KU LEUVEN
FACULTEIT INGENIEURSWETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK
AFDELING ESAT-STADIUS
Kasteelpark Arenberg 10 – B-3001 Leuven

ADAPTIVE FILTERING ALGORITHMS FOR ACOUSTIC ECHO CANCELLATION AND ACOUSTIC FEEDBACK CONTROL IN SPEECH COMMUNICATION APPLICATIONS

Promotoren:
Prof. dr. ir. M. Moonen
Prof. dr. ir. T. van Waterschoot
Prof. dr. ir. S. H. Jensen

Proefschrift voorgedragen
tot het behalen van de
graad van Doctor in de
Ingenieurswetenschappen

door

Jose Manuel GIL-CACHO

December 2013

KU LEUVEN
FACULTEIT INGENIEURSWETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK
AFDELING ESAT-STADIUS
Kasteelpark Arenberg 10, B-3001 Heverlee

ADAPTIVE FILTERING ALGORITHMS FOR ACOUSTIC ECHO CANCELLATION AND ACOUSTIC FEEDBACK CONTROL IN SPEECH COMMUNICATION APPLICATIONS

Jury:
Prof. dr. A. Bultheel, chairman
Prof. dr. ir. M. Moonen, promotor
Prof. dr. ir. T. van Waterschoot, co-promotor
Prof. dr. ir. S. H. Jensen, promotor
(Aalborg University, Denmark)
Prof. dr. ir. J. Swevers, assessor
Prof. dr. ir. J. Wouters, assessor
Prof. dr. ir. S. Gannot
(Bar Ilan University, Israel)
Prof. dr. ir. P. C. W. Sommen
(Eindhoven University of Technology, The Netherlands)

Proefschrift voorgedragen
tot het behalen van de
graad van Doctor in de
Ingenieurswetenschappen

door

Jose Manuel GIL-CACHO

December 2013

ISBN 978-94-6018-758-2

D/2013/7515/143

Voorwoord

A five-year journey has come to an end and I am finally ready to write the preface for my PhD thesis. At this moment, I just want to express my gratitude and countless thanks to all of those who have helped me during my PhD.

I would like to thank Prof. Marc Moonen for giving me the opportunity to join his research group, for his guidance and good ideas. I remember our first meeting when I was asking for this opportunity. Although I sometimes thought that he regretted his decision, he gave me freedom to choose my own way and always showed respect for my own ideas. I learned lots of things from him, some of them worth keeping my entire life.

In that difficult task of finding my own way, Prof. Toon van Waterschoot has been essential all these years. I remember my first ever conference paper, which he encouraged me to write. The fact that he was interested in the idea was something truly motivating. After that conference paper, everything started to be easier. We had very interesting discussions and more ideas came after that. I deeply thank him for all those evening he spent proofreading my messy papers and for becoming a friend. The support and feedback from Prof. Søren Holdt Jensen has without any doubt been very helpful. His suggestions for improvement on my papers were always very much appreciated. My deepest gratitude to the jury members: Prof. Adhemar Bultheel, Prof. Jan van Wouters, Prof. Jan Swevers, Prof. Sharon Gannot and Prof. Piet C. W. Sommen for their time, effort, and valuable comments and suggestions to improve my thesis.

I would like to thank Prof. Johan Shoukens for the time I spent in the ELEC department of the Vrije Universiteit Brussels. He introduced me to the exciting world of nonlinear system identification and honestly I could not imagine this thesis without his initial guidance. I would like to thanks Prof. Johan Suykens for bringing the idea of kernel adaptive filtering and to Dr. Marco Signoretto for collaborating with me on this idea. Thanks to them, an important part of this thesis (and my first ever published journal) has been possible.

To the research group at the KU Leuven and to the people in the SIGNAL project, thank you all for the many travels, the courses we had together and for the inspiring discussions. Thanks to Dr. Alexander Bertrand (Belly) for being my guarding angel all these years. For reminding me all bank holidays, how to ask for a refund, how to book a room, how to re-enroll (every year) and a long etc. I am afraid that without him I was still trying to logging to the intranet. Thanks to almost-Dr. Joseph Szurley (Joe) for being my *brocito* in a period of my life when I really needed it. The middle-life middle-PhD crisis came together at once, however making fun of the same things and sharing great moments with Joe was indeed healing. Thanks to Bruno for being my Dutch translator, to Amin, Giuliano, Giacomo, Niccolo, Rodolfo, Marijn, Aldona, Paschalis, Rodrigo, Kim and Javi. Thanks a lot to Ida for doing all those things that I forgot to do in the most inconvenient moment, without her I would not be here. Thanks of course to all the administrative staff.

I would like to thanks my mother for telling me since I was a kid 'you can do it', to my father for teaching me to stay alert from myself and to my sister for making me feel special. Thanks to my boyfriend for reminding me how fun life can be, to my friends for making things easier and full of joy and relatives to encourage me to continue. Thanks to my wife Ines for her endless love and patience, for always being my best friend, for understanding when I had to work during the weekends, for cheering me up when life was difficult, for sharing the fun of big moments, for always having a moment for me and for showing me that nothing can go wrong if we are together. Thank you also for having our son Jose, the most beautiful thing on earth (after this thesis).

Jose, every day you make me understand the actual meaning of infinity.

Jose Manuel Gil-Cacho Leuven, December 2013

Abstract

Multimedia consumer electronics are nowadays everywhere from teleconferencing, hands-free communications, in-car communications to smart TV applications and more. We are living in a world of telecommunication where ideal scenarios for implementing these applications are hard to find. Instead, practical implementations typically bring many problems associated to each real-life scenario. This thesis mainly focuses on two of these problems, namely, acoustic echo and acoustic feedback. On the one hand, acoustic echo cancellation (AEC) is widely used in mobile and hands-free telephony where the existence of echoes degrades the intelligibility and listening comfort. On the other hand, acoustic feedback limits the maximum amplification that can be applied in, e.g., in-car communications or in conferencing systems, before howling due to instability, appears. Even though AEC and acoustic feedback cancellation (AFC) are functional in many applications, there are still open issues. This means that many of the issues associated to practical AEC and AFC are overlooked. In this thesis, we contribute to the development of a number of algorithms to tackle the main issues associated to AEC and AFC namely, (1) that very long room impulse responses (RIRs) make standard adaptive filters converge slowly and lead to a high computational complexity, (2) that double-talk (DT) situations in AEC and model mismatch in AFC distort the near-end signal , and (3) that the nonlinear response of some of the elements forming part of the audio chain makes linear adaptive filters fail.

In the view of computational complexity reduction, we consider introducing the concept of common-acoustical-pole room modeling into AEC. To this end, we perform room transfer function (RTF) equalization by compensating for the main acoustic resonances common to multiple RTFs in the room. We discuss the utilization of different norms (i.e., 2-norm and 1-norm) and models (i.e., all-pole and pole-zero) for RTF modeling and equalization. A computationally cheap extension from single-microphone AEC to multi-microphone AEC is then presented for the case of a single loudspeaker. The RTF models used for multi-microphone AEC share a fixed common denominator polynomial, which is calculated off-line by means of a multi-channel warped linear prediction. This allows high computational savings. In the context of acoustic feedback

control, we develop a method for acoustic howling suppression based on adaptive notch filters (ANF) with regularization (RANF). This method achieves frequency tracking, howling suppression and improved howling detection in a one-stage scheme. The RANF approach to howling suppression introduces minimal processing delay and minimal complexity, in contrast to non-parametric block-based methods that feature a non-parametric frequency analysis in a two-stage scheme.

To tackle the issue of robustness to double-talk (DT) in AEC and robustness to model mismatch in AFC, a new adaptive filtering framework is proposed. It is based on the frequency-domain adaptive filtering (FDAF) implementation of the so-called PEM-AFROW (FDAF-PEM-AFROW) algorithm. We show that FDAF-PEM-AFROW is related to the best, i.e., minimum-variance, linear unbiased estimate (BLUE) of the echo path. We derive and define the instantaneous pseudo-correlation (IPC) measure between the near-end signal and the loudspeaker signal. The IPC measure serves as an indication of the effect of a DT situation occurring during adaptation due to the correlation between these two signals. Based on the good results obtained using FDAF-PEM-AFROW, we derive a practical and computationally efficient algorithm for DT-robust AEC and for AFC. The proposed algorithm features two modifications in the FDAF-PEM-AFROW: (a) the WIener variable Step sizE (WISE), and (b) the GRAdient Spectral variance Smoothing (GRASS), leading to the WISE-GRASS-FDAF-PEM-AFROW. The WISE modification is implemented as a single-channel noise reduction Wiener filter where the Wiener filter gain is used as a variable step size in the adaptive filter. On the other hand, the GRASS modification aims at reducing the variance in the noisy gradient estimates based on time-recursive averaging of instantaneous gradients.

In the last part of this thesis, the nonlinear response of (active) loudspeakers forming part of the audio chain is studied. We consider the description of odd and even nonlinearities in (active) loudspeakers by means of periodic random-phase multisine signals. The fact that the odd nonlinear contributions are more predominant than the even ones implies that at least a 3rd-order nonlinear model of the loudspeaker should be used. Therefore, we consider the identification and validation of a model of the loudspeaker using several linear-in-the-parameters nonlinear adaptive filters, namely, Hammerstein and Legendre polynomial filters of various orders, and a simplified 3rd-order Volterra filter of various lengths. In our measurement set-up, the obtained results imply however, that a 3rd-order nonlinear filter fails to capture all the nonlinearities, meaning that odd and even nonlinear contributions are produced by higher-order nonlinearities. High-order Volterra filters are impractical in AEC and AFC due to their large computational complexity together with inherently slow convergence. On the other hand, the kernel affine projection algorithm (KAPA) has been successfully applied to many areas in signal processing but not yet to nonlinear AEC (NLAEC). In KAPA, and kernel methods in general,

the kernel trick is applied to work implicitly in a high-dimensional (possibly infinite-dimensional) space without having to transform the input data into this space. This is one of the most appealing characteristics of kernel methods, as opposed to nonlinear adaptive filters requiring explicit nonlinear expansions of the input data as, for instance, Volterra filters. In fact, all computations can be done by evaluating the kernel function in the input space. This fact provides powerful modeling capabilities to kernel adaptive algorithms where the computational complexity will be determined by the input dimension, independent of the order of the nonlinearity. Our contributions in this context are to apply KAPA to the NLAEC problem, to develop a sliding-window leaky KAPA (SWL-KAPA) that is well suited for NLAEC applications, and to propose a suitable kernel function, consisting of a weighted sum of a linear and a Gaussian kernel.

Korte Inhoud

Multimedia-consumentenelektronica is dezer dagen overal te vinden, voor toepassingen zoals teleconferentie, handenvrije communicatie, voertuigcommunicatie, en intelligente TV. We leven in een wereld van telecommunicatie, waar ideale scenario's om deze toepassingen te implementeren zelden voorkomen. Praktische implementaties daarentegen brengen typisch vele problemen mee, gekoppeld aan het realistische scenario waarin men zich bevindt. Dit doctoraatsproefschrift richt zich voornamelijk op twee van deze problemen, namelijk, akoestische echo en akoestische feedback. Aan de ene kant wordt akoestische-echo-onderdrukking (AEC) op grote schaal aangewend in mobiele en handenvrije telefonie, waar de aanwezigheid van echo's de verstaanbaarheid en het luistercomfort aantast. Aan de andere kant begrenst akoestische feedback de maximale versterking die kan worden toegepast in bv. voertuigcommunicatie of in teleconferentie, vooraleer fluittonen optreden door instabiliteit. Hoewel AEC en akoestische-feedback-onderdrukking (AFC) functioneel zijn in vele toepassingen, bestaan er nog een aantal open problemen. Dit betekent dat vele van de problemen gekoppeld aan praktische AEC- en AFC-systemen tot nog toe over het hoofd worden gezien. Dit doctoraatsproefschrift draagt bij tot de ontwikkeling van een aantal algoritmen die de belangrijkste problemen gekoppeld aan AEC en AFC aanpakken, namelijk, (1) dat zeer lange kamerimpulsresponsies (RIRs) de standaard adaptieve filters trager doen convergeren en tot een hoge rekencomplexiteit leiden, (2) dat situaties met dubbelspraak (DT) in AEC en modelafwijkingen in AFC het microfoonsignaal vervormen en (3) dat de niet-lineaire responsie van sommige elementen in de audio-keten de werking van lineaire adaptieve filters doet mislukken.

Met het oog op een reductie van de rekencomplexiteit, introduceren we het concept van kamermodellering via gemeenschappelijke-akoestische-polen in AEC. Hiertoe voeren we een egalisatie van de kamerakoestische overdrachtsfunctie (RTF) uit door de belangrijkste akoetische resonanties te compenseren die gemeenschappelijk zijn voor meerdere RTFs binnen de kamer. We bespreken het gebruik van verschillende normen (nl. 2-norm en 1-norm) en modellen (nl. enkel-polen en pool-nulpunt) voor de modellering en egalisatie van RTFs. Vervolgens wordt een uitbreiding met lage rekencomplexiteit voorgesteld van AEC

met n microfoon naar AEC met meerdere microfoons voor het geval van een enkele luidspreker. De RTF-modellen gebruikt voor AEC met meerdere microfoons delen een vaste gemeenschappelijke noemerveelterm, die off-line berekend wordt via een meerkanaals verdraaide lineaire predictie. Dit laat aanzienlijke besparingen in rekencomplexiteit toe. In de context van akoestische-feedback-beheersing, ontwikkelen we een methode voor akoestische fluittoononderdrukking gebaseerd op adaptieve inkepingsfilters (ANF) met regularisatie (RANF). Deze methode bewerkstelligt frequentietracking, fluittoononderdrukking en een verbeterde fluittoondetectie in een n-staps-schema. De RANF-aanpak voor fluittoononderdrukking introduceert een minimale vertraging en heeft een minimale rekencomplexiteit, in tegenstelling tot niet-parametrische venstergebaseerde methodes, die een niet-parametrische frequentieanalyse uitvoeren in een twee-staps-schema.

Om het probleem van situaties met dubbelspraak (DT) in AEC en het probleem van robuustheid tegen modelafwijkingen in AFC op te lossen, wordt een nieuw adaptief-filter-raamwerk voorgesteld. Het is gebaseerd op de adaptieve filtering frequentiedomeinimplementatie (FDAF) van het zogenaamde PEM-AFROW (FDAF-PEM-AFROW) algoritme. We tonen aan dat FDAF-PEM-AFROW gerelateerd is met de beste, d.i. de minimale-variantie, lineaire zuivere schatting (BLUE) van het echo-pad. We definiren de instantane pseudo-correlatie (IPC) tussen het microfoonsignaal en het luidsprekersignaal. De IPC-maat geeft een indicatie van het effect van een DT-situatie die tijdens de adaptatie voorkomt door de correlatie tussen deze twee signalen. Op basis van de goede resultaten verkregen met FDAF-PEM-AFROW, leiden we een praktisch en efficint algoritme af voor DT-robuuste AEC en voor AFC. Het voorgestelde algoritme bevat twee wijzigingen ten opzichte van FDAF-PEM-AFROW: (a) de Wiener variabele stapgrootte (WISE), en (b) de gradint spectrale variantie smoothing (GRASS), die samen leiden tot het WISE-GRASS-FDAF-PEM-AFROW-algoritme. De WISE-wijziging is gemplementeerd als een nkanaals Wiener-ruisonderdrukkingsfilter, waarbij de Wiener-filterversterking aangewend wordt als variabele stapgrootte in het adaptieve filter. Anderzijds heeft de GRASS-wijziging als doel om de variantie in de ruizige gradintschattingen te verkleinen door middel van een tijdsrecursieve uitmiddeling van instantane gradinten.

In het laatste deel van dit doctoraatsproefschrift wordt de niet-lineaire responsie bestudeerd van (actieve) luidsprekers die deel uitmaken van de audioketen. We beschouwen de beschrijving van oneven en even niet-lineariteiten in (actieve) luidsprekers door middel van periodische multisine-signalen met willekeurige fase. Het feit dat de oneven niet-lineaire bijdragen de even niet-lineaire bijdragen overheersen, impliceert dat een niet-lineair luidsprekermodel van minstens derde orde moet gebruikt worden. Daarom beschouwen we de identificatie en validatie van een luidsprekermodel via verschillende niet-lineaire adaptieve filters die lineair zijn in de parameters, nl. Hammerstein

en Legendre veelterm-filters van verschillende ordes, alsook een vereenvoudigd Volterra-filter van derde orde met verschillende lengtes. Resultaten verkregen via onze meetopstelling impliceren evenwel dat een niet-lineair filter van derde orde er niet in slaagt alle niet-lineariteiten te vatten, wat betekent dat oneven en even niet-lineaire bijdragen geproduceerd worden door niet-lineariteiten van hogere ordes. Volterra filters van hogere ordes zijn niet praktisch voor AEC en AFC omwille van hun hoge rekencomplexiteit en hun inherent trage convergentie. Anderzijds is het kernel-affiene-projectie-algoritme (KAPA) met succes toegepast in verscheidene signaalverwerkingsdomeinen, maar nog niet in niet-lineaire AEC (NLAEC). Bij KAPA, en kernel-methodes in het algemeen, wordt de kernel-truc toegepast om impliciet in een hoog-dimensionale (mogelijk oneindig-dimensionale) ruimte te werken, zonder daarom de ingangsdata naar deze ruimte te moeten transformeren. Dit is n van de meest aantrekkelijke kenmerken van kernel-methodes, in tegenstelling tot niet-lineaire adaptieve filters, die een expliciete niet-lineaire expansie van de ingangsdata vereisen, zoals bijv. Volterra-filters. Dit bezorgt kernel-adaptieve algoritmes krachtige modelleringsmogelijkheden, aangezien de rekencomplexiteit bepaald wordt door de ingangsdimensie, onafhankelijk van de orde van de niet-lineariteit. In deze context bestaat onze bijdrage uit het toepassen van KAPA op het NLAEC-probleem, uit het ontwikkelen van een glijdend-venster lekke KAPA (SWL-KAPA), die geschikt is voor NLAEC-toepassingen, en uit het voorstellen van een gepaste kernel-functie, die bestaat uit een gewogen som van een lineaire en een Gaussiaanse kernel.

