}_{1[m]}|$ and $|\tilde{x}_{2[m]}|$ are from 0 to 8207 (according to the MP3 specifications, [71]).

To overcome this problem, we first limit the range of admissible values of $(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})$. For example, we discussed the bit-rate transcoding scenario from 192 kbps to 128 kbps (Section 10.4.2), and concluded that values of $(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})$ are usually limited in range from 0 to 2, in steps of 0.25. Thus, only 9 lookup tables are needed.

Second, we assume that the majority of the values of $|\tilde{x}_{1[m]}|$ and $|\tilde{x}_{2[m]}|$ are limited in range from 0 to 255. This is consistent with repeated observations of typical MP3 audio. The memory size of each lookup table can then be reduced considerably. Each element of the lookup table can be stored using a single byte ($0 \le |\tilde{x}_{2[m]}| \le 255$), as compared to 2 bytes ($0 \le |\tilde{x}_{2[m]}| \le 8207$) previously. Furthermore, the memory-address index can also be similarly represented using a single byte ($0 \le |\tilde{x}_{1[m]}| \le 255$), as compared to 2 bytes previously ($0 \le |\tilde{x}_{1[m]}| \le 8207$). The total memory required is 2304 bytes (9 lookup tables, each with 256 elements). Compare this with the lookup table using 8208 elements; the total memory required is 147,744 bytes (9 lookup tables, each with 8208 elements, each element requiring 2 bytes).

Note that for the small minority of values exceeding 255, it is possible to handle them separately without incurring significant overhead, by rounding the product of $|\tilde{x}_{1[m]}|$ and the pre-generated values of $2^{-3/4(\mathbb{Z}_{1\{s\}}-\mathbb{Z}_{2\{s\}})}$ (refer to (10.6)).

10.5 Results

10.5.1 Results on audio quality

The transcoders were implemented using the C programming language, on the Windows Intel Pentium-3 1 GHz platform. For our tests, we used the MP3 encoding and decoding packages in [108]-[110]. We conducted 2 listening tests. Details of the tests, and their results are included in Appendix E.

In each test, the audio pieces are shuffled randomly, and the listener is asked to grade based on his opinion (scoring on a scale of 0 to 100) of the audio quality of each audio piece. The mean opinion scores (MOS) are obtained by averaging the individual opinion scores. The results are shown in Figure 10-13 and Figure 10-14. In the figures, the MOS is indicated by a cross 'x' and its 95% confidence interval is indicated by the delimited vertical line. The different audio items are abbreviated as follows:

LT-1	Hidden reference of the original PCM signal.
LT-2	Conventional encoding from original PCM signal to MP3 at 128 kbps.
LT-3	Conventional transcoding (frame-synchronized), from MP3 at 192 kbps to MP3 at 128 kbps.
LT-4	Conventional transcoding (non-frame-synchronized), from MP3 at 192 kbps to MP3 at 128 kbps.
LT-5	Fixed map transcoding from MP3 at 192 kbps to MP3 at 128 kbps.
LT-6	Adaptive map transcoding from MP3 at 192 kbps to MP3 at 128 kbps.

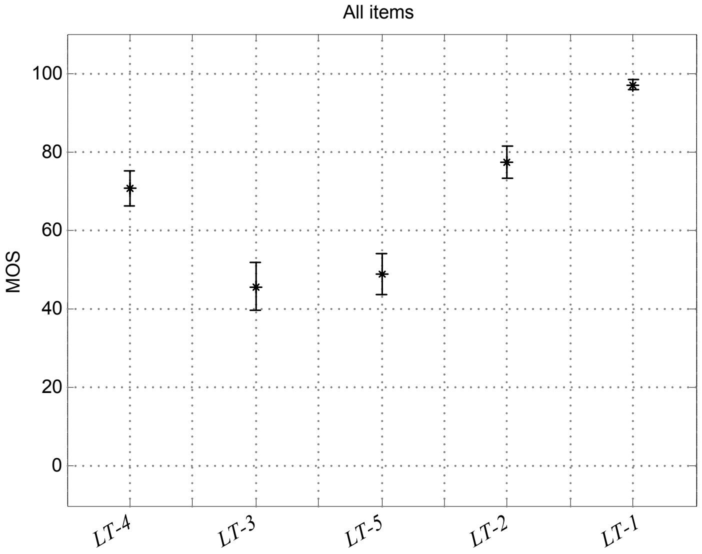


Figure 10-13 Results of Listening Test 1.

The first listening test (shown in Figure 10-13) compares the fixed mapping transcoder (*LT-5*) with the conventional transcoder (*LT-3* and *LT-4*). Results show that it is statistically comparable to the conventional frame-synchronized transcoder (*LT-3*).

For the second listening test (shown in Figure 10-14), we also included the adaptive mapping transcoder (*LT-6*). Results show a slight improvement over the fixed mapping transcoder (*LT-5*).

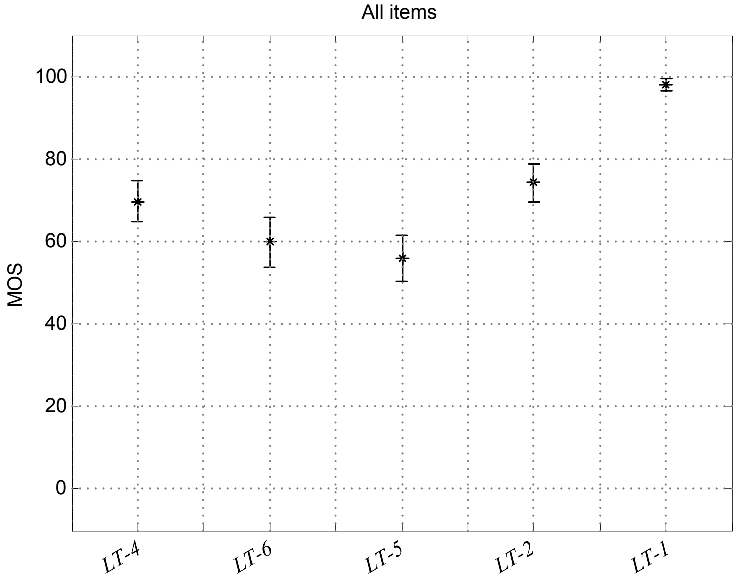


Figure 10-14 Results of Listening Test 2.

Both the proposed transcoders (labeled by *LT-5* and *LT-6*) show comparable audio quality to the conventional method of frame-synchronized transcoding.

10.5.2 Results on execution speed

For these tests, we used the LAME [108] and the Fastmp3 [110] software packages as a basis for comparison. These software packages are used to perform conventional transcoding, by first decoding a 192 kbps MP3 bitstream to an intermediate PCM signal and then re-encoding to a 128 kbps MP3 bitstream.

We used an assortment of songs as material for transcoding. The total playback time of the songs was about 140 mins. For a bit-rate of 128 kbps, these songs take up approximately 128 MB of memory (a typical amount of memory available in a hardware portable player). The total time taken to transcode these songs is tabulated in Table 10-2. The RT (real-time) ratio is the ratio of the total playback time to the total time taken to perform transcoding.

	Time taken	RT ratio
LAME package	27 min 10 secs	5.2
Fastmp3 package	7 mins 45 secs	18.1
Fixed map method	1 min 40 secs	84
Adaptive map method	2 min 2 secs	68.9

Table 10-2 Results of test on transcoding speed.

In this test, the fixed mapping transcoder was operating at a RT ratio of 84 (i.e. 84 times faster than the total playback speed), about 16 times faster than the popular LAME software package, and about 5 times faster than the Fastmp3 software package (which is highly optimized for fast processing). With additional refinements and coding improvements (such as by using hardware language programming), we could expect an even higher RT ratio. The adaptive mapping transcoder required about 20% more processing time than the fixed mapping transcoder.

10.6 Discussions

The drawback to enforcing a frame-synchronous condition was noted in Chapter 9 to have a negative effect on the MP3 audio quality. Nevertheless, we found that very significant opportunities for complexity reduction are presented when transcoding frame-synchronously. To this end, we developed the ultra-fast frame-synchronous MP3 bit-rate transcoder. From test results, the audio quality when using this frame-synchronous method generally falls within acceptable limits, while being able to achieve a large efficiency gain. Thus, the two approaches (frame-synchronous and non-frame-synchronous) present a trade-off between quality and efficiency, and may be useful in different applications having different requirements.

Our work focused on the MP3 to MP3 bit-rate transcoding scenario. The complexity reduction methodology that was discussed can also be extended to other same-format bit-rate transcoding scenarios, e.g. AAC to AAC, or WMA to WMA. In general, this methodology can also be applied to cross-format transcoding (i.e. between different compression methods), if the synthesis block of the first compression method has the same number of subbands and is PR or NPR, when matched with the analysis block of the second compression method. Since many compression methods use analysis-synthesis blocks which do not form PR or NPR pairs with each other, e.g. AAC has 1024 subbands, while MP3 has 576 subbands, this methodology does not apply in most cases of cross-format transcoding.

10.7 Summary and future work

10.7.1 Summary

In this chapter, we developed methods to improve bit-rate transcoding for the MP3 method. By using frame-synchronization, it is possible to bypass the filter bank blocks, rate-distortion loop and psychoacoustic modeling. Furthermore, floating point operations (which are normally required in the filter bank, psychoacoustic model and quantization blocks) can be avoided by using a lookup table. Using our method, the transcoding process is reduced to 3 simple stages: Huffman decoding, an integer-to-integer mapping and Huffman encoding. Both a fixed map (quantization parameters are constant for the duration of each transcoding) and an adaptive map (quantization parameters are selected using a ruleset) were proposed.

The proposed transcoder has a very low complexity and is suitable for power/memory constrained applications. Using a software program written using the C programming language, we observed execution speeds of approximately 5-16 times faster than the best known conventional transcoders. Formal listening tests showed that material transcoded using our implementation did not exhibit any significant deterioration in terms of audio quality, when compared to conventional frame-synchronized transcoders.

10.7.2 Future work

The prototype program that was written in C is not fully optimized, both in terms of audio quality and execution speed. The adaptive map rules that were proposed could be further improved, such as by taking into account the mean, median, etc. of the vector $\tilde{x}_{1\{s\}}$. However, it is to be noted that when more factors are taken into account, more computation time is also required. The execution speed can also be improved, for instance by using hardware programming language and other software tweaks.

Our transcoding method for MP3 can be extended to other methods such as AAC. However, implementations and listening tests would need to be conducted to establish an appropriate set of transcoding parameters for such methods.

In the proposed transcoder, the Huffman decoding and encoding processes require a significant amount of computation (>50%), when compared to the overall amount of computation required for transcoding. Future work might consider the possibility of a mapping that operates in the bitstream domain, i.e. directly operating on the Huffman-encoded data. This could further reduce the total computation required, by eliminating the need for Huffman decoding and encoding.

It would also be interesting to investigate the possibility of complexity reduction in the case of non-frame-synchronized transcoding. The advantages of non-frame-synchronized transcoding for MP3 were discussed in Section 9.3. The main problem faced, however, is that our method of complexity reduction is dependent on the condition of frame-synchronization (Sections 10.2-10.4). Future work could include investigations on the possibility of reducing the filter bank blocks, psychoacoustic model and quantization processes for non-frame-synchronized transcoding. Complexity reduction for the general transcoding scenario, between different compression methods, also provides many opportunities for future work.

References for Part II

[60] Compact Disc Digital Audio System, (IEC/ANSI) CEI-IEC-908, 1987.

[61] P. Craven and M. Gerzon, "Lossless coding for audio discs," *Journal of the AES*, pp. 706–720, Sept. 1996.

[62] "Free Lossless Audio Codec (FLAC)", [online] http://flac.sourceforge.net.

[63] T. Liebchen, "MPEG-4 Lossless Coding for High-Definition Audio", Proc. AES 115th Convention, October 2003.