Glossary

Acronyms and Abbreviations

(K)APA	(Kernel) Affine Projection Algorithm
(N)LMS	(Normalized) Least Mean Squares
(NL)AEC	(Nonlinear) Acoustic Echo Cancellation
(R)LS	(Recursive) Least Squares
A/D	Analog-to-Digital
AFC	Acoustic Feedback Cancellation
AFROW	AFC using Row Operations
ANF	Adaptive Notch Filter
ANSI	American National Standards Institute
AR	Auto Regressive
Att_{max}	Maximum Attenuation
Att_{mean}	Mean Attenuation
BLUE	Best Linear Unbiased Estimator
CAP	Common Acoustical Poles
CPSD	Cross power spectral density
CPZLP	Constrained Pole-Zero Linear Prediction
CVX	Disciplined Convex Programming
D/A	Digital-to-Analog
dB	Decibel
DFT	Discrete Fourier Transform
DT	Double Talk
DTD	Double Talk Detector
ERB	Equivalent Rectangular Bandwidth
ERLE	Echo Return Loss Enhancement
FDAF	Frequency-Domain Adaptive Filter
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FLOPS	Floating-point Operations Per Second
FRF	Frequency Response Function
GLS	Generalized Least Squares
GRASS	Gradient Spectral Variance Smoothing

HA	Hearing aids
Hz	Hertz
IDFT	Inverse DFT
IFFT	Inverse FFT
IIR	Infinite Impulse Response
IPC	Instantaneous pseudo-correlation
kHz	Kilohertz
LEM	Loudspeaker-Enclosure-Microphone
LNLR	Linear-to-Nonlinear Ratio
LP	Linear Prediction
LPC	Linear Prediction Coding
MA	Moving Average
MAE	Mean Absolute Error
MSD	Misadjustment
MSG	Maximum Stable Gain
NHS	Notch-Filter-Based Howling Suppression
NPVSS	Non-Parametric VSS
PA	Public address
PCVSS	Projection-Correlation VSS
PE	Prediction Error
PEM	Prediction Error Method
PSD	Power Spectral Density
PVSS	Practical VSS
RANF	Regularized Adaptive Notch Filter
RHS	Right Hand Side
RIR	Room Impulse Response
RTF	Room Transfer Function
ROW	Row Operations
SD	Sparseness Degree
SD_{max}	Maximum Frequency-Weighted Log-Spectral Signal Distortion
SD_{mean}	Mean Frequency-Weighted Log-Spectral Signal Distortion
SER	Signal-to-Echo Ratio
SF	Spectral Flatness
SNR	Signal-to-Noise Ratio
SW	Sliding Window
TD-NLMS	Transform-Domain NLMS
VoIP	Voice over IP
VR	Variable Regularization
VSS	Variable Step Size
WGN	White Gaussian Noise
WISE	Wiener Variable Step Size
WLP	Warped Linear Prediction
WLS	Weighted Least Squares

e.g.	*exempli gratia*: for example
i.e.	*id est*: that is
w.r.t	with respect to

Mathematical notation

Common notation

Scalars:	small *italic* letters, α, d(i)
Vectors:	small **bold** letters, $\mathbf{w}$, $\boldsymbol{\omega}$, $\mathbf{c}(i)$
Vectors:	small letters with explicit index, $e_k(i), \quad k = 1, .., P$
Matrices:	capital **BOLD** letters, $\mathbf{U}(i)$, $\mathbf{\Phi}(i)$
Matrices:	capital letters with explicit indexes, $G_{(P-k+1,P-j+1)}(i), \quad k, j = 1, .., P$
Time dependency:	Indexes in parentheses, $\mathbf{u}$(i), d(i)
Vector entry:	Subscript indexes, $a_j(i-1)$, $e_k(i)$
Linear Spaces:	Capital LaTeX 'mathbb' letters, $\mathbb{F}$, $\mathbb{H}$
Scalar constants:	Capital *ITALIC* letters, L, N
q^{-1}	time shift operator
$\|\cdot\|_p$	p-norm
$\circ$	element-wise multiplication
$\mathbb{R}^L$	L-dimensional real space
O$(\cdot)$	of the order of a number
$J(\cdot)$	cost function
$\mathbf{I}$	identity matrix
$\mathcal{F}\{\cdot\}$	FFT operator
$\mathcal{F}^{-1}\{\cdot\}$	inverse FFT operator
$\mathcal{E}\{\cdot\}$	expectation operator
$(\cdot)^T$	transpose operation
$(\cdot)^H$	conjugate and transpose operation
$\mathbf{A}^{-1}$	inverse of matrix $\mathbf{A}$
$[\cdot]_{a:b}$	range from a to b elements in a vector

Chapter 2

$p(k)$	kth common pole
$z_i(k,t)$	kth distinct ith zero at time t
$a(k)$	kth common AR coefficient
$b_i(k,t)$	kth distinct ith MA coefficient at time t
$H_i(q,t)$	ith RTF at time t
Q	order of numerator (zeros)
P	order of denominator (poles)

$\mathbf{v}$	matrix of RIR
$\mathbf{W}$	compound matrix
$\mathbf{W}_i$	Toeplitz matrix made with ith RIR
$\mathbf{x}$	coefficient vector
p	order of the norm
$\mathbf{h}_i$	ith measured RIR
$\mathbf{a}$	AR coeffients from measured RIR
$\mathbf{b}_i$	MA coefficients from measured RIR
$\mathbf{D}$	selective matrix
$A(q)$	filter polynomial in q
$\tilde{B}_i(q)$	residual RTF
$\tilde{A}(q)$	approximately $A^{-1}(q)$
$R(n)$	residual RIR
$P(f)$	magnitude of fth frequency
N	DFT size
T	threshold

Chapter 3

$x(t)$	input/loudspeaker signal
$y_i(t)$	ith echo signal
$d(t)$	desired/microphone signal
$e(t)$	error signal
$\hat{y}_i(t)$	ith estimated echo signal
$H_i(q)$	ith RTF
$B_i(q)$	model numerator polynomial in q
$A_c(q)$	model common denominator polynomial in q
$p_c(k)$	kth common pole
$a_c(k)$	kth common AR coefficient
$D(q,\lambda)$	first-order all-pass filter
λ	warping parameter
$H_i^w(q)$	ith warped RTF
$\hat{h}_i^w(t)$	ith all-pole-estimated warped impulse response
$a_c^w(k)$	kth warped common AR coefficient
M	number of AECs (channels)
$\mathbf{a}$	vector of warped common AR coefficient
$\mathbf{W}$	matrix of Toeplitz matrices from measured and warped RIR
$\mathbf{v}$	matrix of measured and warped RIR
$\mathbf{h}_i^w$	measured and warped RIR
$\mathbf{H}_i$	Toeplitz matrix of measured and warped RIR
$\mathbf{b}_i(t), \quad \hat{B}_i(q,t)$	ith channel's adaptive filter coefficients
N	RIR length

n	ERLE time index
i	channel index

Chapter 4

w_0	radial notch frequency
f_0	notch frequency
f_s	sampling frequency
r	pole radius
$H(q)$	notch filter transfer function
$a(n)$	instant frequency updating parameter
$u(n)$	filter state
$t(n)$	filter state
$x(n)$	input signal to RANF
$y(n)$	output signal from RANF
$\Delta_a(n)$	gradient (search direction)
i	index of RANF
λ	regularization parameter
∇f	frequency difference
T	threshold
$S_y(f)$	spectrum of $y(n)$
$S_x(f)$	spectrum of $x(n)$
L	buffer size in decision rule

Chapter 5

$\hat{\mathbf{f}}(t), \hat{F}(q,t)$	estimated echo path
$\hat{\mathbf{F}}(k)$	DFT estimated echo path
$\tilde{\mathbf{f}}(t)$	echo path due to DT
$\tilde{\mathbf{F}}(k)$	DFT echo path due to DT
$u(t)$	input/loudspeaker signal
$\mathbf{u}(t)$	input/loudspeaker signal vector
$\hat{x}(t)$	estimated echo signal
$x(t)$	echo signal
$v(t)$	near-end signal
$e(t)$	error signal
$y(t)$	microphone signal
$y_a[t, \hat{h}(t)]$	prefiltered microphone signal using $\hat{h}(t)$
$u_a[t, \hat{h}(t)]$	prefiltered loudspeaker signal using $\hat{h}(t)$
$e_a[t, \hat{h}(t), \hat{f}(t-1)]$	prefiltered error signal using $\hat{h}(t)$
$e[t, \hat{f}(t)]$	error signal calculated with $\hat{f}(t)$

$\mathbf{e}(t)$	error signal vector
$\mathbf{e}_a(t)$	prefiltered error signal vector
$\mathbf{y}_a$	prefiltered microphone signal vector
$\hat{x}[t, \hat{f}(t)]$	estimated echo signal calculated with $\hat{f}(t)$
$\mathbf{f}$	(true) echo path model parameters
$\mathbf{y}$	vector of microphone signal samples
$\mathbf{v}$	vector of near-end signal samples
$\mathbf{X}$	Toeplitz matrix of loudspeaker samples
$\hat{\mathbf{f}}_{\text{GLS}}$	estimated echo path coefficients minimizing a GLS criterion
$\hat{\mathbf{f}}_{\text{BLUE}}$	estimated echo path coefficients minimizing a BLUE criterion
$\hat{\mathbf{F}}_{\text{BLUE}}$	DFT estimated echo path coefficients minimizing a BLUE criterion
$\hat{\mathbf{f}}_{\text{LS}}$	estimated echo path coefficients minimizing a LS criterion
$\mathbf{M}$	weighting matrix in GLS
$\mathbf{M}_{\text{PEM}}$	weighting matrix in BLUE in PEM
$\mathbf{R}_v$	near-end signal autocorrelation matrix
$w(t)$	near-end excitation signal (white noise)
$H(q,t)$	near-end signal model parameter
$A(q,t)$	inverse near-end signal model parameter
$\hat{A}(q,t)$	estimated inverse near-end signal prefilter
$\mathbf{H}(q)$	matrix of near-end signal model parameters at different time instants
$\mathbf{w}$	vector of near-end signal excitation at different time instants
$a_k(t)$	kth autoregressive model parameter
$\hat{\mathbf{a}}(t)$	estimated AR model parameter vector
n_A	near-end signal model and prefilter order
k	matrix index
$\mathbf{W}_{\text{PEM}}$	near-end excit. signal variance diag. matrix
$\sigma_w(t)$	near-end excitation signal variance
$\hat{\sigma}_w(t)$	estimated near-end excitation signal variance
$\mathbf{U}_a$	input DFT vector
$\mathbf{E}_a$	prefiltered error DFT vector
$z(t)$	IPC measure signal
$\mathbf{r}(k)$	concatenated error signal vectors
$\mathbf{G}$	normalization factor
S_{U_a}	input PSD estimate
S_{E_a}	prefilter error PSD estimate
$\mathbf{\Theta}$	DFT grad. estimate
P	window size to calculate AR coefficients
K	APA order

N	filter length
M	DFT size
$e_o(t)$	'ideal' error signal
$\tilde{e}(t)$	error signal due to DT
α	regularization parameter
α_{VR}	variable regularization
μ_{PVSS}	VSS in PVSS
μ_{PCVSS}	VSS in PCVSS
μ_0	fixed (maximum) step size
μ'	VSS aux. variable in PCVSS
λ_0	PSD forgetting factor
β_0	Corr. forgetting factor
$\mathbf{C}$	correlation vector in PCVSS

Chapter 6

$\hat{\mathbf{f}}(t), \hat{F}(q,t)$	estimated echo path
$u(t)$	input/loudspeaker signal
$\hat{x}(t)$	estimated echo signal
$x(t)$	echo signal
$v(t)$	near-end signal
$e(t)$	error signal
$y_a[t, \hat{h}(t)]$	prefilter microphone signal using $\hat{h}(t)$
$u_a[t, \hat{h}(t)]$	prefilter loudspeaker signal using $\hat{h}(t)$
$e_a[t, \hat{h}(t), \hat{f}(t-1)]$	prefilter error signal using $\hat{h}(t)$
$e[t, \hat{f}(t)]$	error signal calculated with $\hat{f}(t)$
$n(t)$	near-end noise signal
$w(t)$	near-end model excitation signal
$H(q,t)$	near-end model
$A(q,t)$	inverse near-end auto-regressive model
n_A	order near-end model
$\boldsymbol{\vartheta}(t)$	model system
$\hat{\boldsymbol{\vartheta}}(t)$	estimated model parameters
$\mathbf{f}(t)$	acoustic path model parameters
$\mathbf{a}(t)$	near-end signal model system
$\hat{\mathbf{a}}(t)$	near-end signal estimated coefficients
n_F	order acoustic path model
$n_{\hat{F}}$	order acoustic path model estimates
$\hat{\sigma}_w^2(t)$	estimate of the near-end signal excitation variance
$\mathbf{v}(t)$	time-domain (TD) near-end signal vector
$\mathbf{u}(t)$	TD input signal vector
$\mathbf{n}(t)$	TD near-end noise signal vector
$\mathbf{y}(t)$	TD microphone signal vector

N	estimated RIR length
M	size of FFT or FD filter length
k	block-time index
L	total length of the signals
P	block length to estimate $A(q,t)$
$E_a(m,k)$	scalar frequency-domain prefiltered error signal
m	time/frequency index in length-M vector
K	forward path gain
$\mathbf{E}_a(m,k)$	length-M FD prefiltered error signal
$\mathbf{e}_a(k)$	length-M TD error signal after prefiltering
$\mathbf{v}_a(k)$	length-M TD prefiltered near-end signal
$\mathbf{n}_a(k)$	length-M TD prefiltered near-end noise signal
$\mathbf{y}_a(k)$	length-M TD prefiltered microphone signal
$\mathbf{U}(k)$	length-M FD input signal
$\hat{\mathbf{F}}(k)$	length-M FD adaptive filter coeff.
$\boldsymbol{\Theta}(k)$	GRASS
$\theta(m,k)$	length-M FD (noisy) gradient estimate
$\theta_0(k)$	length-M FD true gradient
$\theta(k)$	length-M FD gradient noise
$\boldsymbol{\phi}(k)$	TD aux. gradient variable
$\boldsymbol{\mu}(k)$	length-M normalization factor in FDAF
$\mathbf{W}(k)$	WISE coefficients
$\mu_{\max}$	maximum allowed step-size
$ENR(t)$	echo noise ratio variable
$\sigma_n^2(t)$	near-end noise signal variance
$\sigma_x^2(t)$	echo signal variance
$\sigma_v^2(t)$	near-end speech signal variance
$Y(m,k)$	FD microphone signal
$X(m,k)$	FD echo signal
$V(m,k)$	FD near-end speech signal
$D(m,k)$	FD disturbance signal
$W_0(m,k)$	FD noise-reduction Wiener filter
$P_{XY}(m,k)$	CPSD of $X(m,k)$ and Y(m,k)
$P_Y(m,k)$	power spectral density of and $Y(m,k)$
$P_X(m,k)$	power spectral density of and $X(m,k)$
$P_D(m,k)$	power spectral density of and $D(m,k)$
$\nabla(m,k)$	recursive gradient power estimate
$\alpha(m,k)$	gradient estimate phase
$\hat{P}_{U_a}(m,k)$	recursive PSD estimate of and $U_a(m,k)$
$\hat{P}_{X_a}(m,k)$	recursive PSD estimate of and $X_a(m,k)$
$\hat{P}_{D_a}(m,k)$	recursive PSD estimate of and $D_a(m,k)$
$\hat{ENR}(m,k)$	estimated frequency-domain ENR
$\hat{P}_{\hat{\theta}_0}(m,k)$	estimate of the true gradient PSD
λ_1	forgetting factor WISE: desired signal

λ_2	forgetting factor WISE: noise signal
λ_3	forgetting factor GRASS
Q	APA order
$\bar{\mathbf{n}}$	one realization of a WGN process
δ	small number to avoid division by zero
$\mathbf{f}_1$	original RIR coefficients
$\mathbf{f}_2$	synthesized RIR coefficients
$\mathbf{\Phi}(k)$	WISE-GRASS time-domain gradient constraint

Chapter 7

$u(t)$	input signal
$f[u(t)]$	nonlinear function of the input signal
$y(t)$	echo signal: output of a nonlinear system
$\hat{y}(t)$	estimated echo signal
$d(t)$	microphone signal
$n(t)$	noise signal
$e(t)$	error signal
$H(q,t)$	linear acoustic echo path model
$\hat{H}(q,t)$	estimated echo path model
b_k	time-domain deterministic amplitude of the kth harmonic
a_k	frequency-domain deterministic amplitude of the kth harmonic
F	number of harmonics
ω_0	fundamental radial frequency
f_0	fundamental frequency
k	harmonic index
i_k	harmonic values
ϕ_k	realization of independent uniformly distributed random processes on [0,2π]
$U(j\omega_k)$	frequency-domain input signal of a nonlinear system
$Y(j\omega_k)$	frequency-domain output signal of a nonlinear system
$G(j\omega_k)$	measured FRF of nonlinear system $G(jw_k) = \dfrac{Y(jw_k)}{U(jw_k)}$
$G_0(j\omega_k)$	true underlying linear system
$G_B(j\omega_k)$	systematic nonlinear contribution
$G_S(j\omega_k)$	stochastic nonlinear contribution
$G_{\mathrm{BLA}}(j\omega_k)$	best linear approximation (BLA) of the nonlinear system

$\hat{G}_{\text{BLA}}(j\omega_k)$	estimated BLA
$\hat{G}^{[q]}(j\omega_k)$	FRF data averaged over P periods
$\hat{\sigma}^2_{\hat{G}^{[q]}}(jw_k)$	sample variance averaged over P periods
$\hat{\sigma}^2_{\hat{G}_{\text{BLA}}}(jw_k)$	total variance
$\hat{\sigma}^2_{\hat{G}_{\text{BLA}},n}(jw_k)$	noise variance
$var\left[G_S(jw_k)\right]$	stochastic nonlinear contribution variance
P	number of periods
Q	number of phase realizations
$\hat{Y}(j\omega_k)$	averaged output spectrum
K	harmonic grid parameter
y_f	linear-in-the-parameters nonlinear filter output
$\mathbf{h}_F$	linear-in-the-parameters nonlinear filter model
$\mathbf{u}_F$	linear-in-the-parameters nonlinear filter input
$\mathbf{h}_r$	linear-in-the-parameters nonlinear filter model sub-vector
$\mathbf{u}_r$	linear-in-the-parameters nonlinear filter input sub-vector
$\hat{\mathbf{h}}_F$	estimated linear-in-the-parameters nonlinear filter model coefficients
L	nonlinear filter length
M	order of nonlinearity
ϵ	small regularization term
$h(n)$	linear filter coefficients
g_m	amplitude of nonlinear terms Hammerstein and Legendre
$L_m\left[u(t)\right]$	mth-order Legendre nonlinear expansion
L_{H}	Hammerstein filter total length
L_{L}	Legendre polynomial filter total length
L_{SV}	simplified Volterra filter total length
$h_1(n_1)$	linear filter coefficients in Volterra filter
$h_m(n_1, ..., n_m)$	mth Volterra kernel coefficients
L_{V}	Volterra filter total length
V_{2K}	simplified 2nd-order Volterra kernel
V_{3K}	simplified 3rd-order Volterra kernel
N_d	simplified Volterra kernel size parameters

Chapter 8

$< \cdot, \cdot >$	inner product
P	APA projection order

$\|\cdot\|_2$	2-norm of a vector
L	Input dimension in taps
F	dimension in the feature space
$\mathbf{u}(i)$	input (loudspeaker) signal vector
$\mathbf{U}(i)$	input (loudspeaker) signal matrix
$d(i)$	desired (microphone) signal
$\mathbf{d}(i)$	desired signal vector
$\mathbf{h}$	room impulse response (true filter weights)
$\hat{\mathbf{h}}(i)$	weight vector estimate in an Euclidean space at iteration i
D	storage memory (dictionary size)
$\mathbf{a}(i)$	expansion coefficients vector
$\mathbf{X}(i)$	set of input vectors (dictionary)
$\boldsymbol{\varphi}(\cdot)$	a mapping induced by a reproducing kernel
$\boldsymbol{\varphi}(i)$	transformed input (vector in a feature space)
$\boldsymbol{\Phi}(i)$	transformed input APA matrix
$\kappa(i,j)$	kernel evaluation
$\mathbf{G}(i)$	Gram matrix
$\mathbb{F}$	feature space induced by the kernel mapping
$\boldsymbol{\omega}(i)$	filter weights estimate in a feature space at iteration i
κ_Σ	weighted sum of kernels
κ_L	linear kernel
κ_G	Gaussian kernel
α	weight of κ_L in κ_Σ
β	weight of κ_G in κ_Σ
μ	step-size KAPA
λ	regularization factor
δ	forgetting factor LS
$\mathbf{R}_{uu}$	sample covariance matrix
$\mathbf{r}_{ud}$	sample cross-covariance vector
$\hat{\mathbf{R}}_{uu}$	approximate covariance matrix
$\hat{\mathbf{r}}_{ud}$	approximate cross-covariance vector
$\mathbf{z}$	auxiliary vector

Contents

IV Nonlinear Adaptive Filters

V Conclusions

Part I

Introduction

Chapter 1

Introduction and Overview

THIS chapter introduces the fundamental principles of adaptive filtering. The commonly used adaptive filter structures and algorithms, as well as practical applications employing adaptive filters are described. All problems and difficulties encountered in time-domain adaptive filters are extensively discussed. In this thesis, only discrete-time implementations of adaptive filters are taken into account. It is, therefore, assumed that continuous-time signals, taken from the real world, are properly sampled, i.e., at least at twice their highest frequency, so that the Nyquist or sampling theorem is satisfied [1], [2], [4], [5], [19]. The reader can find extensive discussions on general adaptive signal processing in many reference textbooks [4], [5], [13], [16]. Nowadays, adaptive filtering finds practical applications in several fields such as communications, biomedical engineering, control, radar, sonar, navigation, seismology, etc. In this chapter, we consider two of the state-of-the-art applications where adaptive filters are widely used, namely acoustic echo cancellation and acoustic feedback control applications. The main characteristics of these applications are outlined in terms of issues that a typical adaptive filter implementation would encounter. This chapter also gives a brief introduction to frequency-domain adaptive filters that are computationally more efficient for applications that need longer filter lengths, and are more effective when there is a high correlation in the input signal. For instance, to cancel the acoustic echo in hands-free telephony, high-order adaptive filters are needed. However, high-order adaptive filters, together with a highly correlated input signal, weakens the performance of most time-domain adaptive filters [6], [14], [16], [18].

1.1 Fundamentals of adaptive filtering

A filter, in general, may be seen as a system that extracts or enhances desired information contained in a signal. If we want to process information in an unknown and changing environment, an adaptive filter is needed [13]. Typically, an adaptive filter has an associated adaptive algorithm for updating filter coefficients.

An adaptive algorithm adjusts the filter coefficients (or tap weights) in relation to the signal conditions and performance criterion (or quality assessment). A typical performance criterion is based on the error signal $e(n)$, which is the difference between the filter output signal and the reference (or desired) signal [5], [16].

1.1.1 Objective function

The complexity and performance of the adaptive algorithm is affected by the definition of the objective function. We may list the following forms of objective functions which are widely used in the derivation of adaptive algorithms:

- Mean Squared Error (MSE): $J[e(n)] = \mathcal{E}\{e^2(n)\}$.
- Mean Absolute Error (MAE): $J[e(n)] = \mathcal{E}\{|e(n)|\}$.
- Sum of Squared Errors (SSE): $J[e(n)] = \sum_{i=1}^{n} e^2(i)$.
- Weighted Sum Squared Error (WSSE): $J[e(n)] = \sum_{i=1}^{n} \lambda^{n-i} e^2(i)$, with $0 << \lambda < 1$.
- Instantaneous Squared Error (ISE): $J[e(n)] = e^2(n)$.

where $\mathcal{E}\{\cdot\}$ is the expectation operator. In a strict sense, the MSE function is only of theoretical value, due to the infinite amount of information that is required to be measured. This ideal objective function, in practice, can be approximated by the SSE, WSSE, or ISE functions. These functions lead to filters that differ in the computational complexity and in the convergence characteristics. Generally, the ISE function is cheaper to implement but it exhibits noisy convergence properties [5], [16]. The SSE function is favorable to be used in stationary environments, whereas in slowly varying environments, the WSSE is much better suited.

1.1.2 Adaptive transversal filters

An adaptive filter is a self-designing and time-varying system, which employs a recursive algorithm, in order to adjust its tap weights in an unknown environment. Figure 1.1 illustrates a typical structure of an adaptive filter, consisting of two basic blocks: (1) a digital filter to perform the desired filtering and (2) an adaptive algorithm to adjust the tap weights of the filter [5], [16]. An adap-

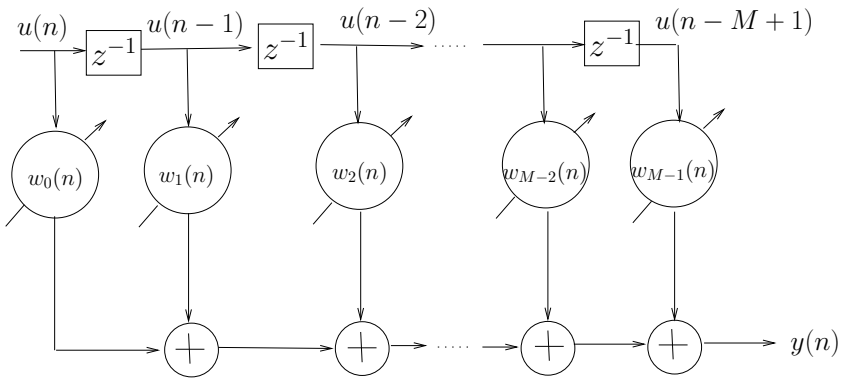


Figure 1.1: Transversal FIR adaptive filter.

tive filter computes the output signal $y(n)$ in response to the input signal $u(n)$. The error signal $e(n)$ is then generated by comparing $y(n)$ with the desired response $d(n)$. An adaptive algorithm adjusts the tap weights based on the error signal $e(n)$ (also called the performance feedback signal). In this section, we only consider real-valued signals and real-valued tap weights. Many different structures can be used to realize the digital filter, e.g., lattice, infinite impulse response (IIR), finite impulse response (FIR). Figure 1.1 shows the commonly used transversal FIR filter.

The adjustable tap weights, $w_m(n)$, $m = 0, 1, ..., M-1$ (indicated by circles with arrows through them) are the filter tap weights at time n and M is the filter length. These time-varying tap weights form an $M \times 1$ weight vector expressed as

$$\mathbf{w}(n) = [w_0(n), w_1(n), ..., w_{M-1}(n)]^T \tag{1.1}$$

where the superscript $(\cdot)^T$ denotes the transpose operation. Similarly, the input signal samples, $u(n-m)$, $m = 0, 1, ..., M-1$, form an $M \times 1$ input vector

$$\mathbf{u}(n) = [u(n), u(n-1), ..., u(n-M+1)]^T. \tag{1.2}$$

The output signal $y(n)$ of the adaptive FIR filter, with these vectors, can be computed as the inner product of $\mathbf{w}(n)$ and $\mathbf{u}(n)$ given as

$$y(n) = \sum_{m=0}^{M-1} w_m(n)u(n-m) = \mathbf{w}^T(n)\mathbf{u}(n). \tag{1.3}$$

1.1.3 Minimum MSE

The difference between the desired signal $d(n)$ and the filter output signal $y(n)$, expressed as

$$e(n) = d(n) - \mathbf{w}^T(n)\mathbf{u}(n) \tag{1.4}$$

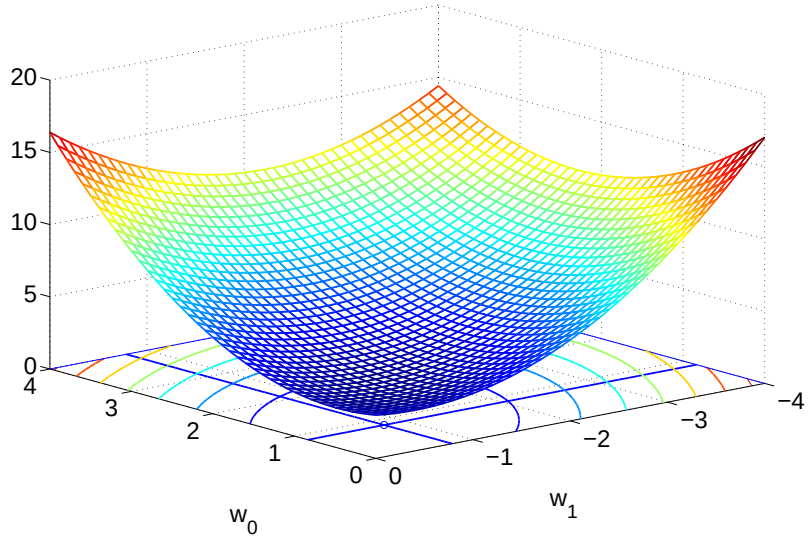


Figure 1.2: Error surface.

is the error signal $e(n)$. The weight vector $\mathbf{w}(n)$ is updated recursively such that the error signal $e(n)$ is minimized. The minimization of the MSE function is regularly used as a performance criterion (or cost function), which is given as

$$J_{\text{MSE}} = \mathcal{E}\{e^2(n)\} \tag{1.5}$$

For a given weight vector $\mathbf{w} = [w_0, w_1, ..., w_{M-1}]^T$ with stationary input signal $u(n)$ and desired response $d(n)$, the MSE can be calculated from (1.4) and (1.5) as

$$J_{\text{MSE}} = \mathcal{E}\{d^2(n)\} - 2\mathbf{p}^T\mathbf{w} + \mathbf{w}^T\mathbf{R}\mathbf{w}, \tag{1.6}$$

where $\mathbf{R} \equiv \mathcal{E}\{\mathbf{u}(n)\mathbf{u}^T(n)\}$ is the input autocorrelation matrix and $\mathbf{p} \equiv \mathcal{E}\{d(n)\mathbf{u}(n)\}$ is the cross-correlation vector between the desired signal and the input vector. The MSE is treated as a stationary function, hence the time index n has been dropped from the vector $\mathbf{w}(n)$ in (1.6). The MSE in (1.6) is a quadratic function of the tap weights $[w_0, w_1, ..., w_{M-1}]$ since they only appear in first and second degrees. Figure 1.2 shows a typical performance (or error) surface for a two-tap transversal filter. A single global minimum MSE corresponding to the optimum vector $\mathbf{w}_o$ is therefore guaranteed to exist in a quadratic performance surface when considering a transversal FIR filter. We can obtain the optimum solution by taking the first derivative of (1.6) with respect to $\mathbf{w}$ and setting the derivative to zero. This leads to the well-known Wiener-Hopf equations [5]

$$\mathbf{R}\mathbf{w}_o = \mathbf{p}. \tag{1.7}$$

If we assume $\mathbf{R}$ is invertible, the optimum weight vector is given as

$$\mathbf{w}_o = \mathbf{R}^{-1}\mathbf{p}. \tag{1.8}$$

By substituting (1.8) into (1.6), the minimum MSE corresponding to the optimum weight vector is obtained as

$$J_{\min} = \mathcal{E}\{d^2(n)\} - \mathbf{p}^T\mathbf{w}_o. \tag{1.9}$$

1.2 Adaptive algorithms

An adaptive algorithm is a set of recursive equations that automatically adjust the weight vector $\mathbf{w}(n)$ in order to minimize the objective function. Ideally, the weight vector converges to the optimum solution $\mathbf{w}_o$ corresponding to the bottom of the performance surface, i.e. the minimum MSE $J_{\min}$.

1.2.1 (Normalized) Least mean squares algorithm

The most widely used adaptive algorithm is the least mean squares (LMS) algorithm [19] because of its robustness and simplicity [4]. Based on the steepest-descent method, using the negative gradient of the ISE function, i.e. $J \approx e^2(n)$, the LMS algorithm weight update equation is written as

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu\mathbf{u}(n)e(n) \tag{1.10}$$

with μ the convergence factor (or step size), that determines the stability and the convergence rate of the algorithm.

The LMS algorithm uses, as shown in (1.10), a recursive approach for adjusting the tap weights in the direction of the optimum Wiener-Hopf solution given in (1.7). The step size is chosen in the range

$$0 < \mu < \frac{2}{\lambda_{\max}} \tag{1.11}$$

to guarantee the stability of the algorithm, where $\lambda_{\max}$ is the largest eigenvalue of the input autocorrelation matrix $\mathbf{R}$. The sum of the eigenvalues (i.e., the trace of $\mathbf{R}$) is used instead of $\lambda_{\max}$. Accordingly, the step size is in the range of $0 < \mu < \dfrac{2}{\text{trace}(\mathbf{R})}$. Taking into account that $\text{trace}(\mathbf{R}) = MP_u$ is related to the average power P_u of the input signal $u(n)$, a common step size bound [5], [16] is obtained as

$$0 < \mu < \frac{2}{MP_u}. \tag{1.12}$$

Convergence of the MSE for Gaussian input signals typically requires $0 < \mu < 2/3MP_u$ [5]. Moreover, as shown in (1.12), for a larger filter length M a smaller

step size μ is used to prevent instability. We may also highlight that the step size is inversely proportional to the input signal power P_u. Consequently, a signal with high P_u must use a smaller step size, while a low-power signal can use a larger step size. We may incorporate this relationship into the LMS algorithm just by normalizing the step size with respect to the input signal power. This type of normalization of the step size leads to a useful and widely used variant of the LMS algorithm, the well-known normalized LMS (NLMS) algorithm [5].