[64] ISO/IEC International Standard 14496-3:2001/FPDAmd 4, "Audio Lossless Coding (ALS)".

[65] G. Theile, G. Stoll, and M. Link, "Low-bit rate coding of high quality audio signals," *Proc. AES 82nd Convention*, preprint 2432, Mar. 1987.

[66] J. D. Johnston, "Transform coding of audio signals using perceptual noise criteria," *IEEE Journal on Selected Areas in Communications*, vol. 6, pp. 314-323, Feb 1988.

[67] K. Brandenburg and J. D. Johnston, "Second generation perceptual audio coding: The hybrid coder," *Proc. 88th AES Convention*, preprint 2937, Mar. 1990.

[68] T. Painter and A. Spanias, "Perceptual coding of digital audio," *Proceedings of the IEEE*, vol. 88, no. 4, pp. 451-513, April 2000.

[69] "Microsoft Windows Media – 9 series codecs – audio (WMA)", [online]http://www.microsoft.com/windows/windowsmedia/9series/codecs/audio.aspx.

[70] "Ogg Vorbis (OGG): Open, free audio", [online] http://www.vorbis.com.

[71] ISO/IEC International Standard IS 11172-3, "Information technology – Coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbit/s", 1992.

[72] K. Brandenburg, G. Stoll, Y.F. Dehery, J.D. Johnston, L.v.d. Kerkhof, E.F. Schroeder, " The ISO/MPEG audio codec: A generic standard for coding of high quality digital audio," *Proc. 92nd AES Convention*, preprint 3336, 1992.

[73] ISO/IEC International Standard IS 13818-3, "Information technology – Generic coding of moving pictures and associated audio information", 1998.

[74] ISO/IEC TR 13818-5:1997/Amd 1:1999, "Advanced Audio Coding (AAC)", 1999.

[75] E. Zwicker, H. Fastl, *Psychoacoustics: Facts and models*, 2nd edition, Springer, 1999.

[76] N. Jayant, J. D. Johnston, and R. Safranek, "Signal compression based on models of human perception," *Proc. of the IEEE*, vol. 81, pp. 1385–1422, Oct. 1993.

[77] E. Zwicker and U. Zwicker, "Audio engineering and psychoacoustics— Matching signals to the final receiver, the human auditory system," *Journal of the AES*, pp. 115–126, March 1991.

[78] J. Johnston, "Estimation of perceptual entropy using noise masking criteria," *Proceedings of ICASSP 1988*, pp. 2524–2527, May 1998.

[79] B. C. J. Moore, "Masking in the human auditory system," *Collected Papers on Digital Audio Bit-Rate Reduction*, N. Gilchrist and C. Grewin, Eds., pp. 9–19, 1996.

[80] H. Fletcher, "Auditory patterns", *Rev. Mod. Physics*, pp. 47-65, Jan 1940.

[81] S. Ritscher and U. Felderhoff, "Cascading of different audio codecs", *AES 100th Convention*, preprint 4174, May 1996.

[82] C. Grewin, "Evaluation of low bit rate audio codecs for a national network", *98th AES Convention*, preprint 3944, Feb 1995.

[83] M. Keyhl, J. Herre, C. Schmidmer, "NMR Measurements on Multiple Generations Audio Coding," *AES 96th Convention*, Paper 3803, Jan. 1994.

[84] W. ten Kate, "Maintaining audio quality in cascaded psychoacoustic coding", *AES 101st Convention*, Paper 4387, Nov 1996.

[85] W. ten Kate, "Repeated decoding and encoding in subband encoders/decoders", United States Patent, No: US 6,430,226 B1, Aug 2002.

[86] Nika Aldrich, *Digital audio explained: For the audio engineer*, Booksurge Publishing, 2005.

[87] D.D. Koning and W. Verhelst, "On psychoacoustic noise shaping for audio requantization", *International Conference on Acoustics, Speech and Signal Processing*, vol. 5, pp. 453-456, 2003.

[88] P. Carbone, "Dithered requantization", *1st International On-line Workshop on Dithering in Measurement: Theory and Applications*, Paper number 2, Feb 1998.

[89] S.C. Heol, et al, "A new requantization method for MPEG-1 to MPEG-4 transcoder", *IEEE Int. Conf. on Multimedia and Expo*, pp. 12-15, 2001.

[90] R. Storey, "ATLANTIC: Advanced television at low bitrates networked transmission over integrated communication systems," *ACTS Common European Newsletter*, Feb. 1997.

[91] N. Gilchrist, "ATLANTIC audio: Preserving technical quality during low bit rate coding and decoding," *Proc. 104th AES Convention*, preprint 4694, May 1998.

[92] J. Fletcher, "ISO/MPEG Layer 2 - Optimum re-encoding of decoded audio using a MOLE signal," *Proc. 104th AES Convention*, Paper 4706, April 1998.

[93] F. Kurth, "An audio codec for multiple generations compression without loss of perceptual quality", *AES 17th International Conference on High Quality Audio Coding*, Paper 028, 1999.

[94] F. Kurth, V. Hassenrik, "A Dynamic Embedding Codec for Multiple Generations Compression", *AES 109th Convention*, Paper 5257, August 2000.

[95] Y. Wang, J. Ojanpera, M. Vilermo, M. Vaananen, "Schemes for re-compressing MP3 audio bitstreams," *AES 111th Convention*, paper 5435, Sep 2001.

[96] J. Bregar, J. Kappes, "The Future of Media Interactivity and Choice", *AES 18th International Conference*, Preprint No. 1856, Feb 2001.

[97] R. Elen, R. Clearmountain, G. Laney, "Optimizing Audio Quality for Internet Streaming and Distribution", *AES 18th International Conference*, Preprint No. 1865, Feb 2001.

[98] G. Bartlett, "Information Appliances: New Toys and New Business Models", *AES 18th International Conference*, Preprint No. 1860, Feb 2001.

[99] M. Hans, R. Schafer, "An MPEG audio layered transcoder", *Proc. 105th AES Convention*, preprint 4851, Sep 1998.

[100] M. vd. Veen, A. Lemma, T. Kalker, “Watermarking and fingerprinting for electronic music delivery”, *SPIE Conference on Security and Watermarking of Multimedia Contents*, vol. 5306, pg. 200-211, 2004.

[101] "Efficient transcoding between different audio formats", PDSL-214/03, Invention disclosure version 1.4.0, 14 Nov 2002.

[102] Y. Nakajima, H. Yanagihara, A. Yoneyama and M. Sugano, "MPEG audio bit rate scaling on coded data domain", *International Conference on Acoustics, Speech and Signal Processing*, vol. 6, pp. 3669-3672, May 1998.

[103] Y. Wang, J. Ojanpera, M. Vilermo, M. Vaananen, "Schemes for recompressing MP3 bitstreams", *Proc. of 111th AES Convention*, preprint 5435, Sep 2001.

[104] J.W. Lee, W. Oomen, F. de Bonts, "Method and device for transcoding", Philips internal reference number PHNL040946EPP, Application date: 31 Aug 2004.

[105] K. Konstantinides, "Fast subband filtering in MPEG audio coding", *IEEE Signal Processing Letters*, vol. 1, pp. 26-28, Feb 1994.

[106] H. Oh, J.S. Kim, C.J. Song, Y.C. Park, D.H. Youn, "Low power MPEG/audio encoders using simplified psychoacoustic model and fast bit allocation", *IEEE. Trans. on Consumer Electronics*, vol. 47, no. 3, Aug 2001.

[107] T.H Tsai, S.W Huang and Y.W Wang , "Architecture design of MDCT-based psychoacoustic model co-processor in MPEG advanced audio coding", *IEEE International Symposium on Circuits and Systems*, vol. 2, pp. 761-764, 2004.

[108] "LAME 3.95: MP3 encoder", [online] http://lame.sourceforge.net

[109] "MADPLAY: MP3 audio decoder", [online] http://www.underbit.com/products/mad

[110] "Fast MP3: LSVD Live Vibes Speed Dump encoder", Philips internal software, refer to PDSL-E Sound Coding Group.

[111] T. Ryden, "Using listening tests to assess audio codecs", *Collected Papers on Digital Audio Bitrate Reduction*, pp. 115-125, 1996.

[112] B. Paillard, P. Mabilleau, S. Morissette, "PERCEVAL: Perceptual evaluation of the quality of audio signals", *Journal of the Audio Engineering Society*, vol. 40, no. 1, p. 21-31, 1992.

[113] G. Soulodre et al., "Subjective evaluation of state-of-the-art two channel audio codecs", *Journal of the Audio Engineering Society*, vol. 46, no. 3, pp. 164-177, Mar 1998.

[114] G. Stoll, F. Kozamernik, "EBU listening tests on Internet audio codecs", *EBU Technical Review*, pp. 1-24, June 2000.

[115] "Method for the subjective assessment of intermediate quality level of coding systems", *ITU-R Recommendation BS.1534*, June 2001.

Chapter 11
Conclusion

In this thesis, we approached the subject of efficient audio processing (in the context of compressed audio) in 2 parts: Part I: Filter banks and Part II: Audio transcoding.

In Part I, Chapter 3, we considered the design and complexity aspects of the Frequency Response Masking (FRM) class of filter banks for single-rate processing. For example, we found that when designing the shaping subfilter $H_{0,0}(z)$, its transition width should not be greater than 0.3π for desirable overall complexity. A 32-channel single-rate Fast Filter Bank (FFB) with very small transition widths was designed and had only 1/4 of the complexity compared to the 32-channel polyphase filter bank.

In Chapter 4, we proposed the node-modulation method which reduces the complexity of the FFB, by constraining the subfilters to having real-valued coefficients. For the processing of real-valued input signals, we further proposed a pruning method to reduce the FFB complexity. Using both the pruning and node-modulation methods, a 16-channel FFB was designed, and it had about half of the complexity compared to the original FFB. A matrix formulation of the FFB for software implementation was also proposed. By expressing the data operations using vectors and matrices, an efficient filtering scheme can be implemented using easily available mathematical packages, such as the Basic Linear Algebra Subprograms (BLAS), which are highly optimized for specific computer architectures. The required computation time was decreased by up to a factor of 3 when compared to the normal method of evaluating the convolution result for each subfilter.

In Chapter 5, we considered the application of the FFB to multi-rate signal processing. A method of decimation was proposed by the removal of redundant computations required by the subfilters at the output stage. It was shown that for some designs with very small transition widths, our improved FFB designs were even more efficient than the polyphase filter bank. We classified the FRM filter banks into 4 types: Type *F-I*, Type *F-II*, Type *F-III* and Type *F-IV*. The structure of each type was explained and the different types offer trade-offs between the filter lengths required for the individual subfilters and the overall complexity of the filter bank.

In Chapter 6, we analyzed the effects of distortion and aliasing caused by a pair of critically-decimated FRM analysis-synthesis filter banks. The distortion and aliasing error functions for the M-channel FFB were derived, and we found that the masking subfilters have a very small contribution to these functions. Therefore, the design of near-perfectly-reconstructing (NPR) FRM filter banks can be suitably achieved by using perfectly-reconstructing (PR), or NPR 2-channel filter banks as the shaping subfilters. Furthermore, since the masking subfilters have a small impact on the distortion and aliasing error functions as well as the overall filter bank transfer functions, we reduced the masking subfilter coefficients to signed-powers-of-two (SPT) terms. We designed a 5-channel variable bandwidth filter bank using SPT terms. The result was a reduction in complexity by a factor of up to 10, when compared to an octave filter bank with similar transition widths.