The NLMS algorithm includes an additional normalization term $\mathbf{u}^T(n)\mathbf{u}(n)$ as

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \frac{\mathbf{u}(n)}{\mathbf{u}^T(n)\mathbf{u}(n) + \epsilon} e(n), \tag{1.13}$$

where the step size is now bounded in the range $0 < \mu < 2$ and ϵ is a small regularization term to prevent division by zero. It is worth noting that the NLMS algorithm may also be derived as the solution to a constrained optimization problem formulating the *principle of minimum disturbance* [5], or as a member of the underdetermined-order recursive least squares (URLS) family [28]. When normalizing the input vector $\mathbf{u}(n)$ the convergence rate becomes independent of the signal power. There is no significant difference between the convergence performance of the LMS and NLMS algorithms for stationary signals provided that the step size of the LMS algorithm is properly chosen. The benefit of the NLMS algorithm only becomes clear for nonstationary signals like speech, where significantly faster convergence for the same level of steady-state MSE can be achieved [4], [13], [16].

Assuming all the signals and tap weights are real-valued, the transversal FIR filter requires M multiplications to produce the output $y(n)$ and the update equation (1.10) requires $(M+1)$ multiplications. Therefore, the adaptive FIR filter with the LMS algorithm requires a total of $(2M+1)$ multiplications per iteration. On the other hand, the NLMS algorithm requires an additional $(M+1)$ multiplications for the normalization term, giving a total of $(3M+2)$ multiplications per iteration. The computational complexity of the LMS and NLMS algorithms is hence proportional to M, which is expressed as $\mathrm{O}(M)$.

1.2.2 Recursive least squares algorithm

Contrary to the (N)LMS algorithm, the recursive least squares (RLS) algorithm [5] is derived from the minimization of the WSSE

$$J_{\mathrm{LS}}(n) = \sum_{i=1}^{n} \lambda^{n-i} e^2(i), \tag{1.14}$$

where $0 << \lambda < 1$ is the forgetting factor. Indeed, in nonstationary environments, the forgetting factor weights the current error more than the past

error values. In this sense, the LS weight vector $\mathbf{w}(n)$ is optimized based on the observation starting from the first iteration ($i = 1$) to the current iteration ($i = n$). It should be noted that the LS solution can be expressed as a special case of the Wiener-Hopf solution defined in (1.7), i.e,

$$\mathbf{w} = \mathbf{R}^{-1}(n)\mathbf{p}(n) \tag{1.15}$$

where the autocorrelation matrix and cross-correlation vector are defined respectively as

$$\mathbf{R} \approx \mathbf{R}(n) = \sum_{i=1}^{n} \lambda^{n-i}\mathbf{u}(i)\mathbf{u}^T(i) \tag{1.16}$$

and

$$\mathbf{p} \approx \mathbf{p}(n) = \sum_{i=1}^{n} \lambda^{n-i}d(i)\mathbf{u}(i). \tag{1.17}$$

By using the matrix inversion lemma, the RLS algorithm can be written as

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mathbf{g}(n)e(n) \tag{1.18}$$

where the updating gain vector is defined as

$$\mathbf{g}(n) = \frac{\mathbf{r}(n)}{1 + \mathbf{u}^T(n)\mathbf{r}(n)} \tag{1.19}$$

and

$$\mathbf{r}(n) = \lambda^{-1}\mathbf{P}(n-1)\mathbf{u}(n). \tag{1.20}$$

The inverse input autocorrelation matrix, $\mathbf{P}(n) \equiv \mathbf{R}^{-1}(n)$, can also be computed recursively as

$$\mathbf{P}(n) = \lambda^{-1}\mathbf{P}(n-1) - \mathbf{g}(n)\mathbf{r}^T(n). \tag{1.21}$$

Due to the fact that both the NLMS and RLS algorithms converge to the same optimum weight vector (under stationarity and ergodicity conditions), there is a strong link between them [7]. The computational complexity of the RLS algorithm is $\mathrm{O}(M^2)$, hence it is more expensive to implement than NLMS. On the other hand, the RLS algorithm typically converges much faster than NLMS. There are diverse efficient versions of the RLS algorithm, including the fast transversal filter with reduced complexity $\mathrm{O}(M)$, but unfortunately these fast algorithms suffer from instability issues [16].

1.2.3 Spectral dynamic range and misadjustment

The convergence behavior of the LMS algorithm is associated with the eigenvalue spread of the autocorrelation matrix $\mathbf{R}$ which is defined by the characteristics of the input signal $u(n)$. The eigenvalue spread is measured by the

condition number defined as $\kappa(\mathbf{R}) = \lambda_{\max}/\lambda_{\min}$, where $\lambda_{\max}$ and $\lambda_{\min}$ are the maximum and minimum eigenvalues, respectively. In addition, the condition number depends on the spectral distribution of the input signal [13] and is bounded by the spectral dynamic range of the input power spectrum $\mathbf{P}_{uu}\left(e^{j\omega}\right)$ as

$$\kappa(\mathbf{R}) \leq \frac{\max_{\omega} P_{uu}\left(e^{j\omega}\right)}{\min_{\omega} P_{uu}\left(e^{\omega}\right)} \tag{1.22}$$

where $\max_{\omega} P_{uu}\left(e^{j\omega}\right)$ and $\min_{\omega} P_{uu}\left(e^{j\omega}\right)$ are the maximum and minimum values of the input power spectrum, respectively. For white input signals, the ideal condition number $\kappa(\mathbf{R}) = 1$ is obtained. The convergence speed of the LMS algorithm decreases for increasing spectral dynamic range [4]. Frequency-domain and transform-domain adaptive filters will be introduced in Section 1.3 as to improve the convergence rate when the input signal has a high spectral dynamic range (i.e., is highly correlated).

A time constant that defines the convergence rate of the MSE [4] may be written as

$$\tau_m = \frac{1}{2\mu\lambda_m}, \quad m = 0, 1, ..., M-1. \tag{1.23}$$

where λ_m is the mth eigenvalue. Therefore, the smallest eigenvalue $\lambda_{\min}$ determines the slowest convergence. Despite the fact that a large step size can result in faster convergence, it must be upper bounded by (1.10).

In practice, due to the use of the instantaneous squared error by the LMS algorithm, the weight vector $\mathbf{w}(n)$ deviates from the optimum weight vector $\mathbf{w}_o$ in the steady state. As a result, after the algorithm has converged, the MSE in the steady state is greater than the minimum MSE $J_{\min}$. The difference between the steady-state MSE $J(\infty)$ and $J_{\min}$ is called the excess MSE, and it is expressed as $J_{ex} = J(\infty) - J_{\min}$. We may also define the misadjustment as

$$\mathcal{M} = \frac{J_{ex}}{J_{\min}} = \frac{\mu}{2} M P_u \tag{1.24}$$

The misadjustment is proportional to the step size, which in turn translates in a tradeoff between the misadjustment and the convergence rate given in (1.23). The misadjustment is normally defined as a percentage and has a typical value of 10% for most applications. A small step size results in a slow convergence, but has the advantage of smaller misadjustment, and vice versa. Under the assumption that all the eigenvalues are equal, (1.23) and (1.24) are related by the following simple expression

$$\mathcal{M} = \frac{M}{4\tau_m} \tag{1.25}$$

Consequently, for achieving a small value of misadjustment, a long filter requires a long time τ_m.

The transversal FIR filter shown in Figure 1.1 is the most commonly used adaptive structure. This structure operates in the time domain and can be implemented in either the sample or the block processing mode [15], [16]. However, the time-domain LMS-type adaptive algorithms, associated with the FIR filter, suffers from high computational cost, when applications (such as acoustic echo cancellation) demand a high-order filter and slow convergence when the input signal is highly correlated.

1.3 Frequency-domain and transform-domain adaptive filters

In this section, we describe the frequency-domain and transform-domain adaptive filters [6], [8], [12], [16]. The advantages of these adaptive filters are fast convergence and low computational complexity. We highlight as well the differences between frequency-domain and transform-domain adaptive filters and the relation between these filters.

1.3.1 Frequency-domain adaptive filters

In a frequency-domain adaptive filter (FDAF) [8], [12] the desired signal $d(n)$ and the input signal $u(n)$ are transformed to the discrete frequency domain using the discrete Fourier transform (DFT). FDAF performs frequency-domain filtering and adaptation based on these transformed signals. As discussed in Sections 1.1 and 1.1.2, the time-domain adaptive filter performs a linear convolution (filtering) and linear correlation (weight updating) on a sample-by-sample basis. The FDAF performs a convolution and correlation on a block-by-block basis. In this way, substantial computational savings, especially for applications that use high-order filters can be achieved. Unfortunately, the frequency-domain operations result in circular convolution and circular correlation. Therefore, to overcome the error introduced by these circular operations, more complicated overlap-add or overlap-save methods [8], [12] are typically implemented. These methods allow to obtain the correct linear convolution and correlation results.

The block diagram of the FDAF using the fast LMS algorithm [8], [12] is shown in Figure 1.3. This algorithm uses the overlap-save method with 50% overlap and needs five $2M$-point DFT/IDFT operations. This type of processing is known as block processing, where the input vector is

$$\mathbf{u}(n)_{2M} = [u(n-2M+1), ..., u(n-M), u(n-M+1), ..., u(n)]^T, \quad (1.26)$$

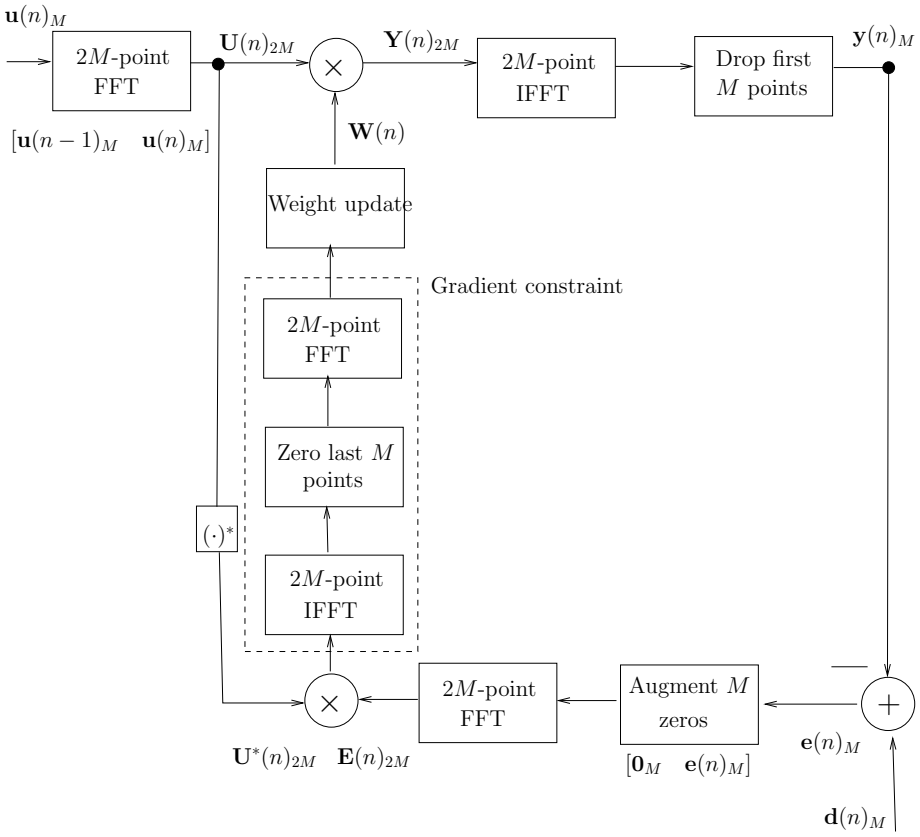


Figure 1.3: Typical frequency-domain adaptive filter block scheme using the overlap-save method.

the DFT of the input vector is

$$\mathbf{U}(n)_{2M} = \mathcal{F}\{\mathbf{u}(n)_{2M}\} = [U_{2M-1}(n), ..., U_M(n), U_{M-1}(n), ..., U_0(n)]^T, \tag{1.27}$$

the error vector is

$$\mathbf{e}(n)_M = [e(n), e(n-1), ..., e(n-M+1)]^T, \tag{1.28}$$

and the DFT of the error vector as

$$\mathbf{E}(n)_{2M} = \mathcal{F}\{[\mathbf{0}_M \quad \mathbf{e}(n)_M]\} = [E_0(n), ..., E_{2M-1}(n)]^T. \tag{1.29}$$

The subscripts $2M$ and M denote the length of the vectors, e.g., $\mathbf{u}(n)_{2M}$ and $\mathbf{e}(n)_M$. Note that the input vector $\mathbf{u}(n)_{2M}$ concatenates the previous M input samples with the current M input samples, and then, a $2M$-point DFT operates

on this input vector resulting in the frequency-domain vector $\mathbf{U}(n)_{2M}$ (i.e., a $2M$-point complex vector). The subscripts in the elements of a vector, e.g., $E_0(n)$ and $E_{2M-1}(n)$, denote the entry number of the vector at time n. We can compute the output signal vector from the adaptive filter by taking the element-wise multiplication between the input vector and the $2M$-point weight vector as

$$\mathbf{Y}(n) = \mathbf{W}(n) \circ \mathbf{U}(n), \tag{1.30}$$

where '$\circ$' is the element-wise multiplication operator and

$$\mathbf{W}(n) = [W_0(n), W_1(n), ..., W_{2M-1}(n)]^T \tag{1.31}$$

is the complex-valued weight vector at time n.

In order to obtain the correct result using circular convolution, the frequency-domain output vector, $\mathbf{Y}(n) = [Y_0(n), Y_1(n), ..., Y_{2M-1}(n)]^T$, is transformed back to the time domain using the inverse DFT (IDFT) and the first M points of the $2M$-point IDFT outputs are discarded to obtain the output vector $\mathbf{y}(n)_M = [y(n), y(n-1), ..., y(n-M+1)]^T$. The output vector is subtracted from the desired vector $d(n) = [d(n), d(n-1), ..., d(n-M+1)]^T$ to produce the error signal vector $\mathbf{e}(n)_M$. As shown in Figure 1.3, the error vector $\mathbf{e}(n)$ is then augmented with M zeros and transformed to the frequency-domain vector $\mathbf{E}(n)$ using the DFT.

We can express the complex weight update equation as

$$W_k(n+M) = W_k(n) + \mu\nabla[U_k^*(n)E_k(n)], \quad k = 0, 1, ..., 2M-1, \tag{1.32}$$

where $U_k^*(n)$ is the complex conjugate of the frequency-domain signal $U_k(n)$, and $\nabla[\cdot]$ represents the operation of gradient constraint, which is explained as follows. As shown in the 'gradient constraint box' in Figure 1.3, the weight-updating term $[U_k^*(n)E_k(n)]$ is inverse transformed, and the last M points are set to zero before taking the $2M$-point DFT for the weight update equation. The so-called unconstrained FDAF [8] is an alternative where the gradient constraint block is removed, thus producing a simpler implementation that involves only three DFT operations [8], [12]. Unfortunately, this simplified algorithm no longer produces a linear correlation between the transformed error and input vectors, which results in poorer performance compared to the FDAF with gradient constraints. The weight vector is not updated sample by sample as in the time-domain LMS algorithm, but updated once for each block of M samples. The FDAF typically features an input power normalization factor, similar to the NLMS, to ensure (approximately) equal convergence rate to all frequency bins, hence the name FDAF-NLMS.

1.3.2 Transform-domain adaptive filters

As explained in Section 1.2, the eigenvalue spread of the input signal autocorrelation matrix plays an important role in determining the convergence speed

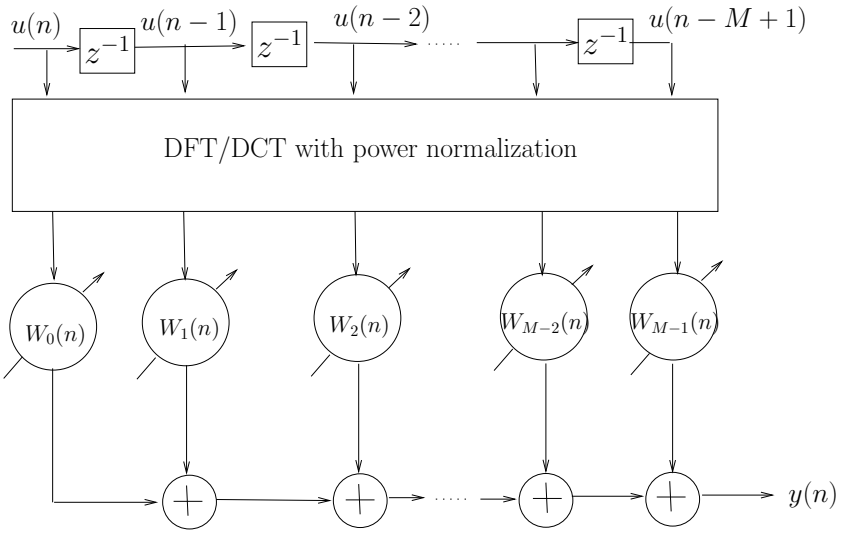


Figure 1.4: Transform-domain adaptive filter using DFT/DCT with power normalization.

of time-domain adaptive filters. Indeed, the convergence of the LMS algorithm is very slow when the input signal is highly correlated. In the literature, several methods have been developed to solve the problem of slow convergence due to highly correlated input signals. One method is to use the RLS algorithm (see Section 1.2). The RLS algorithm may be seen to extract past information for decorrelating the present input signal. This approach suffers from high computational cost, poor robustness, and slow tracking ability in nonstationary environments. Another method to improve convergence in the LMS algorithm is to decorrelate the input signal using a unitary transform such as the discrete Fourier transform (DFT) or discrete cosine transform (DCT) [6], [15], [16]. The transform-domain NLMS (TD-NLMS) is also known as the self-orthogonalizing adaptive filter. Compared with the RLS approach, the self-orthogonalizing adaptive filter saves computational cost since it is independent of the characteristics of the input signal.