In Part II, Chapter 8, we analyzed the tandem quantization error for the MPEG 1 Layer 2 method. Two methods for reducing the total quantization error power of the signal at the output of the transcoder were proposed. In the modified quantizer method, the quantization characteristics of either the first quantizer $q_{1[m]}$, or the second quantizer $q_{2[m]}$ were modified to reduce the tandem noise power at the output. The quantizer selection method takes an approach that does not modify the characteristics of the quantizers. Instead, when the tandem noise power at the output exceeds a threshold value, a different quantizer is selected instead of the originally intended $q_{2[m]}$, such that the resultant tandem noise power is decreased.

In Chapter 9, the impact of sample-synchronization on audio transcoding was investigated for the MPEG 1 Layer 2 and MP3 methods. For the MPEG 1 Layer 2 method, we proposed a measure termed the effective scaling factor $\zeta_{E[m]}$, which can be used to estimate the amount of quantization error incurred during transcoding. For a typical piece of audio, we found that the effective scaling factor can be estimated to lie between 1 and 1.23. For the MP3 method, we found that the subband samples in the mid-to-high frequency region are sensitive to transcoding. In this frequency region, the subband components are prone to a condition which we call fluctuating tandem gain, which can give rise to an irregular and audible 'loud-soft-loud-soft' sensation during audio playback.

In Chapter 10, we developed methods to improve bit-rate transcoding for the MP3 method. We reduced the transcoding process to 3 simple stages: Huffman decoding, an integer-to-integer mapping and Huffman encoding. Both a fixed map and an adaptive map were proposed. The result is a transcoder with very low complexity that is suitable for power/memory constrained applications. Using a software program written using the C programming language, we observed execution speeds of approximately 5-16 times faster than the best known conventional transcoders. Formal listening tests showed that material transcoded using our implementation did not exhibit any significant deterioration in terms of audio quality, when compared to conventional frame-synchronized transcoders.

Appendix A Length of a Filter

Known methods of estimating the minimum length of an equiripple lowpass FIR filter, based on its design requirements, exist ([1]-[2]). The estimates are provided in (A.1) and (A.2).

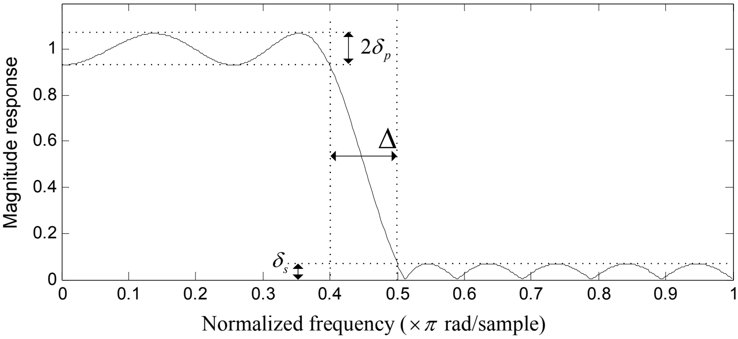


Figure A-1 Specifications for a lowpass FIR filter.

The estimated length of the FIR filter (see Figure A-1) with transition width Δ (with Δ being normalized to a sampling frequency of 2π), passband and stopband ripple δ_p and δ_s respectively, is given by:

$$\hat{N} = \frac{D_\infty(\delta_p, \delta_s)}{0.5\Delta/\pi} - 0.5\left[11.01 + 0.51(\log_{10}\delta_p - \log_{10}\delta_s)\right]\Delta/\pi + 1, \tag{A.1}$$

where:

$$\begin{aligned} D_\infty(\delta_p, \delta_s) = &\left[5.31\times10^{-3}(\log_{10}\delta_p)^2 + 7.11\times10^{-2}(\log_{10}\delta_p) - 4.76\times10^{-1}\right]\log_{10}\delta_s \\ &+\left[-2.66\times10^{-3}(\log_{10}\delta_p)^2 - 5.94\times10^{-1}(\log_{10}\delta_p) - 4.28\times10^{-1}\right]. \end{aligned} \tag{A.2}$$

Equations (A.1) and (A.2) can be further approximated under certain conditions. For the situation when the passband and stopband ripples are equal, i.e. $\delta = \delta_p = \delta_s$, then (A.1) can be simplified to:

$$\hat{N} = \frac{D_\infty(\delta)}{0.5\Delta/\pi} - 5.506\Delta/\pi + 1, \tag{A.3}$$

and (A.2) can be simplified to:

$$D_\infty(\delta) = 5.31\times 10^{-3}(\log_{10}\delta)^3 + 0.0685(\log_{10}\delta)^2$$
$$-1.07\log_{10}\delta - 0.428. \tag{A.4}$$

For relatively long filters, (A.1) can be approximated by:

$$\hat{N} = \frac{-40\log_{10}\sqrt{\delta_p\delta_s} - 26}{14.6\Delta/\pi} + 1. \tag{A.5}$$

Figure A-2 shows the estimated filter lengths for various values of Δ, when $\delta_p = 0.01 = 0.086$ dB and $\delta_s = 0.001 = -60$ dB. The dotted line represents the estimated filter length using (A.5) and the solid line represents the estimated filter length using (A.1). In general, (A.5) is a reasonable approximation of (A.1) for filter lengths which are greater than 20.

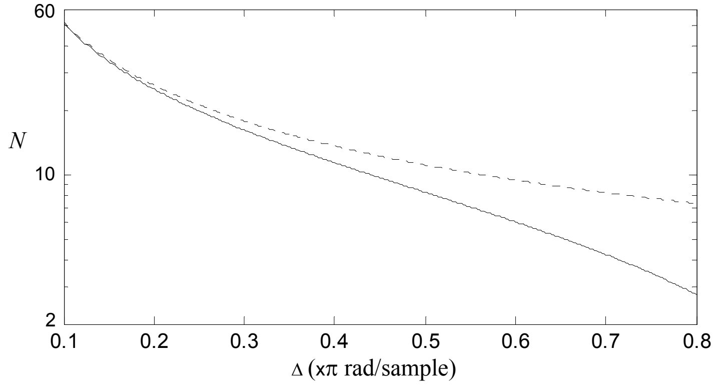


Figure A-2 Plot of estimated filter length against Δ for values of $\delta_p = 0.086$ dB and $\delta_s = -60$ dB.

Appendix B Complexity of a Filter

B.1. Overview

In this thesis, we normally assume that the complexity of a filter is approximately given by the number of real multiplications required per input sample processed into the filter. For example, in hardware designs, the processing time, power consumption and silicon area required by a multiplier are significantly greater than that of the adder and delay elements. We also make the general assumption that 4 real multiplications are required for the multiplication between 2 complex numbers, and 2 real multiplications are required for the multiplication between a complex number and a purely real/imaginary number.

In this appendix, we summarize and explain how we arrive at our complexity estimates for a variety of scenarios that are encountered throughout this thesis.

B.2. Summary

Consider an FIR filter $h(n)$ with length N, and input signal $x(n)$. We assume the causal form of the filter here, where $h(n)$ is defined from n=0 to n=N-1.

Filter	Description	Complexity	Comments
General-case FIR	$h(n)$ real, $x(n)$ real.	N	
	$h(n)$ real, $x(n)$ complex; or $h(n)$ complex, $x(n)$ real.	$2N$	
	$h(n)$ complex, $x(n)$ complex	$4N$	
Linear-phase FIR	$h(n)$ real, $x(n)$ is real:		
	N is odd.	$(N+1)/2$	
	N is even.	$N/2$	
Half-band FIR	$h(n)$ real, $x(n)$ real.	$1 + (N-3)/4$	Refer to Section B.3
Modulated linear-phase FIR, N is odd.	$H(e^{-j(\theta-d)}z)$:		Refer to Section B.4
	$x(n)$ is real.	N	
	$x(n)$ is complex.	$2N$	
	$H(e^{-j(\theta-d)}z),\ \theta = c\pi/4$: (where c is an arbitrary integer)		
	$x(n)$ is real.	$(N+1)/2$	
	$x(n)$ is complex.	$N+1$	
Taking the real part of the outputs: $[\![h(n)*x(n)]\!]$ N is odd.	General-case FIR filter:		Refer to Section B.5
	$h(n)$ real, $x(n)$ complex; or $h(n)$ complex, $x(n)$ real.	N	
	$h(n)$ complex, $x(n)$ complex	$2N$	

(*continued from previous page*)

Filter	Description	Complexity	Comments
	Linear-phase FIR filter:		
	$h(n)$ real, $x(n)$ complex; or	$(N+1)/2$	
	$h(n)$ complex, $x(n)$ real.		
	$h(n)$ complex, $x(n)$ complex	$(N+1)$	

Table B-1 Summary of filter complexities.

Note: We only listed the basic properties in the above table. The complexity of filters with mixed properties can be further derived. For example, a complex half-band FIR filter for a real input signal would have a complexity of $2[1 + (N-3)/4]$.

B.3. Half-band FIR filter

Half-band FIR filters have the following properties:

- N is odd, and N is defined as $N=4a+3$ where $a \geq 0$ is an integer;
- $h(n) = 0$ for all odd n except for $n=(N-1)/2$;
- $h\big((N-1)/2\big) = 0.5$;
- it is symmetrical about its center coefficient.

The delay d is equal to $(N-1)/2$. The convolution operation of $h(n)$ with $x(n)$ can be described by:

$$
\begin{aligned}
h(n) * x(n) &= \sum_{n'=0}^{N-1} h(n')x(n-n') \\
&= h(d)x(n-d) + \sum_{n'=0}^{(N-3)/4} h(2n')\big[x(n-2n') + x(n-N+1+2n')\big]. \qquad \text{(B.1)}
\end{aligned}
$$

The zero-valued coefficients require no computations. Furthermore, since the center coefficient is equal to 0.5, its computation does not require any multiplication and is trivial. Since $h(d)x(n-d) = x(n-d)/2$, we can simply perform a single-bit right-shift of $x(n'-n)$ to arrive at the result. Alternatively, we might choose to normalize the filter by a factor of 2 such that $h(d) = 1$.

Thus, the number of multiplications required for real $h(n)$ and real $x(n)$ is given by $a+1$. This is equal to $1 + (N-3)/4$.

B.4. Modulated linear-phase FIR filter

Consider the linear-phase FIR filter $H(z)$ with odd N, where all its coefficients are real. When the filter is modulated by an angle of θ radians, the resultant filter can be described by $H'(z) = H(e^{-j(\theta-d)}z)$, where $d=(N-1)/2$ is the delay of the filter.

Then, $H'(z)$ is conjugate-symmetrical:

$$\begin{aligned} h'(d+n) &= e^{j\theta n}h(d+n) \\ &= e^{j\theta n}h(d-n) \\ &= h'(d-n)^* \quad . \end{aligned} \tag{B.2}$$

The convolution operation of $h'(n)$ with $x(n)$ can be described by:

$$\begin{aligned} h'(n)*x(n) &= h'(d)x(n-d) + \sum_{n'=0}^{(N-3)/2} h'(n')\left[x(n-n')+x(n-N+1+n')\right] \\ &= h'(d)x(n-d) + \sum_{n'=0}^{(N-3)/2} \operatorname{Re}\{h'(n')\}\left[x(n-n')+x(n-N+1+n')\right] \\ &\quad + \sum_{n'=0}^{(N-3)/2} \operatorname{Im}\{h'(n')\}\left[x(n-n')-x(n-N+1+n')\right], \end{aligned} \tag{B.3}$$

where Re{.} denotes taking the real part of, and Im{.} denotes taking the imaginary part of.

When $x(n)$ is real, N real multiplications are required. When $x(n)$ is complex, $2N$ real multiplications are required.

Let us now consider a few special values of θ which we encounter in this thesis.

$\theta = \pi/4$:

The function $e^{j\pi n/4}$ takes on values with unity magnitude and a periodic phase difference of $\pi/4$. This expands into a series that takes on values of …, 1, $(1+j)/\sqrt{2}$, j, $(-1+j)/\sqrt{2}$, -1, $(-1-j)/\sqrt{2}$, -j, $(1-j)/\sqrt{2}$, … .

Therefore, the coefficients of the filter $H'(z) = H(e^{-j(\pi/2-d)}z)$ take on values that are either: (1) purely real, (2) purely imaginary, or (3) complex numbers with equal magnitude real and imaginary parts. Only 1 real multiplication is required per coefficient for such a filter when $x(n)$ is real.

$\theta = \pi/2$:

The function $e^{j\pi n/2}$ expands into a series that consists of purely real and purely imaginary numbers. Similarly, only 1 real multiplication is required per coefficient for such a filter when $x(n)$ is real.

In general, for a filter with transfer function $H'(z) = H(e^{-j(\theta-d)}z)$, where $\theta = c\pi/4$ and c is an arbitrary integer, the complexity in terms of real multiplications is taken to be equal to the real filter $H(z)$. However, we note that the number of additions is increased for the modulated filters.

B.5. Taking the real-outputs of a filter

We denote this operation as $[\![h(n) * x(n)]\!]$, or equivalently $\mathrm{Re}\{h(n) * x(n)\}$, where:

$$
\begin{aligned}
[\![h(n) * x(n)]\!] &= \left[\!\!\left[\sum_{n'=0}^{N-1} h(n')x(n-n') \right]\!\!\right] \\
&= \sum_{n'=0}^{N-1} [\![h(n')x(n-n')]\!] .
\end{aligned} \tag{B.4}
$$

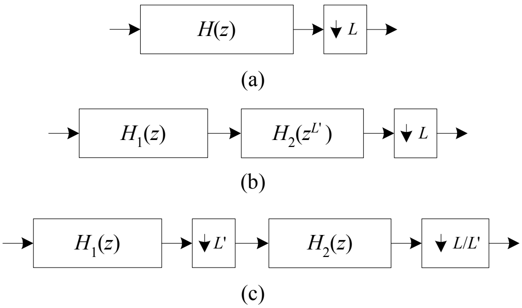
Consider the operation $[\![hx]\!]$, where h and x are two numbers. This can be evaluated as:

$$
[\![hx]\!] = \mathrm{Re}\{h\}\,\mathrm{Re}\{x\} - \mathrm{Im}\{h\}\,\mathrm{Im}\{x\} . \tag{B.5}
$$

When both the numbers h and x are complex, we require 2 real multiplications to evaluate $[\![hx]\!]$. When either h or x is complex and the other is real, we require 1 real multiplication to evaluate $[\![hx]\!]$. Thus, the evaluation of $[\![h(n) * x(n)]\!]$ requires $2N$ real multiplications if both $h(n)$ and $x(n)$ are complex and N real multiplications if only one of $h(n)$ and $x(n)$ is complex for a general-case FIR filter. For a linear phase FIR filter, the evaluation of $[\![h(n) * x(n)]\!]$ requires $N+1$ real multiplications if both $h(n)$ and $x(n)$ are complex and $(N+1)/2$ real multiplications if only one of $h(n)$ and $x(n)$ is complex.

B.6. Filter complexity in a multirate system

Consider the general-case FIR filter $H(z)$ with length N. The complexity of the filter is approximately N. Let us assume that the output of the filter is decimated by a factor of L, Figure B-1(a). The filtering operations need to be evaluated only once for every L input samples, and thus the complexity of the filter is reduced by a factor of L. Therefore, N/L multiplications are required per input sample. Consider now a cascade of 2 filters (Figure B-1(b)), $H_1(z)$ and $H_2(z^{L'})$ where the second filter is interpolated by a factor L'. Their lengths are N_1 and N_2 respectively. If L' is an integer factor of L, then that integer decimation factor can be carried to the front of H_2, as shown in Figure B-1(c). The complexity of the 2 filters can then be calculated as $N_1/L' + N_2/L$.



(a)

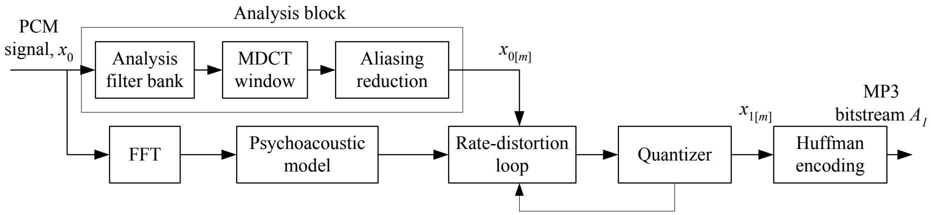
(b)

(c)

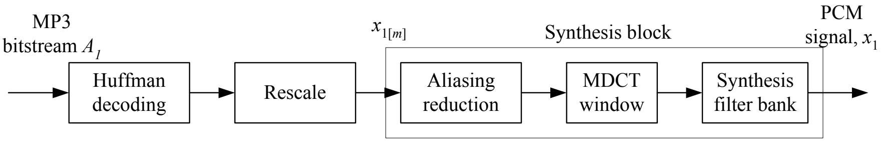
Figure B-1 Complexity of filters in a multirate system.

Appendix C MP3 Description

A detailed description of the MP3 method can be obtained from the ISO MP3 specifications [71]. In this appendix, we provide an overview of the MP3 method which is relevant to our work on audio transcoding. Figures C-1(a) and (b) show the general block diagrams for a typical MP3 encoder and decoder respectively.



(a) MP3 encoder block diagram.



(b) MP3 decoder block diagram.

Figure C-1 MP3 encoder and decoder block diagrams.

C.1. Filter bank block

The input PCM signal, which we denote by x_0, is first separated into its subband components $x_{0[m]}$ (for $0 \le m \le M-1$ where M is the total number of subbands). The analysis filter bank separates x_0 into 32 subbands and the Modified Discrete Cosine Transform (MDCT) window further separates each of these 32 subbands into 18 subbands (long window) or 6 (short window) subbands. Therefore, M is either 576 (long window) or 192 (short window).

The psychoacoustic model analyzes the audio content of the input and selects the window. Normally, long windows are selected (more than 90% of the time for a typical piece of music). Short windows are selected for sections of the audio material where there is a significant change in signal power in a short span of time, such as for staccato music. In this thesis, we simplify by assuming only long windows. The selection of short windows does not impact our results.

The aliasing reduction block is used to reduce the aliasing caused by overlap between adjacent subbands of the analysis filter bank. The spectral content of x_0 is concurrently analyzed using a 1024-point FFT and a psychoacoustic model, in a parallel branch.

We represent the subband samples in vector form consisting of elements $x_{0[m]}$:

$$\mathbf{x}_0 = \begin{bmatrix} x_{0[0]} & x_{0[1]} & \cdots & x_{0[M-1]} \end{bmatrix}^T, \tag{C.1}$$

where $\mathbf{x}_0$ is a vector of length M, and assumes real values from –1.0 to 1.0.

C.2. Scalefactor bands

The samples in the vector $\mathbf{x}_0$ are grouped into scalefactor bands. There are 22 scalefactor bands for long windows and 13 scalefactor bands for short windows. We represent the group of subband samples, which belongs to the scalefactor band denoted by s, using the vector $\mathbf{x}_{0\{s\}}$:

$$\mathbf{x}_{0\{s\}} = \begin{bmatrix} x_{0[m_s]} & x_{0[m_s+1]} & \cdots & x_{0[m_{s+1}-1]} \end{bmatrix}^T, \tag{C.2}$$

where m_s is the index of the subband corresponding to the first element of the scalefactor band s, $0 \le s \le 21$ for long windows and $0 \le s \le 12$ for short windows. Information on the values of m_s can be obtained from the MP3 specifications [71].

The vector $\mathbf{x}_0$ can be represented using the scalefactor band vectors:

$$\mathbf{x}_0 = \begin{bmatrix} \mathbf{x}_{0\{0\}} \\ \mathbf{x}_{0\{1\}} \\ \vdots \\ \mathbf{x}_{0\{21\}} \end{bmatrix}, \text{ for long windows and } \mathbf{x}_0 = \begin{bmatrix} \mathbf{x}_{0\{0\}} \\ \mathbf{x}_{0\{1\}} \\ \vdots \\ \mathbf{x}_{0\{12\}} \end{bmatrix} \text{ for short windows.} \tag{C.3}$$

The grouping of each scalefactor band varies (i.e. the values of m_s vary), depending on the sampling frequency (sampling frequencies of 32, 44.1 and 48 kHz are supported). Again, sampling frequency does not have an impact on our results, and we consider only the sampling frequency of 44.1 kHz in this thesis. For illustrative purposes, Figure C-2 shows the division of the subband samples into 22 scalefactor bands for a sampling frequency of 44.1 kHz and a long window. Subbands which are located between 2 vertical markers belong to the same scalefactor band. In this example, m_s=342 for s=20, and m_s=418 for s=21. Therefore, the scalefactor band vector for s=20 is denoted as $\mathbf{x}_{0\{21\}} = \begin{bmatrix} x_{0[342]} & x_{0[343]} & \cdots & x_{0[417]} \end{bmatrix}^T$.

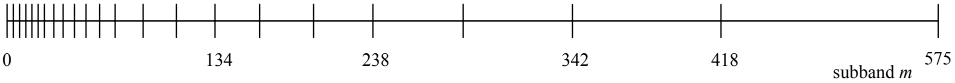


Figure C-2 Scalefactor band division for MP3, 44.1 kHz, long window.

C.3. Quantization

The objective of the rate-distortion loop is to minimize perceptual distortion, within the limits of the bit-rate constraint (e.g. 192 kbps). Perceptual distortion is determined by weighting the quantization error in each scalefactor band, using the psychoacoustic model. The quantization of the subband samples is performed on a per-scalefactor band basis. The subband samples in each scalefactor band, $\mathbf{x}_{0\{s\}}$ are quantized to the values $\mathbf{x}_{1\{s\}}$ by the quantizer $q_{1\{s\}}$:

$$\begin{aligned} x_{1[m]} &= q_{1\{s\}}\left(x_{0[m]}\right) \\ &= \pm\left|\tilde{x}_{1[m]}\right|^{4/3} \cdot 2^{\mathbb{Z}_{g1}/4-\mathbb{Z}_{\alpha 1}\mathbb{Z}_{f1\{s\}}+\mathbb{Z}_{\phi 1}}, \end{aligned} \tag{C.4}$$

where $x_{0[m]}$ is an element of $\mathbf{x}_{0\{s\}}$, $x_{1[m]}$ is an element of $\mathbf{x}_{1\{s\}}$, and:

$\tilde{x}_{1[m]}$	is an integer, and is constrained to take on values in the range of -8206 to 8206 (according to the MP3 specifications);
$\mathbb{Z}_{g1}$	is the value of the global gain;
$\mathbb{Z}_{f1\{s\}}$	is the value of the scalefactor;
$\mathbb{Z}_{\alpha 1}$	is the scalefactor multiplier and takes on a value of either 0.5 or 1 (selected by the encoder);
$\mathbb{Z}_{\phi 1}$	consists of other encoder-specific variables. These variables are sometimes used for fine-tuning and do not have a significant impact in our context;

and the signs of $x_{1[m]}$ and $\tilde{x}_{1[m]}$ are taken equal to the sign of $x_{0[m]}$.

The quantizer $q_{1\{s\}}$ is defined by the lumped quantization parameter $\mathbb{Z}_{1\{s\}}$, where $\mathbb{Z}_{1\{s\}} = -\mathbb{Z}_{g1}/+\mathbb{Z}_{\alpha 1}\mathbb{Z}_{f1\{s\}} - \mathbb{Z}_{\phi 1}$. The quantizers used for MP3 are non-uniform, i.e. the quantization step-sizes are not equal. The amount of compression achieved (bit-rate) is mainly determined by the number of bits required to code the integer vector $\tilde{\mathbf{x}}_1$. Low bit-rates are usually achieved by increasing the global gain $\mathbb{Z}_{g1}$ or decreasing the scalefactor $\mathbb{Z}_{f1\{s\}}$.