The TD-NLMS adaptive filter, shown in Figure 1.4, has a similar structure to the $M-$tap adaptive transversal filter shown in Figure 1.1, but with preprocessing (transform and normalization) on the input signal using the DFT or DCT, and thus named DFT-NLMS and DCT-NLMS. The transformation is performed on a sample-by-sample basis. The transformed signals $U_k(n)$ are normalized by the square root of its respective power in the transform domain as

$$U'_k(n) = \frac{U_k(n)}{\sqrt{P_k(n) + \epsilon}}, \quad k = 0, 1, ..., M-1, \tag{1.33}$$

where ϵ is a small regularization term to prevent division by zero, and $P_k(n)$ is the input power that can be estimated recursively as

$$P_k(n) = (1-\lambda)P_k(n-1) + \lambda U_k^2(n) \tag{1.34}$$

Note that the power $P_k(n)$ is updated recursively for every new input sample and λ is a forgetting factor that is usually chosen as $1/M$.

1.4 Problem statement

In this section, we consider two important and timely applications in which adaptive filters have been widely used. The main characteristics of these applications are outlined in terms of issues that a typical adaptive filter implementation would encounter. From this perspective, we review the common assumptions applying to each case.

1.4.1 Acoustic Echo Cancellation

The typical set-up for an acoustic echo canceler is depicted in Figure 1.5. A far-end speech signal $u(n)$ is played back in an enclosure (i.e., the room) through a loudspeaker. In the room there is a microphone to record a near-end speech signal which is to be transmitted to the far-end side. An acoustic echo path between the loudspeaker and the microphone exists so that the microphone signal $y(n) = x(n)+v(n)+n(n)$ contains an undesired echo signal $x(n)$ plus the near-end speech signal $v(n)$, generating a so-called double-talk (DT) situation, and the near-end noise signal $n(n)$.

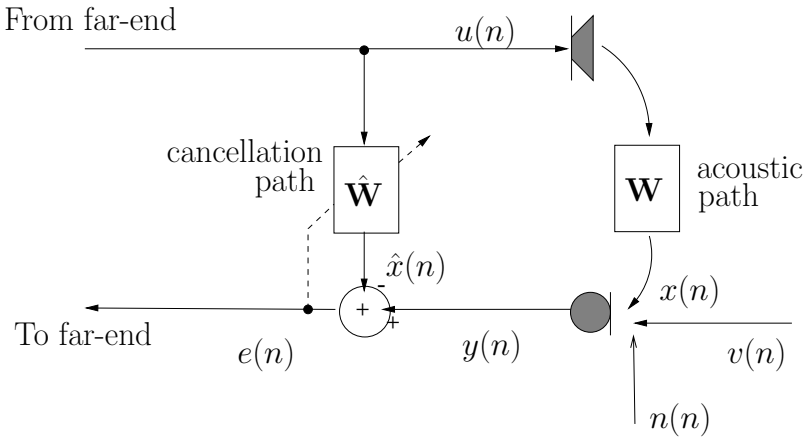


Figure 1.5: Typical set-up for AEC.

The echo signal $x(n)$ can be considered as the loudspeaker signal $u(n)$ filtered by the echo path. An acoustic echo canceler seeks to cancel the echo signal component $x(n)$ in the microphone signal $y(n)$, ideally leading to an *echo-free* error signal $e(n)$, which is then transmitted to the far-end side. This is done by subtracting an estimate of the echo signal $\hat{x}(n)$ from the microphone signal, i.e., $e(n) = y(n) - \hat{x}(n)$. Standard approaches to AEC rely on the assumption that the echo path can be modeled by a linear FIR filter [3], [14]. The coefficients of the echo path are collected in the parameter vector $\mathbf{w} = [w_0, w_1, , ..., w_{M-1}]^T \in \mathbb{R}^M$ such that $x(n) = \mathbf{w}^T\mathbf{u}(n)$ with $\mathbf{u}(n) = [u(n), u(n-1), \ldots, u(n-M+1)]^T$. An adaptive filter of sufficient order is used to provide an estimate $\hat{\mathbf{w}}(n) \in \mathbb{R}^M$ of $\mathbf{w}$, such that the echo signal estimate is $\hat{x}(n) = \hat{\mathbf{w}}^T(n)\mathbf{u}(n)$.

The AEC task [3], [14] can be seen as a system identification task, thus, the main objective of an echo canceler is to identify a model that represents a best fit to the echo path. Typically, 'best' in the LS sense is considered. Therefore, by (1.15) the echo path model estimate can be written as

$$\hat{\mathbf{w}}(n) = \mathbf{R}^{-1}(n)\mathbf{p}(n) \tag{1.35}$$

where

$$\mathbf{R}(n) = \sum_{i=1}^{n} \lambda^{n-i}\mathbf{u}(n)\mathbf{u}^T(n) \tag{1.36}$$

$$\begin{aligned}\mathbf{p}(n) &= \sum_{i=1}^{n} \lambda^{n-i} y(n)\mathbf{u}(n) \\ &= \sum_{i=1}^{n} \lambda^{n-i} x(n)\mathbf{u}(n) + \sum_{i=1}^{n} \lambda^{n-i}[v(n) + n(n)]\mathbf{u}(n)\end{aligned} \tag{1.37}$$

One particular assumption in most AEC applications, is that the noise signal $n(n)$ and the near-end signal $v(n)$ are uncorrelated with the loudspeaker signal $u(n)$. Consequently, the second term of the cross-correlation vector (1.37) tends to zero and, then, $\hat{\mathbf{w}}$ is an unbiased estimator minimizing the LS criterion.

There are many issues associated to practical AEC. In particular, long impulse responses make (time-domain) adaptive filters converge slowly and increase the overall computational complexity. The issue of regularization is also a matter of ongoing research [18]. Robustness to double-talk is obviously of vital importance since it occurs 20% of the time in a normal conversation [29]. A recent trend in consumer electronics is to utilize low-cost and small-sized analog components such as loudspeakers. These components usually have nonlinearities, so the hope is to rely on signal processing algorithms to mitigate the nonlinear effects. Nonlinearities can be roughly divided into two types: nonlinearities with and without memory. Nonlinearities with memory (or dynamic) usually occur in high-quality audio equipment where the time constant of the loudspeaker's electro-mechanical system is large compared to the sampling rate [21]. Memoryless (or static) nonlinearities typically occur in low-cost power amplifiers and

loudspeakers [24]. The level of the nonlinearities depends on the quality of the (active) loudspeaker, although eventually any (active) loudspeaker suffers from significant nonlinear distortion [17]. In AEC applications, nonlinear distortions are of great importance if the levels are high. As opposed to the situation when only background noise is present (i.e. where increasing the level of the loudspeaker signal will increase the SNR of the echo signal; hence improve the performance of the AEC), if loudspeaker nonlinearities are present, increasing the level of the input signal will also increase the level of the nonlinear echo component, which can be considered as an additional noise source. Therefore nonlinear adaptive filters are needed in AEC applications to account for nonlinear distortions. In the literature, second-order Volterra filters with memory are generally used [27]; however they can model only second-order nonlinearities. The use of third-order adaptive Volterra filters could become prohibitive in AEC applications [22], [23]. However it has been shown that some loudspeaker nonlinearities can be modeled as static (memoryless) functions of the voice coil position. Hence, the application of memoryless adaptive filters [24] featuring a block structure (i.e. static nonlinearities and linear dynamic blocks combined), would reduce the number of parameters to be estimated. All the above practical AEC issues will be tackled in the different chapters of this thesis.

1.4.2 Acoustic Feedback Control

Acoustic howling is well known to appear as a result of an acoustic feedback path, i.e., an acoustic coupling between a loudspeaker and a microphone. Due to this coupling, the signal from the loudspeaker is captured by the microphone, and then this signal is fed back to the loudspeaker after amplification. This phenomenon is also referred to as acoustic feedback. From a closed-loop system point of view, howling will occur if certain instability conditions are met. The analysis is based on the *Nyquist stability criterion* [25] which can be derived from the closed-loop frequency response of the system depicted in Figure 1.6, i.e

$$G_{\mathrm{CL}}(f) = \frac{W(f)}{1 - G(f)W(f)} \tag{1.38}$$

The second term in the denominator is called the loop response and is given as

$$G_{\mathrm{L}}(f) = G(f)W(f) \tag{1.39}$$

The feedback path response $W(f)$ is actually a combination of acoustic, mechanical and electromagnetic feedback, while $G(f)$ is a combination of A/D and D/A-converters, amplification and signal processing. The Nyquist stability criterion says that, if there exists a frequency f such that

$$\begin{cases} |W(f)G(f)| & \geq 1 \\ \angle W(f)G(f) & = n2\pi, \qquad n = ... - 2, -1, 0, 1, 2... \end{cases}$$

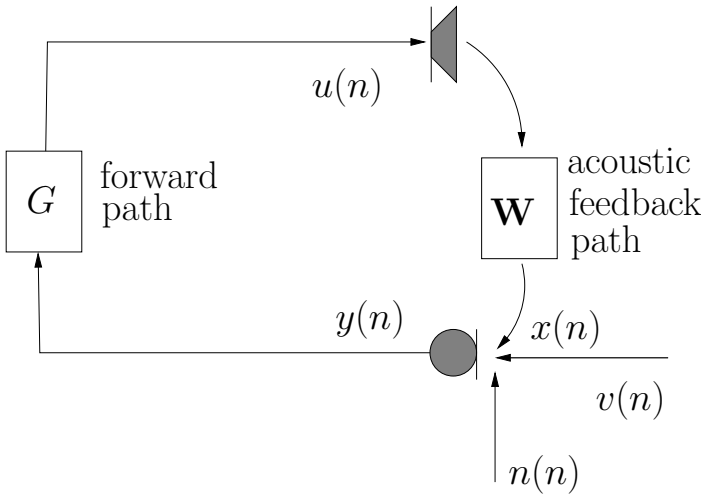


Figure 1.6: Closed-loop system resulting from acoustic feedback in a scenario with one loudspeaker and one microphone

then the closed-loop system is unstable. If the system is moreover excited with an input signal containing the critical frequency f, then an oscillation producing acoustic howling will occur. Howling is perceived as a very narrowband or sinusoidal signal at this critical frequency f.

Acoustic feedback limits the achievable amplification, which is critical in hearing aids (HA) and public address (PA) systems [9], [11]. It is noted that the two systems mentioned here (HA and PA) are quite different in nature. For instance, in HA systems usually one loudspeaker and one or two microphones are used, whereas in PA systems multichannel configurations are used. Yet, not only the number of channels but also the acoustic scenario inherent to these systems determines the preferred acoustic feedback control method. In HA systems, for instance, the feedback path impulse response is much shorter than in PA systems while, on the other hand, the computational power is much smaller than in PA systems. Therefore, it seems natural that different acoustic feedback control methods have been developed for these different systems. The state-of-the-art methods for acoustic feedback control in HA are based on acoustic feedback cancellation (AFC) [26], while for PA systems notch-filter-based howling suppression (NHS) methods are often preferred [11]. The typical set-up of an AFC is depicted in Figure 1.7. The AFC approach is similar to AEC in the sense that a model of the feedback path is estimated to predict the feedback signal component in the microphone signal. Like for AEC, long RIRs and nonlinearities are an issue for AFC. However, the main issue in AFC is the correlation that exists between the near-end speech component in the microphone signal and the loudspeaker signal itself. The correlation between

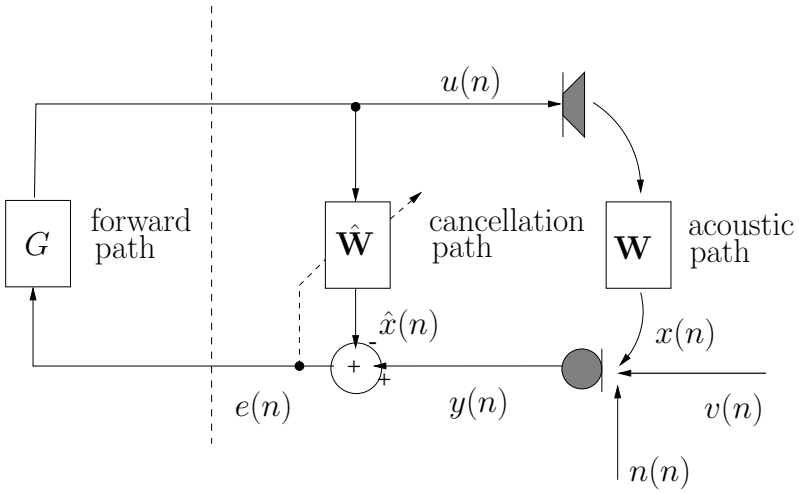


Figure 1.7: Continuous adaptation in AFC.

these two signals makes the second term in (1.37) non-zero. This correlation issue, which is caused by the closed loop, makes standard adaptive filtering algorithms to converge to a *biased* solution [9], [11]. This means that the adaptive filter does not only predict and cancel the feedback component in the microphone signal, but also part of the near-end speech. One approach to reduce the bias in the feedback path model identification is to prefilter the loudspeaker and microphone signal with the inverse near-end speech model, which is estimated jointly with the adaptive filter [9], [11] using the prediction error method (PEM) [10].

1.5 Outline of the thesis

A general overview of the thesis and its major contributions are now given:

Part II: Room Acoustic Modeling

Chapter 2. Estimation of acoustic resonances for room transfer function equalization: In this chapter, the utilization of different norms (i.e., 2-norm and 1-norm) and models (i.e., all-pole and pole-zero) for room transfer function (RTF) modeling and equalization is discussed. It is known that strong acoustic resonances create long room impulse responses (RIRs) which may harm the speech transmission in an acoustic space and, hence, reduce speech intelligibility. Equalization is, therefore, performed by compensating for the main acoustic resonances common to multiple room transfer functions

(RTFs) measured in the same room. These acoustic resonances may be modeled by means of the common poles of the RTFs. In the literature, however, there is no consensus about which model (i.e., all-pole or pole-zero) generates pole estimates that perform better in RTF equalization. Furthermore, MSE (i.e., 2-norm) minimization is typically employed for the estimation of the poles. In this chapter, an MAE (i.e., 1-norm) minimization is further proposed for pole estimation. A comparative evaluation of these different norms and models in terms of their residual RTF and residual RIR (i.e., the residuals after equalization) is provided.

Chapter 2 is based on:

- **Gil-Cacho J. M.**, van Waterschoot T., Moonen M. and Holdt Jensen S., "Estimation of acoustic resonances for room transfer function equalization", in Proc. of the International Workshop on Acoustic Echo and Noise Control (IWAENC'10), Tel Aviv, Israel, Aug. 2010.

Chapter 3. Multi-Microphone acoustic echo cancellation using warped multi-channel linear prediction of common acoustical poles: In this chapter, a computationally cheap extension from single-microphone AEC to multi-microphone AEC is presented for the case of a single loudspeaker. It employs the idea of common-acoustical-pole and zero RTF modeling. The RTF models used for multi-microphone AEC share a fixed common denominator polynomial, which is calculated off-line by means of a multi-channel warped linear prediction. By using the common denominator polynomial as a prefilter, only the numerator polynomial has to be estimated recursively for each microphone, hence adapting to changes in the room. This approach allows the number of numerator coefficients to decrease by one order of magnitude for each microphone compared with all-zero modeling. In a first configuration, the prefiltering is done on the adaptive filter signal, hence achieving a pole-zero model of the RTF in the AEC. In a second configuration, the prefiltering is done on the loudspeaker signal, hence achieving a dereverberation effect, in addition to AEC, on the microphone signals.

Chapter 3 is based on:

- **Gil-Cacho J. M.**, van Waterschoot T., Moonen M. and Holdt Jensen S., "Multi-Microphone acoustic echo cancellation using warped multi-channel linear prediction of common acoustical poles", in Proc. 18th European Signal Processing Conference 2010 (EUSIPCO'10), Aalborg, Denmark, Aug. 2010.