The quantization error is given by:

$$\begin{aligned} e_{1[m]} &= q_{1\{s\}}\left(x_{0[m]}\right) - x_{0[m]} \\ &= 2^{\mathbb{Z}_{1\{s\}}}\left\{\Re\left[\left(x_{0[m]}2^{-\mathbb{Z}_{1\{s\}}}\right)^{3/4}\right]\right\}^{4/3} - x_{0[m]} \\ &= 2^{\mathbb{Z}_{1\{s\}}}\left\{\Re\left[\left(x_{0[m]}2^{-\mathbb{Z}_{1\{s\}}}\right)^{3/4}\right] - \left(x_{0[m]}2^{-\mathbb{Z}_{1\{s\}}}\right)^{3/4}\right\}^{4/3}, \end{aligned} \tag{C.5}$$

where $\Re[.]$ denotes rounding the bracketed term to its nearest integer. Since the value of $\Re\left[\left(x_{0[m]}2^{-\mathbb{Z}_1}\right)^{3/4}\right] - \left(x_{0[m]}2^{-\mathbb{Z}_1}\right)^{3/4}$ is distributed between -0.5 and 0.5 (error of rounding a real number to its nearest integer), we can see from (C.5) that the magnitude of the quantization error $e_{1[m]}$ tends to increase with $2^{\mathbb{Z}_1}$ as $\mathbb{Z}_1$ is increased. Therefore, decreasing the bit-rate constraint increases the coarseness of the quantization.

C.4. MP3 frame

The quantized vector $\tilde{\mathbf{x}}_1$ is Huffman-encoded (lossless) to further reduce the bit-rate. The Huffman-encoded data, quantizer parameters and information relevant to reconstruction are then arranged into a bitstream (one-dimensional string of bits), which we denote as A_1. An MP3 bitstream is made up of a series of MP3 frames. The organization of an MP3 frame is shown in Figure C-3.

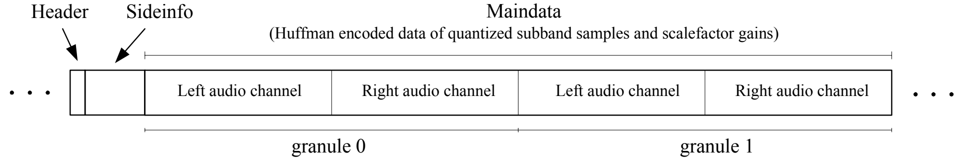


Figure C-3 Layout of an MP3 frame.

Header – Contains general information about the encoded frame, such as the compression method (e.g. MP3), encoded bit-rate, number of audio channels, and sampling frequency, etc.

Sideinfo – Contains information about the composition of the current MP3 frame, such as the selection of short or long windows, number of bits used for the encoding, and the Huffman codes that were used, etc.

Maindata – Consists of 2 granules. An MP3 granule contains information for the left and right audio channels. Each audio channel contains 576 samples of quantized subband samples $\tilde{x}_{1[m]}$ (which are Huffman-encoded). In the case of short windows (M=192), 3 consecutive vectors of $\tilde{\mathbf{x}}_1$ form a granule-channel. In the case of long windows (M=576), $\tilde{\mathbf{x}}_1$ forms a granule-channel by itself. For MP3, a single-channel of a single granule is equivalent to a quantization-frame (q-frame, see Section 7.4.1).

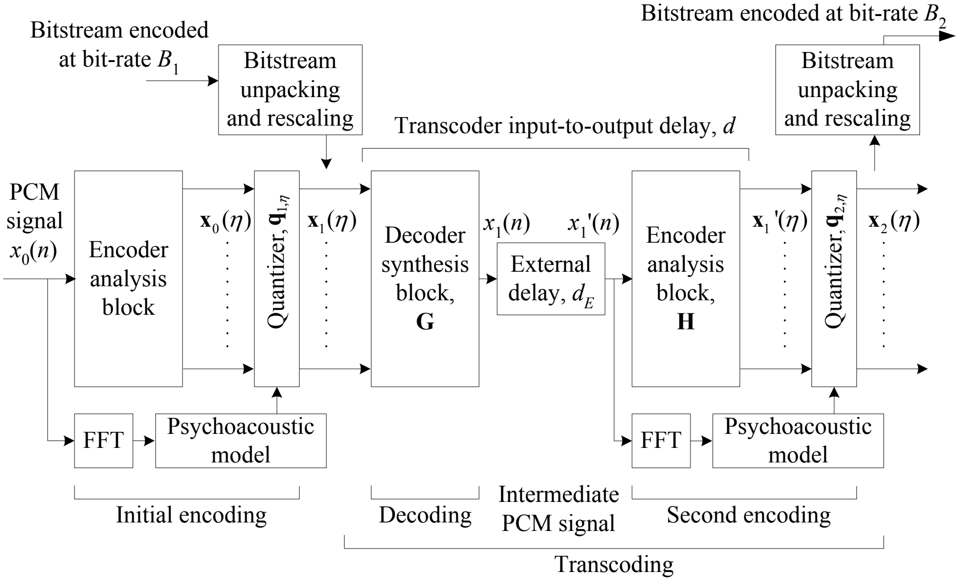
C.5. Decoding

The MP3 bitstream A_1 can be decoded back to the PCM signal x_1 by using an MP3 decoder. The decoder functions are the reverse of the encoder functions. The MP3 bitstream is first Huffman-decoded. The real-valued subband samples $x_{1[m]}$ are retrieved by rescaling the integer subband data $\tilde{x}_{1[m]}$. The PCM signal x_1 is then reconstructed from $x_{1[m]}$ by using the aliasing reduction block, MDCT and synthesis filter bank.

Appendix D Input-to-output Relationship for a Transcoder

This appendix provides mathematical substantiation to the section on sample-synchronization in Chapter 7.

D.1. System overview



Bitstream encoded at bit-rate B_1
Bitstream unpacking and rescaling
Bitstream encoded at bit-rate B_2
Bitstream unpacking and rescaling
Transcoder input-to-output delay, d
PCM signal $x_0(n)$
Encoder analysis block
$\mathbf{x}_0(\eta)$
Quantizer, $\mathbf{q}_{1,\eta}$
$\mathbf{x}_1(\eta)$
Decoder synthesis block, **G**
$x_1(n)$
$x_1'(n)$
External delay, d_E
Encoder analysis block, **H**
$\mathbf{x}_1'(\eta)$
Quantizer, $\mathbf{q}_{2,\eta}$
$\mathbf{x}_2(\eta)$
FFT
Psychoacoustic model
FFT
Psychoacoustic model
Initial encoding
Decoding
Intermediate PCM signal
Second encoding
Transcoding

(Repeat of Figure 7-4.)

We refer to the transcoder model shown in Figure 7-4, repeated here for convenience. Note that n and η are time indices for different sampling rates. The sampling rate of η is $1/M$ (M being the total number of subbands) times that of the sampling rate of n for the purpose of our discussion. The description of Figure 7-4 was provided in Section 7.3. In this appendix, we focus on the transcoding stage. Consider the cascaded synthesis-analysis blocks **G** and **H**. The inputs to these blocks are denoted by $\mathbf{x}_1(\eta)$ and the outputs by $\mathbf{x}'_1(\eta)$, where:

$$\mathbf{x}_1(\eta) = \begin{bmatrix} x_{1[0]}(\eta) & \cdots & x_{1[m]}(\eta) & \cdots & x_{1[M-1]}(\eta) \end{bmatrix}^T, \tag{D.1}$$

and:

$$\mathbf{x}'_1(\eta) = \begin{bmatrix} x'_{1[0]}(\eta) & \cdots & x'_{1[m]}(\eta) & \cdots & x'_{1[M-1]}(\eta) \end{bmatrix}^T. \tag{D.2}$$

Let $x_{1[m]\uparrow}(n)$ be the interpolated version of $x_{1[m]}(\eta)$:

$$x_{1[m]\uparrow}(n)=\begin{cases} x_{1[m]}(\eta), & \text{when } n=M\eta \\ 0, & \text{otherwise.}\end{cases} \tag{D.3}$$

Let $x'_{1[m]\uparrow}(n)$ be the interpolated version of $x'_{1[m]}(\eta)$:

$$x'_{1[m]\uparrow}(n)=\begin{cases} x'_{1[m]}(\eta), & \text{when } n=M\eta \\ 0, & \text{otherwise.}\end{cases} \tag{D.4}$$

D.2. Mathematical description of transcoding stage

Define $x_{1hg[m]\uparrow}(n)$ to be the m-th output of **H** prior to being decimated by a factor of M, when **H** is cascaded with **G**. The input-to-output relationship of the cascaded synthesis-analysis blocks from $x_{1[m]\uparrow}(n)$ to $x_{1hg[m]\uparrow}(n)$ is:

$$x_{1hg[m]\uparrow}(n)=h_{[m]}(n)*\left(\sum_{m'=0}^{M-1} g_{[m']}(n)*x_{1[m']\uparrow}(n)\right), \tag{D.5}$$

where $*$ is the convolution operator.

Define the cascaded transfer function $F_{[m',m]}(z)=G_{[m']}(z)H_{[m]}(z)$, or in the time domain:

$$f_{[m',m]}(n)=g_{[m']}(n)*h_{[m]}(n)\,. \tag{D.6}$$

Then:

$$x_{1hg[m]\uparrow}(n)=\sum_{m'=0}^{M-1}\sum_{n'=0}^{N-1} f_{[m',m]}(n')x_{1[m']\uparrow}(n-n')\,, \tag{D.7}$$

where N is the length of the transfer function $F_{[m',m]}(z)$.

Since $x_{1[m]\uparrow}(n)$ is interpolated by a factor of M, $x_{1[m]\uparrow}(n)$ consists of only 1 non-zero value for every M samples. Define the variables n_0 and n_1, such that $0\le n_1\le M-1$ and $n = n_0+n_1$. Furthermore, let n_0 take on values such that:

$$x_{1hg[m]\uparrow}(n_0)=\sum_{m'=0}^{M-1}\sum_{n'=0}^{(N/M-1)} f_{[m',m]}(Mn')x_{1[m']\uparrow}(n_0-Mn')\,, \tag{D.8}$$

where we assume that N is a multiple of M. In the case that N is not a multiple of M, we can simply append zeroes at the end of $F_{[m',m]}(z)$ until N is a multiple of M, without affecting the result.

Note: We can alternatively view n_0 as the set of sample times at which $x_{1[m]\uparrow}(n)$ takes on non-zero values. The variable n_0 takes on the set of values that are incremented by M each time.

Then, $x_{1hg[m]\uparrow}(n)$ can be expressed as:

$$x_{1hg[m]\uparrow}(n_0+n_1)=\sum_{m'=0}^{M-1}\sum_{n'=0}^{(N/M-1)} f_{[m',m]}(n_1+Mn')x_{1[m']\uparrow}(n_0-Mn')\,. \tag{D.9}$$

Since $x'_{1[m]}(\eta)$ is decimated by a factor of M, $x'_{1[m]\uparrow}(n)$ consists of only 1 non-zero value for every M samples. Let the sample time at which $x'_{1[m]\uparrow}(n)$ is non-zero be $n=n_0+d_D$, where d_D is a constant value and $0\le d_D\le M-1$. Then:

$$\begin{aligned} x'_{1[m]\uparrow}(n_0+d_D)&=x_{1hg[m]\uparrow}(n_0+d_D)\\ &=\sum_{m'=0}^{M-1}\sum_{n'=0}^{(N/M-1)} f_{[m',m]}(d_D+Mn')x_{1[m']\uparrow}(n_0-Mn')\,, \end{aligned} \tag{D.10}$$

and:

$$x'_{1[m]\uparrow}(n_0+n_1)=0\,,\text{ when } n_1\ne d_D\,. \tag{D.11}$$

The delay of the cascaded synthesis-analysis blocks (represented by the set of transfer functions $F_{[m',m]}(z)$) is defined as d_F. The external delay (such as delays inserted by the user) is defined as d_E. Then, the system is said to be sample-synchronized if the total delay d is a multiple of M:

$$d=d_E+d_F-d_D=kM\ , \tag{D.12}$$

where k is an integer.

For a pair of PR synthesis-analysis blocks, when the system is sample-synchronized, the output $x'_{1[m]\uparrow}(n)$ becomes a delayed version of the input $x_{1[m]\uparrow}(n)$, i.e. $x'_{1[m]\uparrow}(n+kM)=x_{1[m']\uparrow}(n)$. The values of $x'_{1[m]\uparrow}(n)$ are:

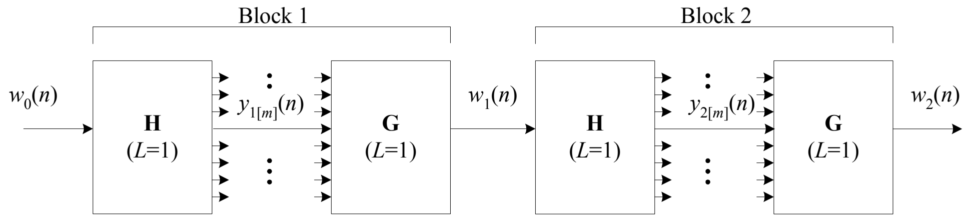
$$x'_{1[m]\uparrow}(n_0+kM)=x_{1[m']\uparrow}(n_0)\,,$$

$$x'_{1[m]\uparrow}(n_0+n_1+kM)=x_{1[m']\uparrow}(n_0+n_1)=0\,,\text{ for } 1\le n_1\le M-1\,. \tag{D.13}$$

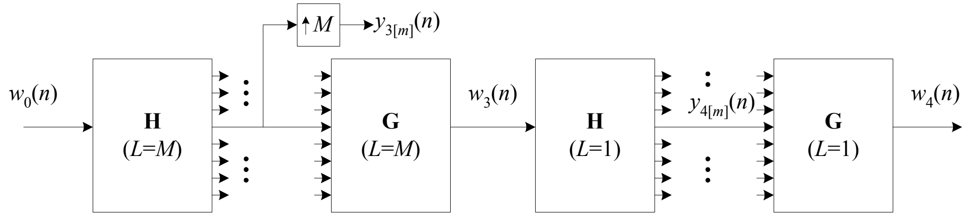
This is proved in the next sub-section.

D.3. Proofs

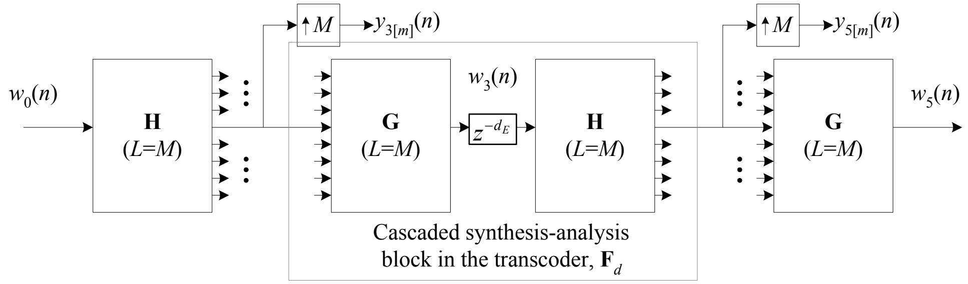
In this section, we prove that $x'_{1[m]\uparrow}(n+kM)=x_{1[m']\uparrow}(n)$ for a pair of PR synthesis-analysis blocks, when the total delay d is a multiple of M.



(a) Blocks 1 and 2 are single-rate.



(b) Block 1 is multi-rate, Block 2 is single-rate.



(c) Blocks 1 and 2 are multi-rate.

Figure D-1 Illustration of cascaded analysis-synthesis blocks having different decimation factors.

Figure D-1 illustrates the notations for the signals $w_0(n),\ldots, w_5(n)$ and $y_{0[m]}(n),\ldots, y_{5[m]}(n)$ that are used in this section. It is to be noted that in this section, the signals which we represent using the symbols $w_0(n),\ldots, w_5(n)$ and $y_{0[m]}(n),\ldots, y_{5[m]}(n)$ are not to be associated with signals having the same symbol outside of this section.

Let the pair of analysis-synthesis blocks **H** and **G** be perfectly reconstructing when they are critically decimated (L=M), and the sum of their delays be d_F. The requirement to prove that $x'_{1[m]\uparrow}(n+kM) = x_{1[m']\uparrow}(n)$, when $d = d_E + d_F - d_D = kM$, can be established by proving that $y_{5[m]}(n+kM) = y_{3[m]}(n)$ when $d = kM$, in the context here (Figure D-1(c)).

Consider the system in Figure D-1(a), where the analysis and synthesis blocks are operated in single-rate mode, i.e. the decimation and interpolation factors L=1.

Property: When **H** and **G** are operated in single-rate mode (L=1), the input-to-output PR property is retained, i.e. $w_1(n)$ and $w_2(n)$ are delayed versions of $w_0(n)$.

Proof of property: A PR pair of **H** and **G** when operated in critically-decimated mode (L=M) has the property (refer to Section 6.1):

$$\sum_{m=0}^{M-1} H_{[m]}(z)G_{[m]}(z) = z^{-d_F} . \tag{D.14}$$

When the same **H** and **G** are operated in single-rate mode (L=1), (D.14) exactly describes the input-to-output (from $w_0(n)$ to $w_1(n)$) transfer function of a pair of **H** and **G**. Therefore, $w_2(n+2d_F) = w_1(n+d_F) = w_0(n)$ in Figure D-1(a).

Since $y_{1[m]}(n) = h_{[m]}(n) * w_0(n)$ and $y_{2[m]}(n) = h_{[m]}(n) * w_1(n) = h_{[m]}(n) * w_0(n-d_F)$, we further conclude that $y_{2[m]}(n)$ is a delayed version of $y_{1[m]}(n)$:

$$y_{2[m]}(n + d_F) = y_{1[m]}(n) . \tag{D.15}$$

Consider now the system in Figure D-1(b), where the first set of analysis-synthesis blocks is critically decimated (L=M) and the second set is single-rate (L=1). Since **H** and **G** are PR when L=M, $w_4(n+2d_F) = w_3(n+d_F) = w_0(n)$. Relate $y_{3[m]}(n)$ in Figure D-1(b) to $y_{1[m]}(n)$ in Figure D-1(a):

$$y_{3[m]}(n_0 + n_1) = \begin{cases} y_{1[m]}(n_0 + n_1), & \text{when } n_1 \text{ is a multiple of } M \\ 0, & \text{when } n_1 \neq 0, \end{cases} \tag{D.16}$$

where n_0 is the set of sample times when $y_{3[m]}(n)$ takes on non-zero values, and takes on values which are incremented by M each time; and n_1 is an integer such that $n = n_0 + n_1$ and $0 \le n_1 \le M - 1$.

Since **H** and **G** are PR, $w_3(n+d) = w_0(n)$. Thus, $w_3(n) = w_1(n)$.

It follows that $y_{4[m]}(n) = y_{2[m]}(n)$. Thus, $y_{4[m]}(n_0 + d_F) = y_{3[m]}(n_0)$.

Consider finally the system in Figure D-1(c), where both sets of analysis-synthesis blocks are critically decimated (L=M). Relate $y_{5[m]}(n)$ in Figure D-1(c) to $y_{4[m]}(n)$ in Figure D-1(b):

$$y_{5[m]}(n_0 + n_1) = \begin{cases} y_{4[m]}(n_0 + n_1 - d_E), & \text{when } n_1 = d_D \\ 0, & \text{when } n_1 \neq d_D, \end{cases} \tag{D.17}$$

where d_D is a constant ($0 \le d_D \le M - 1$), such that $y_{5[m]}(n)$ takes on non-zero values only at the sample times given by $n = n_0 + d_D$, and d_E is a delay external to the transcoder, that is arbitrarily chosen by the user. In practice, this can be easily performed by delaying the decoded PCM signal $w_3(n)$, prior to the encoding stage.

Therefore, $y_{5[m]}(n_0 + d_D) = y_{1[m]}(n_0 + d_D - d_F - d_E)$.

Since $y_{1[m]}(n_0+d_D-d_F-d_E)=y_{3[m]}(n_0+d_D-d_F-d_E)$ only when $d_D-d_F-d_E$ is a multiple of M (i.e. $d_F+d_E-d_D=kM$, where k is an integer), $y_{5[m]}(n_0+d)=y_{3[m]}(n_0)$ only when:

$$d=d_F+d_E-d_D=kM\ . \tag{D.18}$$

When d is a multiple of M, we say that the transcoding system is 'sample-synchronized'. When the transcoding system is not sample-synchronized, the external delay d_E can be varied to enforce sample-synchronization.

D.4. Summary

Let the set of transforms relating the inputs $\mathbf{x}_1(\eta)$ of the cascaded synthesis-analysis blocks **H** and **G** to the outputs $\mathbf{x}'_1(\eta)$ be represented simplistically as:

$$\mathbf{x}'_1(\eta)=\mathbf{F}_d\left(\mathbf{x}_1(\eta)\right), \tag{D.19}$$

where $\mathbf{x}_1(\eta)$ and $\mathbf{x}'_1(\eta)$ are vectors comprising $x_{1[m]}(\eta)$ and $x'_{1[m]}(\eta)$ respectively (see (D.1) and (D.2)).

The transform operator $\mathbf{F}_d$ consists of $f_{[m',m]}$ for $0\le m,m'\le M-1$ (see (D.6)), and is described as follows.

Let $x_{1[m]\uparrow}(n)$ be the interpolated version of $x_{1[m]}(\eta)$ (see (D.3)), and $x'_{1[m]\uparrow}(n)$ be the interpolated version of $x'_{1[m]}(\eta)$ (see (D.4)). Then:

$$\begin{aligned} x'_{1[m]\uparrow}(n_0+d_D) &= x_{1hg[m]\uparrow}(n_0+d_D) \\ &= \sum_{m'=0}^{M-1}\sum_{n'=0}^{(N/M-1)} f_{[m',m]}(d_D+Mn')x_{1[m']\uparrow}(n_0-Mn')\,, \end{aligned} \tag{D.20}$$

where n_0 takes on values such that $x_{1[m]\uparrow}(n_0)$ are non-zero values, and d_D takes on values such that $x'_{1[m]\uparrow}(n_0+d_D)$ are non-zero values.

Let the total delay of the transcoder be d, where $d=d_E+d_F-d_D$, d_E is the external delay, and d_F is the delay due to only **H** and **G**.

When d is equal to 0 or a multiple of M, the transcoding system is sample-synchronized. The output becomes a delayed version of the input:

$$\mathbf{x}'_1(\eta+d/M)=\mathbf{x}_1(\eta)\,. \tag{D.21}$$

Appendix E Listening Test Results

In Chapter 10, an ultra-fast method of implementing an MP3 bit-rate transcoder was proposed. Listening tests were conducted based on our implementation of the transcoder in the C programming language. The platform used for our tests was Microsoft Windows running on an Intel Pentium 3-1 GHz processor. In this appendix, we provide a detailed overview of the conduct of the listening tests as well as the results.

E.1. Listening test overview

We performed 2 separate listening tests. Listening Test 1 was targeted at our implementation of a fixed mapping transcoder, and Listening Test 2 was targeted at our implementation of an adaptive mapping transcoder.