Part III: Linear Adaptive Filters

Chapter 4. Regularized adaptive notch filters for acoustic howling suppression: In this chapter, a method for the suppression of acoustic howl-

ing is developed, based on adaptive notch filters (ANF) with regularization (RANF). The method features three RANFs working in parallel to achieve frequency tracking, howling detection and suppression. The ANF-based approach to howling suppression introduces minimal processing delay and minimal complexity, in contrast to non-parametric block-based methods featuring a non-parametric frequency analysis. Compared to existing ANF-based howling suppression methods, the proposed method allows for a more advanced howling detection such that tonal components in the source signal are not affected. The RANFs proposed in this chapter are implemented in direct form and are updated using a gradient-descent-type algorithm. Results show that, under certain conditions, the level of suppression and sound quality are similar to what is obtained with block-based methods.

Chapter 4 is based on:

- **Gil-Cacho J. M.**, van Waterschoot T., Moonen M. and Holdt Jensen S., "Regularized adaptive notch filters for acoustic howling suppression", in Proc. 17th European Signal Processing Conference 2009 (EUSIPCO'09), Glasgow, UK, Aug. 2009.

Chapter 5. A frequency-domain adaptive filter (FDAF) prediction error method (PEM) framework for double-talk-robust acoustic echo cancellation: In this chapter, we propose a new framework to tackle the double-talk (DT) problem in acoustic echo cancellation (AEC). It is based on a frequency-domain adaptive filter (FDAF) implementation of the so-called PEM-AFROW algorithm (FDAF-PEM-AFROW). We show that FDAF-PEM-AFROW is by construction related to the best linear unbiased estimate (BLUE) of the echo path. We depart from this framework to show an improvement in performance with respect to other adaptive filters minimizing the BLUE criterion, namely the PEM-AFROW and the FDAF-NLMS with near-end signal normalization. One of the contributions is to propose the instantaneous pseudo-correlation (IPC) measure between the near-end signal and the loudspeaker signal. The IPC measure serves as an indication of the effect of a DT situation occurring during adaptation. We motivate the choice of FDAF-PEM-AFROW over PEM-AFROW and FDAF-NLMS with near-end signal normalization, based on performance, computational complexity and related IPC measure values. Moreover, we use the FDAF-PEM-AFROW framework to improve several state-of-the-art variable step-size (VSS) and variable regularization (VR) algorithms. The FDAF-PEM-AFROW versions significantly outperform the original versions in every simulation by at least 6 dB during DT. In terms of computational complexity the FDAF-PEM-AFROW versions are themselves 10 times cheaper than the original versions.

Chapter 5 is based on:

- **Gil-Cacho J. M.**, van Waterschoot T., Moonen M. and Holdt Jensen

S., "A frequency-domain adaptive filter (FDAF) prediction error method (PEM) framework for double-talk-robust acoustic echo cancellation", Submitted to IEEE Transaction Audio Speech Language Processing, Sept. 2013.

Chapter 6. Wiener variable step size and gradient spectral variance smoothing for double-talk-robust acoustic echo cancellation and acoustic feedback cancellation: In this chapter, we derive a robust and computationally efficient algorithm based on the frequency-domain adaptive filter prediction error method using row operations (FDAF-PEM-AFROW) for DT-robust AEC and for AFC. The proposed algorithm features two main modifications: (a) the WIener variable Step sizE (WISE), and (b) the GRAdient Spectral variance Smoothing (GRASS). In AEC simulations, the WISE-GRASS-FDAF-PEM-AFROW algorithm obtains improved robustness and smooth adaptation in highly adverse scenarios such as in bursting DT at high levels, and in a change of acoustic path during continuous DT. Similarly, in AFC simulations, the algorithm outperforms state-of-the-art algorithms when using a low-order near-end speech model and in colored non-stationary noise.

Chapter 6 is based on:

- **Gil-Cacho J. M.**, van Waterschoot T., Moonen M. and Holdt Jensen S., "Wiener variable step size and gradient spectral variance smoothing for double-talk-robust acoustic echo cancellation and acoustic feedback cancellation", submitted to Elsevier Signal Processing, May 2013.

Part IV: Nonlinear Adaptive Filters

Chapter 7. Linear-in-the-parameters nonlinear adaptive filters for acoustic echo cancellation: In this chapter, we consider the description of odd and even nonlinearities in electrodynamic loudspeakers by means of periodic random-phase multisine signals. It is shown that the odd nonlinear contributions are more predominant than the even ones. This fact implies that *at least* a 3rd-order nonlinear model of the loudspeaker should be used, which is in clear conflict with the extensive, almost unique, use of second-order (quadratic) Volterra filters. We, therefore, consider the identification and validation of a model of the loudspeaker using several linear-in-the-parameters nonlinear adaptive filters, namely, Hammerstein and Legendre polynomial filters of various orders, and a simplified 3rd-order Volterra filter of various lengths. In our measurement set-up, the obtained results imply however, that a 3rd-order nonlinear filter can not capture all the nonlinearities, meaning that odd and even nonlinear contributions are produced by higher-order nonlinearities. Legendre polynomial filters are shown to improve performance monotonically with increasing order whereas Hammerstein filters do not. In the simplified 3rd-order Volterra filter case, the performance is remarkably poorer than in the Legendre polynomial case for the same filter length. The differences between

identification and validation appear to be very small in the Legendre polynomial filter case. However, the simplified 3rd-order Volterra filter presents clear signs of overfitting.

Chapter 7 is based on:

- **Gil-Cacho J. M.**, van Waterschoot T., Moonen M. and Holdt Jensen S., "Study and characterization of odd and even nonlinearities in electrodynamic loudspeakers by means of periodic random-phase multisine", in Proc. of the 127th Audio Engineering Society (AES) convention, NY, USA, Oct. 2009.
- **Gil-Cacho J. M.**, van Waterschoot T., Moonen M. and Holdt Jensen S., "Linear-in-the-parameters nonlinear adaptive filters for acoustic echo cancellation.", submitted to Journal of the Audio Engineering Society (JAES), July 2013.

Chapter 8. Nonlinear acoustic echo cancellation based on a sliding-window leaky kernel affine projection algorithm: In this chapter, we propose an algorithm based on the kernel affine projection algorithm (KAPA). KAPA has been successfully applied to many areas in signal processing but not yet to nonlinear AEC (NLAEC). The contribution of this chapter is three-fold: (1) to apply KAPA to NLAEC applications, (2) to develop a sliding-window leaky KAPA (SWL-KAPA) that is well suited for NLAEC applications, and (3) to propose a kernel function, consisting of a weighted sum of a linear and a Gaussian kernel. In our experiment set-up, the proposed SWL-KAPA for NLAEC consistently outperforms the linear APA, resulting in up to 12 dB of improvement in echo return loss enhancement (ERLE) at a computational cost that is only 4.6 times higher. Moreover, it is shown that the SWL-KAPA outperforms, by $4 - 6$ dB, a Volterra-based NLAEC, which itself has a much higher computational cost than the linear APA.

Chapter 8 is based on:

- **Gil-Cacho J. M.**, Signoretto M., van Waterschoot T., Moonen M. and Holdt Jensen S., "Nonlinear Acoustic Echo Cancellation based on a Sliding-Window Leaky Kernel Affine Projection Algorithm", IEEE Transactions on Audio, Speech, and Language Processing, vol. 21, no. 9, pp. 1867-1878, Sept. 2013.

Bibliography

[1] A. Papoulis, *Signal Analysis*. New York, NY: McGraw Hill, 1977.

[2] A. V. Oppenheim, A. S. Willsky, S. H. Nawab, *Signals and Systems*. Englewood Cliffs, New Jersey: Prentice Hall, 1997.

[3] E. Hänsler, G. Schmidt, *Acoustic Echo and Noise Control: A Practical Approach*. New York: John Wiley & Sons, Inc., 2004.

[4] B. Widrow, S. D. Stearns, *Adaptive Signal Processing*. Upper Saddle River, New Jersey: Prentice Hall, 1985.

[5] S. Haykin, B. Widrow, *Least-Mean-Square Adaptive Filters*. New York: John Wiley & Sons, Inc., 2003.

[6] K-A. Lee, W-S. Gan, S. M. Kuo, *Subband adaptive filtering : Theory and implementation*. Chichester, West Sussex: John Wiley & Sons, Ltd, 2009.

[7] M. Montazeri and P. Duhamel, "A set of algorithms linking NLMS and block RLS algorithms", *IEEE Trans. Signal Process.* 43, no. 2, February 1995, 444-453.

[8] D. Mansour, A. H. Gray, "Unconstrained frequency-domain adaptive filters", *IEEE Trans. Acoust. Speech Signal Process.*, 30(3), October 1982, 726-734.

[9] A. Spriet, I. Proudler, M. Moonen, J. Wouters, "Adaptive feedback cancellation in hearing aids with linear prediction of the desired signal", *IEEE Trans. Signal Process.* 53, no. 10 (2005) 3749–3763.

[10] L. Ljung, *System Identification: Theory for the user*. Englewood Cliffs, New Jersey: Prentice Hall, 1987.

[11] T. van Waterschoot, M. Moonen, "Fifty years of acoustic feedback control: state of the art and future challenges", *Proc. IEEE* 99, no. 2 (2011) 288–327.

[12] J. J. Shynk, "Frequency-domain and multirate adaptive filtering", *IEEE Signal Process. Mag.* 9, no. 1 (1992) 14–37.

[13] B. Farhang-Boroujeny,*Adaptive filters: Theory and applications*. Chichster, UK: Wiley, 1998.

[14] J. Benesty, T. Gänsler, D. R. Morgan, M. M. Sondhi, and S. L. Gay, *Advances in Network and Acoustic Echo Cancellation*. Berlin: Springer-Verlag, 2001.

[15] S. Haykin, *Adaptive Filter Theory*. Upper Saddle River, NY: Prentice Hall, 2002.

[16] P. S. R. Diniz, *Adaptive filtering: Algorithms and Practical Implementations*. Boston, MA: Springer, 2008.

[17] P. Gil-Cacho, T. van Waterschoot, M. Moonen, and S. H. Jensen, "Study and characterization of odd and even nonlinearities in electrodynamic loudspeakers by means of periodic random-phase multisine," in *Preprints AES 127th convention*, New York, USA, Oct. Oct. 2009, Preprint 7841.

[18] T. van Waterschoot, G. Rombouts, and M. Moonen, "Optimally regularized adaptive filtering algorithms for room acoustic signal enhancement," *Signal Process.*, vol. 88, no. 3, pp. 594–611, Mar. 2008.

[19] B. Widrow and M. Hoff, "Adaptive switching circuits," in *Proc. WESCON Conv. Rec.*, vol. 4, 1960, pp. 96–140.

[20] K. Ozeki and T. Umeda, "An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties," *Electronics and Communication in Japan*, vol. 67, no. 5, pp. 19–27, Aug. 1984.

[21] A. J. M. Kaizer, "Modeling of the nonlinear response of an electrodynamic loudspeaker by a Volterra series expansion," *J. Audio Eng. Soc.*, vol. 35, no. 6, pp. 421–432, June 1987.

[22] V. J. Mathews and G. L. Sicuranza, *Polynomial Signal Processing*. New York: Wiley, 2000.

[23] G. L. Sicuranza and A. Carini, "On the accuracy of generalized hammerstein models for nonlinear active noise control," in *Proc. 2006 IEEE Inst. Meas. Tech. Conf.*, Sorrento, Italy, Apr. 2006, pp. 24–27.

[24] F. T. Agerkvist,"Modelling loudspeaker non-linearities", in *Proc. 32nd AES Conference*, Hillerød Denmark, Sep. 2007.

[25] H. Nyquist, "Regeneration theory," *Bell Syst. Tech. J.*, vol. 11, pp. 126–147, 1932.

[26] A. Spriet, S. Doclo, M. Moonen, , and J. Wouters, *Feedback control in hearing aids*. Springer, Germany, ch. 6 in "Part H. Speech Enhancement of the Springer Handbook of Speech Processing and Speech Communication (Benesty J., Huang Y. A., Sondhi M., eds.)".

[27] A. Stenger and R. Rabenstein, "Adaptive Volterra filters for acoustic echo cancellation," in *Proc. 1999 IEEE-EURASIP Workshop Nonlinear Signal and Image Process.*, (NSIP'99), Antalya, Turkey, June 1999.

[28] B. Baykal and A. Constantinides, "Underdetermined-order recursive least-squares adaptive filtering: the concept and algorithms," *IEEE Trans. Signal Process.*, vol. 45, no. 2, pp. 346–362, Feb. 2008.

[29] M. M. Sondhi and D. A. Berkley, "Silencing echoes on the telephone network," *Proc. IEEE*, vol. 68, pp. 948963, Aug. 1980.

.

Part II

Room Acoustics Modeling

Chapter 2

Estimation of acoustic resonances for room transfer function equalization

Estimation of acoustic resonances for room transfer function equalization

Jose M. Gil-Cacho, Toon van Waterschoot, Marc Moonen and Søren Holdt Jensen

Published in Proc. of the International Workshop on Acoustic Echo and Noise Control (IWAENC'10), Tel Aviv, Israel, Aug. 2010.

Contributions of first author

- literature study
- design of simulation scenarios
- software implementation and computer simulations
- co-interpretation of simulation results
- text redaction and editing

Abstract

In this chapter, the utilization of different norms (i.e., 2-norm and 1-norm) and models (i.e., all-pole and pole-zero) for room transfer function (RTF) modeling and equalization is discussed. It is known that strong acoustic resonances create long room impulse responses (RIRs) which may harm the speech transmission in an acoustic space and, hence, reduce speech intelligibility. Equalization is, therefore, performed by compensating for the main acoustic resonances common to multiple room transfer functions (RTFs) measured in the same room. These acoustic resonances may be modeled by means of the common poles of the RTFs. In the literature, however, there is no consensus about which model (i.e., all-pole or pole-zero) generates pole estimates that perform better in RTF equalization. Furthermore, MSE (i.e., 2-norm) minimization is typically employed for the estimation of the poles. In this chapter, an MAE (i.e., 1-norm) minimization is further proposed for pole estimation. A comparative evaluation of these different norms and models in terms of their residual RTF and residual RIR (i.e., the residuals after equalization) is provided.

2.1 Introduction

THE sound transmission characteristics between a loudspeaker and a microphone are described in the frequency domain by the room transfer function (RTF) or in the time domain by the room impulse response (RIR) which depends on the loudspeaker-microphone position. In some audio applications (e.g., sound reproduction systems in train stations or other large spaces) where speech intelligibility is an issue, an equalization filter is commonly used to compensate for the frequency response of the room. Such a filter may remove, by inverse filtering, acoustical artifacts present in spaces with long RIR originating from strong acoustic resonances. The equalization performance then depends on the model from which the inverse filter is derived. The most common model for the RTF is an *all-zero* model [1]. It represents the physics of room acoustics where a microphone signal is a weighted sum of discrete reflections of the loudspeaker signal. Its drawback is that any change in the loudspeaker-microphone or obstacle position inside the room will change all the coefficients of the model. The equalization filter is then a single point inverse filter which is only valid for a single point in the room and therefore a recalculation of every coefficient will be needed at each and every other point in the room [6].

One of the alternatives is to use a *pole-zero* model for the RTF [1]. This model represents the physics of room acoustics by also including the modeling of the acoustic resonances by means of the poles of the transfer function. Poles can represent long impulse responses caused by resonances, while zeros represent

time delays and anti-resonances [7]. Yet another alternative model is the *all-pole* model, which represents only the acoustic resonances in an effort to model the RTF spectral envelope [1]. This provides an appropriate strategy to cancel only the main resonances discarding the cancellation of the zeros. Moreover, the concept of common acoustical poles, first introduced in [7], can be applied to these two alternative models. The underlying idea is that the acoustic resonances in a room depend on the dimensions and shape of the enclosure and not on the loudspeaker-microphone position. Each RTF in the room may then be expressed using a common set of poles and different zeros. Hence an inverse filter that cancels the common acoustical poles can be created that equalizes the physically common main resonances in multiple points in the room [6]. Finding the poles of the transfer function may be seen as an optimization problem in which the set of poles that minimizes the error between the model and the measurements (i.e., the actual impulse response) is sought for. Different models and error criteria will obviously render different pole estimates and consequently different equalization filters and residual RTFs. Typically a least squares error criterion (2-norm minimization) is employed. In the literature, however, it is not clear which of these models generate pole estimates that perform better in the RTF equalization contest.

In this chapter, the use of a least absolute error criterion (1-norm minimization) for RTF pole estimation is proposed. In [2], sparse linear prediction of speech signals is used to obtain sparse residuals with minimum number on non-zeros elements. With this idea in mind, 1-norm minimization is proposed here to calculate the poles of an all-pole model, so that the inverse filtering will render a residual impulse response having discrete separated reflection rather than a dense impulse response. On the other hand, it is found that the poles calculated from a pole-zero model using 1-norm minimization gives results similar to the 2-norm pole-zero approach and therefore this will not be considered any further. The aim of this chapter is hence to make a comparative evaluation involving 2-norm minimization using an all-pole and a pole-zero model and 1-norm minimization using an all-pole model. The comparative evaluation will be presented both in time and frequency domain. Several questions on how the choice of a model and norm affects the residual RTF and RIR will be addressed.

The chapter is organized as follows. In Section 2.2, the mathematical formulation is presented for each model. In Section 2.3, equalization results using the presented techniques on real measured room impulse responses are presented. Finally, Section 2.4 concludes the chapter.

2.2 Pole estimation using different norms

Although the RTFs are different for each loudspeaker-microphone position, all RTFs in a room share the same resonance frequencies. These resonance frequencies may be visible as spectral peaks in the RTFs [1]. If only the zeros cause RTF variation then the RTFs can be expressed using a common denominator for all and a different numerator for each of them. This can be represented by either common poles, $p(k)$, and distinct zeros, $z_i(k,t)$, or in polynomial form using common autoregressive (AR), $a(k)$, and distinct moving average (MA), $b_i(k,t)$, coefficients [7],

$$H_i(q,t) = \frac{\prod_{k=1}^{Q}(1 - z_i(k,t)q^{-1})}{\prod_{k=1}^{P}(1 - p(k)q^{-1})} = \frac{\sum_{k=1}^{Q} b_i(k,t)q^{-k}}{1 - \sum_{k=1}^{P} a(k)q^{-k}} \tag{2.1}$$

where Q and P are the order of numerator and denominator respectively, $i = 1, ..., M$ the number of RTFs and where q denotes the time shift operator, i.e., $q^{-k}u(t) = u(t-k)$.

Mathematically, the class of problems considered in this chapter can be cast into one general optimization problem associated with finding the filter coefficient vector $\mathbf{x}$ from a set of measured impulse responses cast in $\mathbf{v}$ and $\mathbf{W}$[1], so that the error $\mathbf{e} = \mathbf{v} - \mathbf{W}\mathbf{x}$ is minimized as

$$\min_{\mathbf{x}} \|\mathbf{v} - \mathbf{W}\mathbf{x}\|_p^p \tag{2.2}$$

where $\|\cdot\|_p$ is the p-norm defined for $p > 1$ as $\|\mathbf{x}\|_p = \left(\sum_{t=1}^{N} |x(t)|^p\right)^{1/p}$. Matrix $\mathbf{W}$ and vector $\mathbf{v}$ involved in the minimization problem (2.2) depend on whether the model for pole estimation is all-pole or pole-zero and whether common acoustical poles are considered. In the pole-zero model with common acoustical poles case, matrix $\mathbf{W}$ and vector $\mathbf{v}$ are formed as

$$\mathbf{v} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_M]^T \tag{2.3}$$

$$\mathbf{h}_i = [h_i(0), h_i(1), ..., h_i(N-1), 0, 0, 0]^T \tag{2.4}$$

$$\mathbf{x} = [\mathbf{a}, \mathbf{b}_1, ..., \mathbf{b}_M]^T \tag{2.5}$$

$$\mathbf{a} = [a(1), a(2), ..., a(P)]^T \tag{2.6}$$

$$\mathbf{b}_i = [b_i(0), b_i(1), ..., b_i(Q)]^T \tag{2.7}$$

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_1 & \mathbf{D} & 0 & 0 & 0 \\ \mathbf{W}_2 & 0 & \mathbf{D} & 0 & 0 \\ \vdots & \cdots & 0 & \ddots & 0 \\ \mathbf{W}_M & \cdots & \cdots & \cdots & \mathbf{D} \end{bmatrix} \tag{2.8}$$

[1]Note the notation is different from the Introduction chapter

$$\Rightarrow \text{size } [M(N+P-1) \times (P+M(Q+1))]$$

$$\mathbf{D} = \begin{bmatrix} 1 & & & \\ & 1 & 0 & \\ & & \cdot & \\ & 0 & & 1 \\ 0 & & & 0 \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ 0 & \cdot & \cdot & 0 \end{bmatrix} \tag{2.9}$$

$$\Rightarrow \text{size } [M(N+P-1) \times (P+M(Q+1))]$$

$$\mathbf{W}_i = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ h_i(0) & 0 & \cdots & 0 \\ h_i(1) & h_i(0) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & h_i(0) \\ h_i(N-1) & 0 & \cdots & \vdots \\ 0 & h_i(N-1) & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & h_i(N-1) \end{bmatrix} \tag{2.10}$$

$$\Rightarrow \text{size } [(N+P-1) \times P]$$

where h_i is the i^{th} length-N measured impulse response. If an all-pole model is considered, matrix $\mathbf{D}$ and vectors $\mathbf{b}_j$ are just set to zero and vector $\mathbf{x}$ will only consist of AR coefficients (i.e., $\mathbf{x} = \mathbf{a}$). Conversely, when a pole-zero model is considered, vector $\mathbf{x}$ will consist of both AR and MA coefficients (i.e., $\mathbf{x} = [\mathbf{a}, \mathbf{b}_1, ..., \mathbf{b}_M]^T$). The set of $\mathbf{a}$ coefficients is estimated using either 2-norm or 1-norm minimization and forms the filter polynomial $A(q) = 1 - a(1)q^{-1} - ... - a(P)q^{-P}$. The residual $\tilde{B}_i(q)$ is the result of multiplying the actual RTF with the inverse filter $\tilde{A}(q)$ as

$$\tilde{A}(q)H_i(q,t) = \tilde{A}(q)\frac{B_i(q)}{A(q)} = \tilde{B}_i(q) \tag{2.11}$$

where $\tilde{A}(q) \cong A(q)^{-1}$. The description of $\tilde{B}_i(q)$ in both time and frequency domain (i.e., residual RIR and residual RTF respectively) is directly affected by how $\tilde{A}(q)$ is calculated. In 2-norm (i.e., least squares error) minimization the optimal filter coefficient vector may be given in closed-form as

$$\mathbf{x} = \left(\mathbf{W}^T\mathbf{W}\right)^{-1}\mathbf{W}^T\mathbf{v} \tag{2.12}$$

However, in 1-norm (i.e., least absolute error) minimization there exists no closed-form solution and therefore the filter coefficient vector is calculated as a solution to a convex optimization problem, which can be solved efficiently, e.g. using `CVX` [8]. It is found, experimentally, that similar ARMA coefficients estimates are extracted from a pole-zero model using 1-norm or 2-norm minimization, so only 2-norm pole-zero model will be considered in the sequel.

2.3 Results from measured impulse responses

In this section, results obtained from two measured room impulse responses are presented. The three methods (i.e., 1-norm all-pole and 2-norm all-pole and pole-zero) are compared by inspecting both the residual RTF and the residual RIR. Matlab computer simulations are performed at $fs = 16$ kHz. The impulse responses, shown in Figure 2.1, (h_1 and h_2) of length $N = 2001$ samples are measured in a rectangular room of about $5 \times 3 \times 3$ m. The order is chosen as $P = 500$ and in the ARMA case also $Q = 500$. This order is found to be sufficient to accurately model the measured impulse response in the ARMA case and thus serve as a quality reference. It is noted that parameters $P = 500$ and $Q = 500$ are suitable for these two impulse responses but other values for P and Q may be needed for other lengths or type of Impulse responses.

Two objective measures are employed:

- Spectral Flatness (SF)

The spectral flatness is calculated by dividing the geometric mean of the power spectrum by the arithmetic mean of the power spectrum, i.e.

$$SF = \frac{\sqrt[N]{\prod_{f=0}^{N-1} P(f)}}{\frac{\sum_{f=0}^{N-1} P(f)}{N}} \tag{2.13}$$

where $P(f)$ represents the magnitude of the f^{th} frequency bin, with $N = 512$.

- Sparseness Degree (SD)

The sparseness degree is the number of elements in the residual RIR that have an absolute value smaller than some threshold T close to zero, i.e.

$$SD = \{R(n) : |R(n)| < T\} \tag{2.14}$$

where $R(n)$ represents the residual RIR and $T = 2 \cdot 10^{-6}$.

Two different cases for each method are presented. In the **first case**, the coefficients of the inverse filter are calculated from one single impulse response,

h_1, shown in Figure 2.1(a), and used to equalize the same h_1. In the **second case**, the coefficients of the inverse filter are calculated using the set of two measured impulse responses (i.e., with common acoustical poles) and used to equalize h_1 . The frequency response of h_1 is shown in Figure 2.1(b).

- **First case**

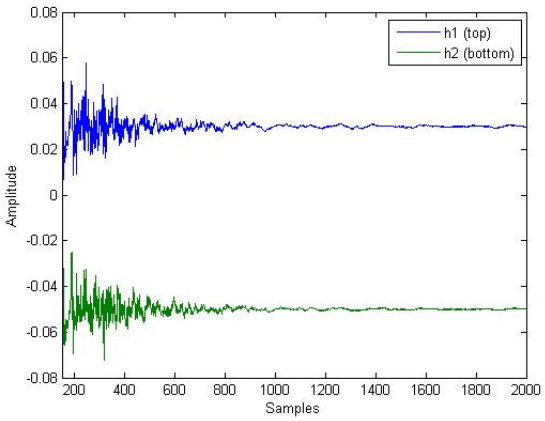
When $\tilde{A}(q)$ is calculated from the all-pole model using 2-norm minimization, in Figure 2.2(a) and 2.2(b), the flattest residual RTF is achieved. On average, the magnitude difference between peaks and dips is very small; however, it exhibits a long noise-like residual RIR because the 2-norm minimization shapes the residual into coefficients that exhibit Gaussian like features [2]. When $\tilde{A}(q)$ is calculated from the pole-zero model using 2-norm minimization, in Figure 2.2(c) and 2.2(d), a highly colored residual RTF is achieved, which is a perceptually undesirable characteristic in RTF equalization [5]. However, the short residual RIR may be desirable in other audio applications such as acoustic echo or feedback cancellation [3], [4]. When $\tilde{A}(q)$ is calculated from the all-pole model using 1-norm minimization, in Figure 2.2(e) and 2.2(f), the residual RTF has been flattened with respect to the true RTF and, in addition, the main low-frequency resonances have been canceled. The residual RIR exhibits a sparse distribution of non-zero coefficients, which implies that the acoustic reflections modeled by the residual RIR are forced to be more spaced in time. The residual RIR from the 1-norm all-pole model shows the largest number of zero coefficients, i.e., the highest degree of sparseness. These results are summarized in Table 2.1, where it can be seen that the 1-norm all-pole model and 2-norm all-pole model present the highest SD and the highest SF respectively.

- **Second case**

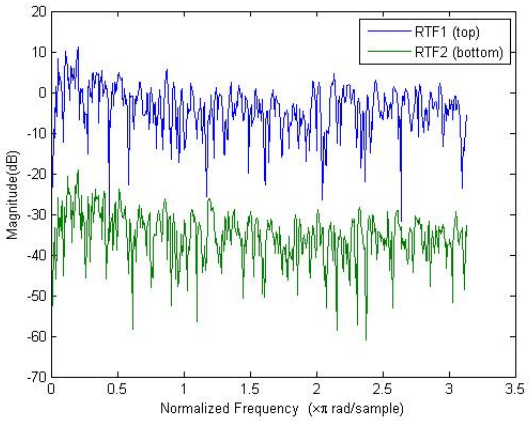
Figure 2.3 shows that although in the common-acoustical-poles case the overall performance has been deteriorated with respect to the single impulse response case, the same main features can be observed on the residuals.

	2-norm all-pole	pole-zero	1-norm all-pole
SF	0.7	0.06	0.2
SD	5	43	503

Table 2.1: SF and SD for the 1-norm all-pole, 2-norm all-pole and pole-zero case.

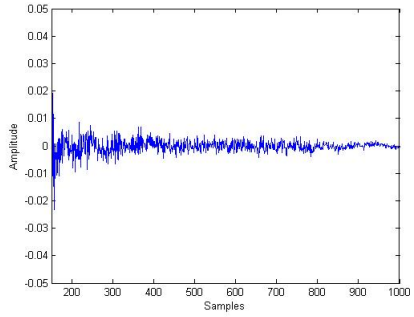


(a) Time evolution of h_1 and h_2

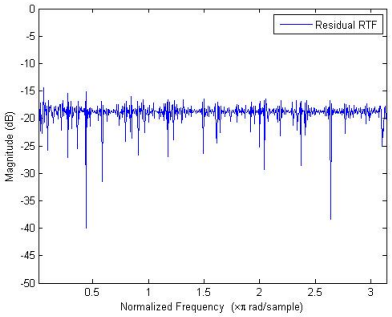


(b) Spectrum of h_1 and h_2 (RTF1 and RTF2)

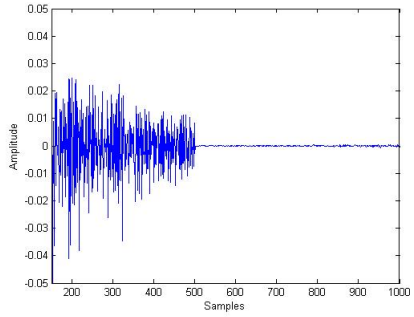
Figure 2.1: Frequency response function and time evolution of h_1 and h_2. Normalized frequency π represents $f_s/2$.



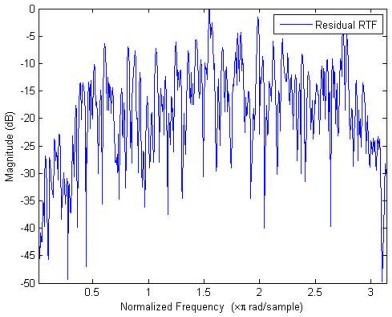
(a) 2-norm all-pole model



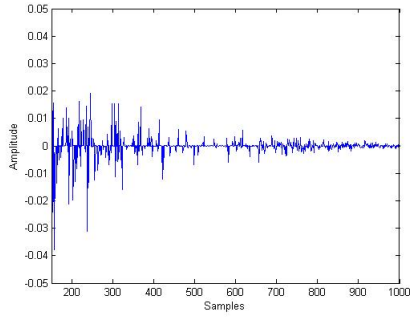
(b) 2-norm all-pole model



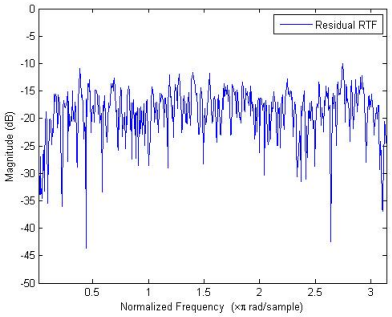
(c) pole-zero model



(d) pole-zero model

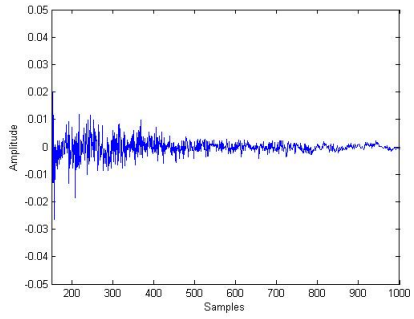


(e) 1-norm all-pole model

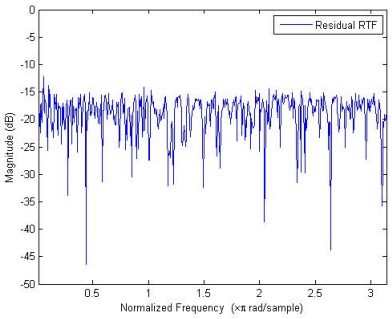


(f) 1-norm all-pole model

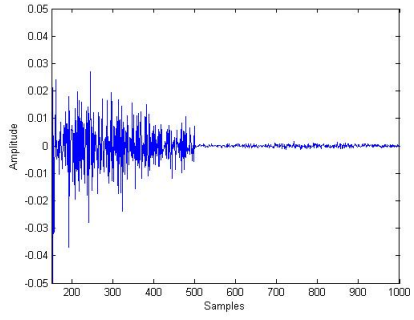
Figure 2.2: Equalization prefilter obtained from h_1. (a),(c),(e) h_1 residual RIRs. (b),(d),(f) h_1 residual RTFs. Normalized frequency π represents $f_s/2$.



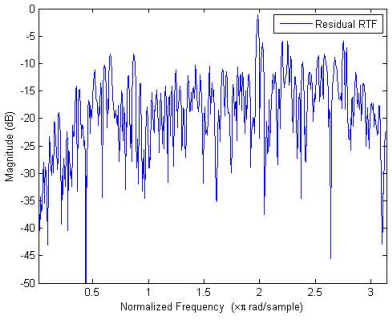
(a) 2-norm all-pole model



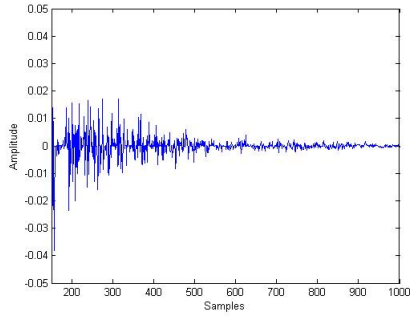
(b) 2-norm all-pole model



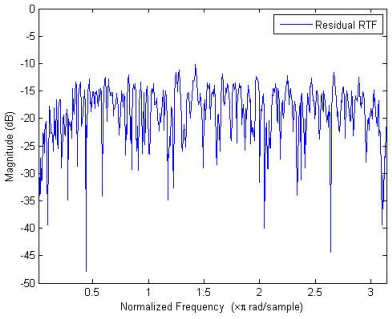
(c) pole-zero model



(d) pole-zero model



(e) 1-norm all-pole model



(f) 1-norm all-pole model

Figure 2.3: Equalization prefilter obtained from from common-acoustical-poles calculation. (a),(c),(e) h_1 residual RIRs. (b),(d),(f) h_1 residual RTFs.