In each listening test, listeners were asked to listen to a range of audio material. The audio material was processed using both our methods and conventional methods of encoding and transcoding. For the conventional encoders/decoders, we use the Fast MP3 encoder (fmp3ENC) and the Fast MP3 decoder (fmp3DEC) ([110]). The fmp3ENC and fmp3DEC are two very efficient and high quality MP3 encoder and decoder respectively, developed by the Philips PDSL Sound Coding Group. The various encoding/transcoding methods that were used in the generation of audio samples for the listening test are shown in Figure E-1. The generated audio samples are designated labels from *LT-1* to *LT-6*.

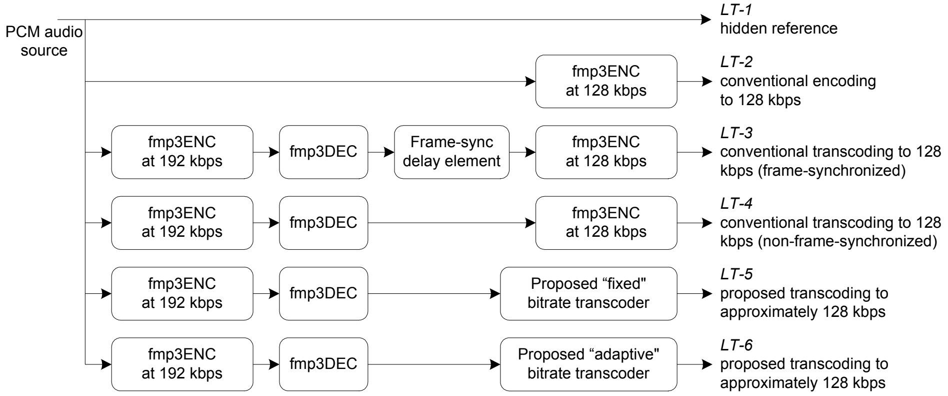

PCM audio source
LT-1
hidden reference
fmp3ENC at 128 kbps
LT-2
conventional encoding to 128 kbps
fmp3ENC at 192 kbps
fmp3DEC
Frame-sync delay element
fmp3ENC at 128 kbps
LT-3
conventional transcoding to 128 kbps (frame-synchronized)
fmp3ENC at 192 kbps
fmp3DEC
fmp3ENC at 128 kbps
LT-4
conventional transcoding to 128 kbps (non-frame-synchronized)
fmp3ENC at 192 kbps
fmp3DEC
Proposed "fixed" bitrate transcoder
LT-5
proposed transcoding to approximately 128 kbps
fmp3ENC at 192 kbps
fmp3DEC
Proposed "adaptive" bitrate transcoder
LT-6
proposed transcoding to approximately 128 kbps

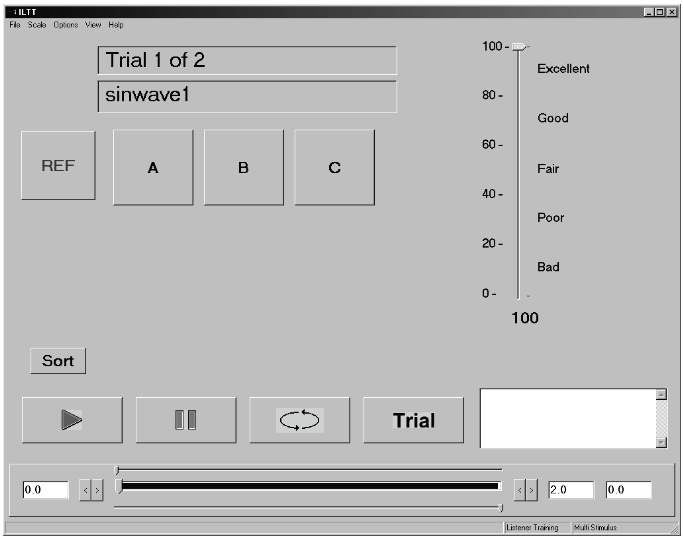
Figure E-1 Illustration of the various encoding/transcoding methods used to generate the listening test samples.

The listeners then provided an opinion score (on a scale of 0 to 100) to judge their perceived audio quality of the material. The EBU MUSHRA (European Broadcasting Union Multi-stimulus test with hidden reference and anchors) 100-points scale [111]-[115] was used (see Table E-1).

Score	Label
100	
90	Excellent
80	
70	Good
60	
50	Fair
40	
30	Poor
20	
10	Bad

Table E-1 The grading scale used in the listening test.

Figure E-2 shows the user interface for the listening test tool.



Trial 1 of 2

sinwave1

100

0.0

2.0

0.0

Figure E-2 User interface for the listening test tool.

E.2. Listening test setup

The listening test was held in an enclosed listening room with minimal external interference. The test was conducted using headphones. Eight listeners took part in the test.

The test contained 9 stereo items (see Table E-2), and the sampling frequency used for all items is 44.1 kHz. These 9 items have been used in the development of MP3 audio encoders and are known to be critical to encoding artifacts during compression.

For each session, there are 9 trials, 1 trial for each of the 9 stereo items. One trial consists of 4 different encodings, as well as the original (used as a hidden reference). The order of the trials and items are randomized for each session. It takes a listener about twenty minutes to half an hour to complete the listening test.

Item no.	Item name	Duration in seconds
S-1	Castanets	7
S-2	Eye in the sky (by Alan Parson's Project)	17
S-3	Pop music (by ABBA)	27
S-4	Tom's Diner (by Suzan Vega)	10
S-5	Pitch pipe	27
S-6	Trumpet	10
S-7	Layla (by Eric Clapton)	19
S-8	Harpsichord	7
S-9	Orchestral	12

Table E-2 Items used in the listening test.

E.3. Listening Test 1 and 2 summary results

The following charts plot the results of the listening tests. The results for Listening Test 1 was shown in Figure E-3 (repeated from Figure 10-13), and the results for Listening Test 2 was shown in Figure E-4 (repeated from Figure 10-14). We repeat these 2 figures below for convenience. The x-axis shows the encoding/transcoding method used (see Figure E-1). The y-axis represents the Mean Opinion Score (MOS) according to the EBU MUSHRA 100-points scale. The MOS is the average of the individual opinion scores, and is marked with a cross 'x'. The 95% confidence interval range is indicated with a delimited vertical line.

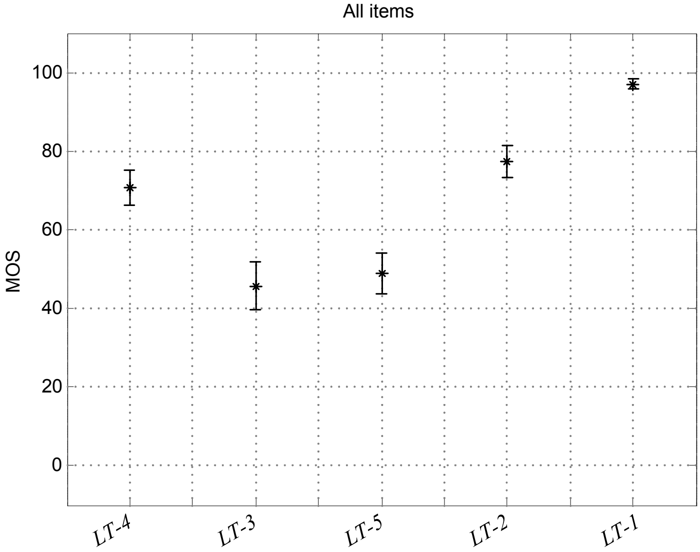


Figure E-3 (Repeat of Figure 10-13) - Results of Listening Test 1.

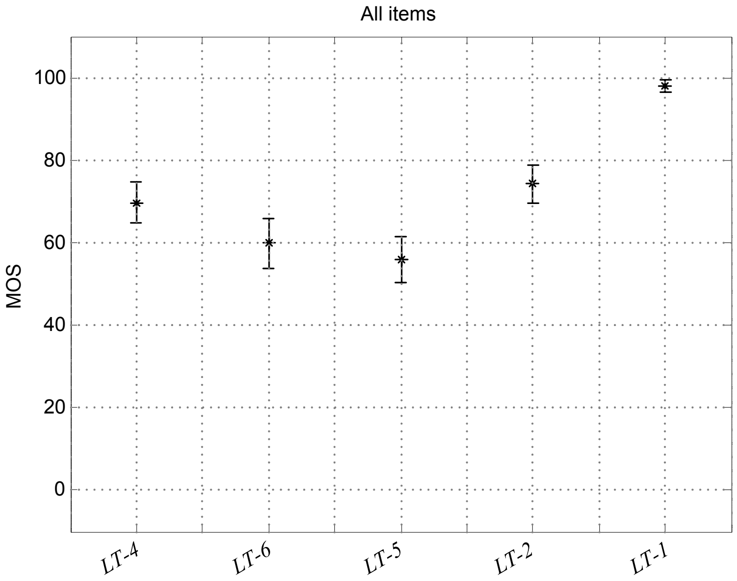
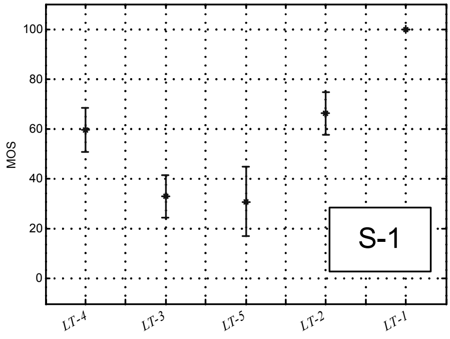
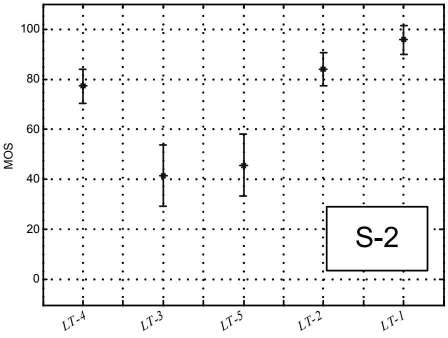


Figure E-4 (Repeat of Figure 10-14) - Results of Listening Test 2.

E.4. Listening Test 1 individual results

The following figures show the individual results for Listening Test 1, for the items numbering from S-1 to S-9.





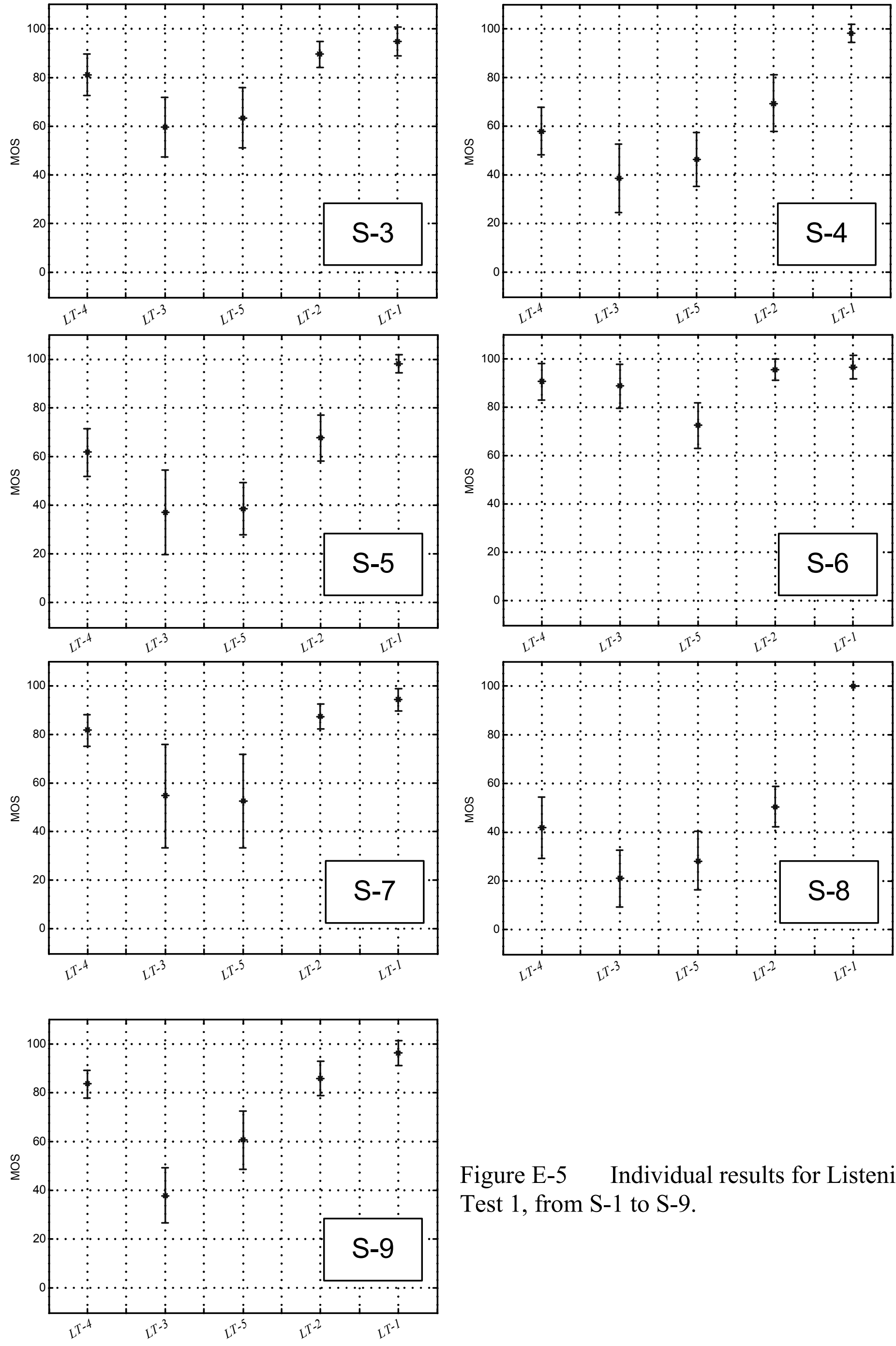
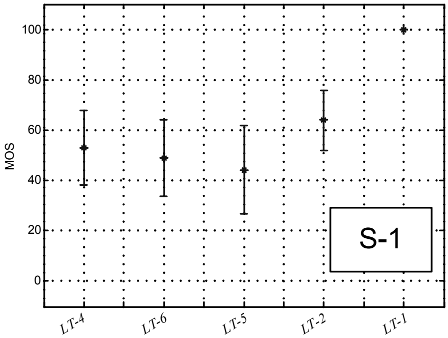
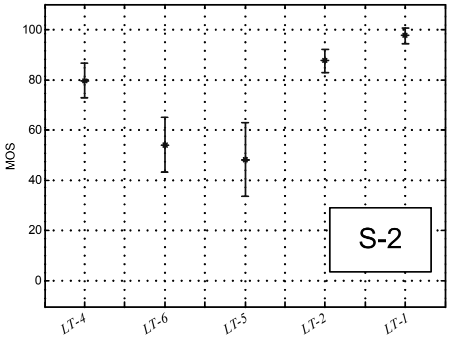


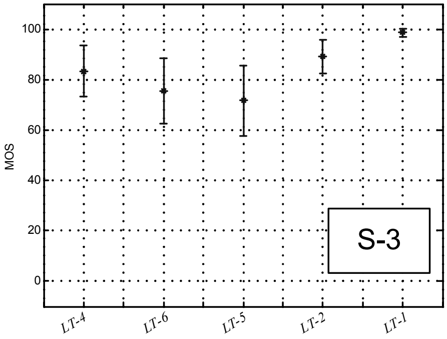
Figure E-5 Individual results for Listening Test 1, from S-1 to S-9.

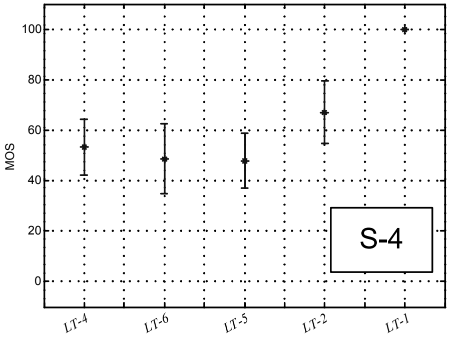
E.5. Listening Test 2 individual results

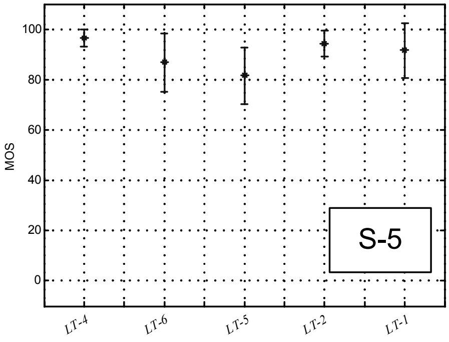
The following figures show the individual results for Listening Test 2, for the items numbering from S-1 to S-9.

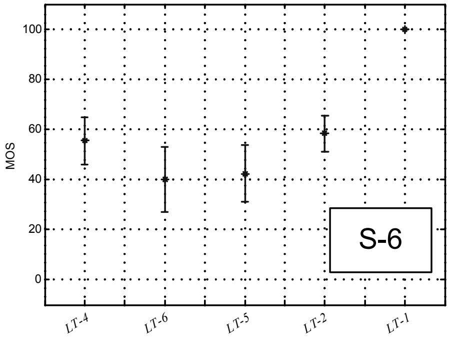


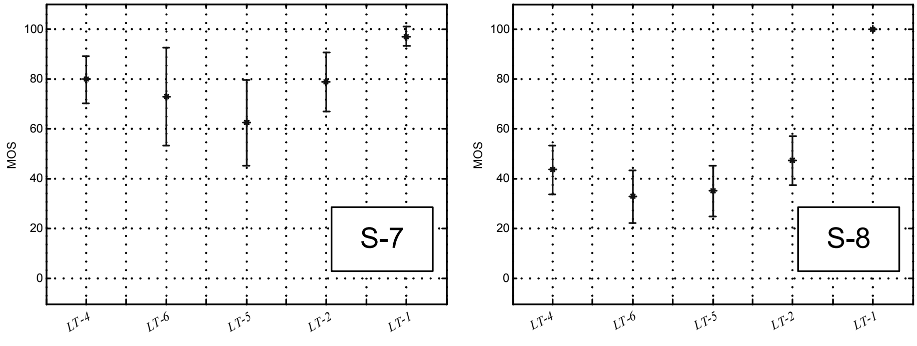












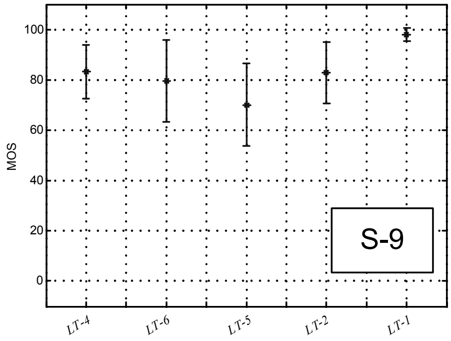


Figure E-6 Individual results for Listening Test 2, from S-1 to S-9.

E.6. Conclusions

Listening Test 1 (Figure 10-13) shows that our fixed mapping transcoder (*LT-5*) performs reasonably well, when compared to the conventional frame-synchronized transcoder (*LT-3*). The MOS for *LT-5* is slightly higher than *LT-3*, and the confidence intervals overlap significantly. It can be seen that the non-frame-synchronized transcoder (*LT-4*) results in better audio quality than both the frame-synchronized methods, but is still inferior to non-transcoded 128 kbps material (*LT-2*).

In Listening Test 2 (Figure 10-14), item *LT-3* was replaced by *LT-6* (which represents our adaptive mapping transcoder). It can be seen that the MOS for the adaptive mapping transcoder (*LT-6*) is slightly higher than the fixed mapping transcoder (*LT-5*). Furthermore, its upper confidence interval range overlaps with the lower confidence interval range of the non-frame-synchronized transcoder (*LT-4*). Contrast this with the fixed mapping transcoder (*LT-5*), which does not overlap with *LT-4*.

Samenvatting

Audio transcodering is de conversie van digitale audio van een gecomprimeerde vorm A naar een gecomprimeerde vorm B met andere compressie-eigenschappen, zoals bitsnelheid, bemonsteringsfrequentie, of compressiemethode. De gebruikelijke methode van transcodering is om eerst A te decomprimeren, gevolgd door een compressiestap naar B. De hiertoe benodigde rekenoperaties betreffen in belangrijke mate de synthesefilterbank die gebruikt wordt bij de decompressiestap, en de analysefilterbank die gebruikt wordt bij de daaropvolgende compressiestap.

In dit proefschrift, dat bestaat uit twee delen, worden methoden voor efficiënte implementatie van filterbanken en audio transcoders gepresenteerd. In het eerste deel wordt een nieuwe klasse van Frequency Response Masking (FRM) filterbanken geïntroduceerd. Deze filterbanken worden normaliter gekarakteriseerd als een cascade van subfilters in een boomstructuur, waarbij de individuele filters een kleine lengte hebben. Er worden diverse methoden van complexiteitsreductie voorgesteld voor scenario’s waarbij de filterbanken in single rate of multi-rate mode werken, alsook voor scenario’s waarbij een reëelwaardig of complex ingangssignaal gebruikt wordt. Een efficiënte variabele bandbreedte FRM filterbank wordt ontworpen door gebruik te maken van 'signed-powers-of-two' reductie van de subfilter coëfficiënten. Ons ontwerp heeft een complexiteit die een orde van grootte lager is dan die van een octaaf filterbank met gelijke specificaties.

In het tweede deel wordt het audio transcodeerproces geanalyseerd. Het transcoderen van audio wordt gemodelleerd als een gecascadeerd quantisatie proces, wat vervolgens voor een ingangssignaal onder verschillende condities geanalyseerd wordt voor de MPEG 1 Layer 2 en MP3 compressie methoden. Een belangrijke parameter is de transcodeervertraging van ingang tot uitgang, die een invloed heeft op de uiteindelijke audiokwaliteit. Er worden ook methoden voorgesteld om de fout in een gecascadeerd quantisatieprocess te verkleinen. Tenslotte presenteren we een ultra-snelle MP3 transcoder die slechts gebruik maakt van integer operaties en geïmplementeerd is in software. Onze implementatie laat een verbetering in executiesnelheid zien met een factor 5 tot 16 ten opzichte van andere best bekende transcoders.

Curriculum Vitae

Jun Wei, Lee was born in Singapore on October 26, 1976. He received the B.Eng (Electrical) degree from the National University of Singapore in 2001 with First Class Honours and the Vice-Chancellor's List award. From 2001 to 2006, he worked on research towards a Joint PhD degree between the National University of Singapore (NUS) and the Technische Universiteit Eindhoven (TU/e), Netherlands. In 2005, he joined the Signal Processing System-on-Chip Group in Temasek Laboratories, Nanyang Technological University, Singapore.

His research interests and strengths lie mainly in digital signal processing, filter banks and audio compression technologies. In addition, he also possesses supporting proficiencies in the area of IC design and communications fundamentals.

During his PhD candidature, he worked mainly on the design and implementation of digital filters and multirate filter banks. His work includes the design of highly efficient filters and filter banks using the Frequency Response Masking technique, and their reduction using signed-powers-of-two coefficients. From 2002 to 2004, he worked on a collaborative project between the Defence Science Organization (Singapore) and NUS, in the area of efficient filter bank design and hybrid filter bank parallel ADC design. In 2004, he carried out his research on audio transcoding in the Signal Processing Systems Group in TU/e. During this period, he was also a guest researcher at the Sound Coding Group, Philips Digital Systems Laboratory at Eindhoven, Netherlands.