2.4 Conclusions

In this chapter, results from RTF equalization have been presented. Equalization is achieved by canceling the poles associated with the main resonances common to multiple RTFs in a room. Poles were estimated by means of three different methods. Poles estimated using 2-norm minimization and an all-pole model offered a residual RTF having the flattest response. Poles estimated using 2-norm minimization and a pole-zero model offered a large reduction in the main low-frequency acoustic resonances. However the residual RTF exhibited a highly colored response, while the residual RIR was shorter in time. Finally, poles estimated using 1-norm minimization and an all-pole model offered a flat residual RTF with its main resonances canceled and a sparse residual RIR. This means that the residual RIR will represent sparsely distributed discrete reflections. This feature may be desirable in speech applications although a deeper study taking into account perceptual considerations would be needed.

Bibliography

[1] J. Mourjopoulos and M. A. Paraskevas, "Pole and zero modeling of room transfer functions", *J. Sound and Vibration*, vol. 146, no. 2, pp. 281-302, April 1991.

[2] J. A. Cadzow, "Minimum l_1, l_2, and l_∞ Norm Approximate Solutions to an Overdetermined System of Linear Equations", *Dig. Sig. Proc.*, vol. 12, no. 4, pp. 524-560, Oct. 2002.

[3] T. van Waterschoot and M. Moonen, "Adaptive feedback cancellation for audio applications," *Signal Processing*, vol. 89, no. 11, pp. 2185-2201, Nov. 2009.

[4] P. S. R. Diniz, *Adaptive Filtering: Algorithms and Practical Implementations*. Springer, Boston, MA., 2008.

[5] D. M. Howard and J. A. S. Angus, *Acoustics and Psychoacoustics*. Focal Press, Elsevier, 2006.

[6] Y. Haneda, S. Makino and Y. Kaneda, "Multiple-Point Equalization of Room Transfer Functions by Using Common Acoustical Poles," *IEEE Trans. Speech Audio Process.*, vol. 5, no. 4, pp. 325-333, July 1997.

[7] Y. Haneda, S. Makino and Y. Kaneda, "Common Acoustical Pole and Zero Modeling of Room Transfer Function," *IEEE Trans. Speech Audio Process.*, vol. 2, no. 2, pp. 320-328, Apr. 1994.

[8] M. Grant and S. Boyd, CVX: Matlab software for disciplined convex programming (web page and software). http://cvxr.com/cvx, April, 2010.

Chapter 3

Multi-Microphone acoustic echo cancellation using warped multi-channel linear prediction of common acoustical poles

Multi-Microphone acoustic echo cancellation using warped multi-channel linear prediction of common acoustical poles

Jose M. Gil-Cacho, Toon van Waterschoot, Marc Moonen and Søren Holdt Jensen

Published in Proc. 18th European Signal Processing Conference 2010 (EUSIPCO'10), Aalborg, Denmark, Aug. 2010.

Contributions of first author

- literature study
- design of simulation scenarios
- software implementation and computer simulations
- co-interpretation of simulation results
- text redaction and editing

Abstract

In this chapter, a computationally cheap extension from single-microphone AEC to multi-microphone AEC is presented for the case of a single loudspeaker. It employs the idea of common-acoustical-pole and zero RTF modeling. The RTF models used for multi-microphone AEC share a fixed common denominator polynomial, which is calculated off-line by means of a multi-channel warped linear prediction. By using the common denominator polynomial as a prefilter, only the numerator polynomial has to be estimated recursively for each microphone, hence adapting to changes in the room. This approach allows the number of numerator coefficients to decrease by one order of magnitude for each microphone compared with all-zero modeling. In a first configuration, the prefiltering is done on the adaptive filter signal, hence achieving a pole-zero model of the RTF in the AEC. In a second configuration, the prefiltering is done on the loudspeaker signal, hence achieving a dereverberation effect, in addition to AEC, on the microphone signals.

3.1 Introduction

ACOUSTIC echo cancellation (AEC) is used in speech communication applications where the existence of echoes degrades the intelligibility and listening comfort, such as in mobile and hands-free telephony and in teleconferencing. An acoustic echo canceler seeks to cancel the echo signal component $y(t)$ in the microphone signal, ideally leading to an *echo-free* error signal $e(t)$. This is done by subtracting an estimate of the echo signal $\hat{y}(t)$ from the microphone signal as in Figure 3.1. Therefore, an adaptive filter is used to provide a model that represents the best fit to the echo path H or room impulse response (RIR) [6]. This model is used to filter the input signal $x(t)$ to obtain the estimated echo signal. The most common model for the RIR is the FIR filter corresponding with an all-zero model of the room transfer function (RTF). This is due to well-known behavior and ensured stability [5]. It is connected to the physics of room acoustics as being a weighted sum of previous input samples or, in other words, discrete reflections of the loudspeaker signal. Its drawback is that the number of filter coefficients required to model a RIR increases dramatically if the RIR is long which is typical in room acoustics applications. Besides, any change of loudspeaker-microphone or obstacle position inside the room will change the coefficients of the model and therefore a recalculation of every coefficient will be needed.

One of the alternatives is to use an IIR filter which corresponds to a pole-zero model of the RTF. This model reduces the number of parameters to be estimated for the same modeling capabilities [5]. Moreover, this model represents the physics of room acoustics including the modeling of the acoustic resonances

by means of the poles of the transfer function. Poles can represent long impulse responses caused by resonances and the zeros represent time delays and anti-resonances [8]. The well known drawback of this model is the nonlinear shape of the cost function and potential instability [5]. Although different algorithms have been proposed to overcome these limitations with more or less success, their use, especially using higher order filters is very limited in practice [5]. In [8], the concept of a common-acoustical-pole model is introduced. The underlying idea is that the acoustic resonances depend on the dimensions and shape of the enclosure and not on loudspeaker-microphone position. Each RTF might be expressed using a common set of poles and different zero functions.

By using the common denominator polynomial as a prefilter, see Figure 3.1 and Figure 3.2, only the numerator polynomial has to be estimated recursively for each microphone, avoiding the problems of adaptive IIR filtering but reducing the number of parameters to be estimated. In order to further increase the reduction in the number of coefficients, the calculation of common poles will be performed in the frequency-warped domain. By frequency-warping the RIR, one is able to focus the computational resources in frequency regions of interest [1], [2] relaxing the modeling on those frequency regions that are of less interest [4].

This chapter is organized as follows. In Section 3.2, the proposed model is shown. In Section 3.2.1 the concept of the common-acoustical-pole and zero model is further explained. In Section 3.2.2 the standard procedure for frequency warping is explained. In Section 3.2.3 the equations for multi-channel warped linear prediction are presented. Section 3.3 shows the adaptive algorithm employed in the proposed AEC and the signals participating which depend on whether the prefilter is located in the adaptive filter signal path or in the loudspeaker signal path. In Section 3.4, we describe simulation results that illustrate the performance of our proposed model in terms of echo reduction. Finally Section 3.5 concludes the chapter.

3.2 Proposed Model

3.2.1 Common-acoustical-pole and zero model

Although the RTFs are different for each loudspeaker-microphone position, all RTFs in a room share the same resonance frequencies. Resonance frequencies may be visible in the spectral peaks of each RTF and are considered to be independent of the loudspeaker-microphone position [5]. If only the zeros cause RTF variation, then each RTF can be expressed using a common denominator for all and a different numerator for each of them, i.e. $H_i(q) = B_i(q)/A_c(q)$, as in Figure 3.2.

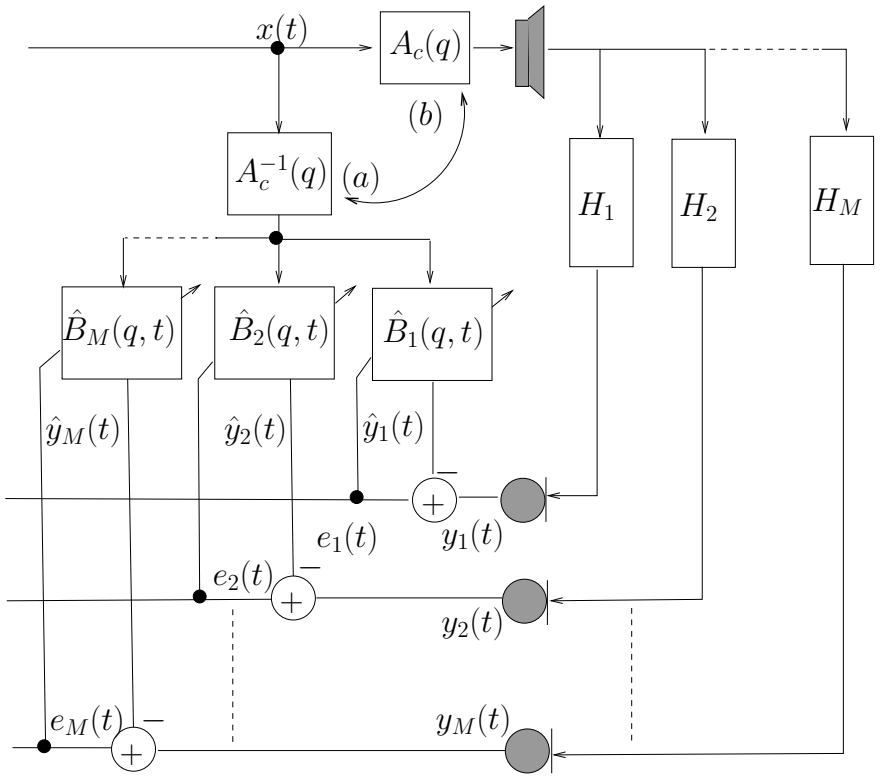


Figure 3.1: Multi-microphone acoustic echo canceler set-up.

This can be represented by either common poles $p_c(k)$ and distinct zeros $z_i(k,t)$, or in polynomial form using common autoregressive (AR) $a_c(k)$ and distinct moving average (MA) $b_i(k,t)$ coefficients [8],

$$H_i(q,t) = \frac{\prod_{k=1}^{Q}(1 - z_i(k,t)q^{-1})}{\prod_{k=1}^{P}(1 - p_c(k)q^{-1})} = \frac{\sum_{k=1}^{Q} b_i(k,t)q^{-k}}{1 - \sum_{k=1}^{P} a_c(k)q^{-k}} \tag{3.1}$$

where Q and P are the order of numerator and denominator respectively, $i = 1, ..., M$, M the number of microphones and q denotes the time shift operator, i.e., $q^{-k}u(t) = u(t-k)$.

3.2.2 Warped linear prediction

In order to get the AR coefficients of an impulse response, a set of simultaneous equations must be solved. This set of simultaneous equations are in our case of the same mathematical form comparing Wiener-Hopf equations for lin-

ear prediction, Yule-Walker equation for an autoregressive model [6] and the equation error between a measured impulse response and the estimated all-pole impulse response [7]. In this chapter, it is used the term warped linear prediction (WLP) as in [3]. The standard procedure in producing frequency-warped impulse responses involves replacing the unit delay operator, q^{-1} of the original RTF, $H_i(q) = \sum_{t=0}^{N} h_i(t) q^{-t}$, by first-order all-pass filter [1] $D(q, \lambda)$, i.e.,

$$H_i^w(q) = \sum_{t=0}^{N} h_i(t) D^t(q, \lambda) \tag{3.2}$$

with the warping parameter $\lambda \in (-1, 1)$, N denotes the impulse response length and

$$D(q, \lambda) = \frac{q^{-1} - \lambda}{1 - \lambda q^{-1}}, \tag{3.3}$$

and where the superscript w means that the impulse response or RTF is transformed to the warped domain.

The inverse mapping or dewarping (i.e., from warped domain to original domain) follows directly by applying again the same mapping with the sign of λ

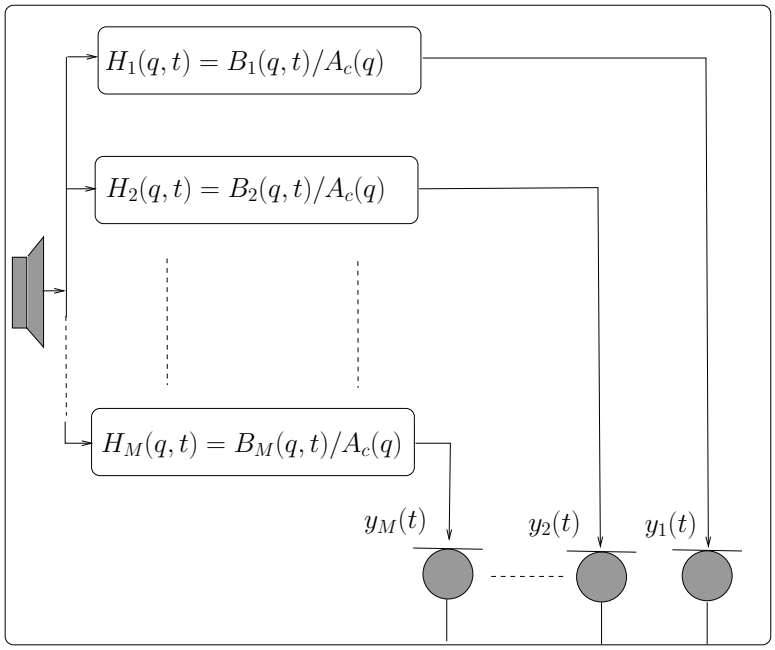


Figure 3.2: Different RTFs in a room. They have the same denominator based on their common acoustical poles.

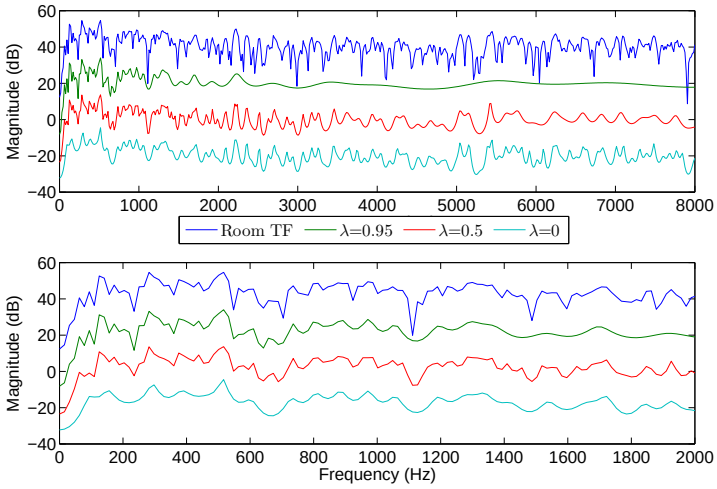


Figure 3.3: Room transfer function (upper line) and frequency response function of warped AR coefficients using different warping parameter $\lambda = 0.95$, $\lambda = 0.5$, $\lambda = 0$.

changed [1] as,

$$H_i(q) = \sum_{t=0}^{N} h_i^w(t) D^t(q, -\lambda). \tag{3.4}$$

Figure 3.3 shows the RTF of a measured room impulse response of 2001 samples length ($f_s = 16$ kHz) together with the AR coefficients spectra calculated with order $P = 200$ and warping the impulse response with $\lambda = 0$, $\lambda = 0.5$ and $\lambda = 0.9$ respectively. With $\lambda = 0$ the AR coefficients model uniformly over the frequency axis. With $\lambda = 0.5$ the modeling effort is put in low frequency regions and $\lambda = 0.9$ even in lower regions. As it can be seen, the low frequency region contains the main resonant peaks. Therefore it seems obvious to employ the limited computational resources in that area.

3.2.3 Warped multi-channel linear prediction of common acoustical poles

From the set of warped impulse responses a set of common AR coefficients are first calculated and then transformed back to the original domain. This set of AR coefficients corresponds to the main resonances of the room. Assuming that $h_i^w(t)$ is the time-domain version of the warped RTF $H_i^w(q)$, the estimated

warped all-pole version of the impulse response is [1]

$$\hat{h}_i^w(t) = \sum_{k=1}^{P} a_c^w(k) h_i^w(t-k) \tag{3.5}$$

The warped common AR coefficients can be calculated as those that minimize the cost function

$$\min_{a_c^w(k)} \sum_{i=1}^{M} \sum_{t=0}^{\infty} e_i^2(t) \tag{3.6}$$

with

$$\begin{aligned} e_i(t) &= h_i^w(t) - \hat{h}_i^w(t) \\ &= h_i^w(t) - \sum_{k=1}^{P} a_c^w(k) h_i^w(t-k). \end{aligned} \tag{3.7}$$

The warped multi-channel linear prediction of the common AR coeffients, i.e., $a_c^w(k)$, that minimize (3.6) is calculated by solving the normal equations [7] as

$$\mathbf{a} = \left(\mathbf{W}^T\mathbf{W}\right)^{-1}\mathbf{W}^T\mathbf{v} \tag{3.8}$$

where

$$\mathbf{a} = [a_c^w(1), a_c^w(2), ..., a_c^w(P)]^T \tag{3.9a}$$

$$\mathbf{W} = [\mathbf{H}_1^T, \mathbf{H}_2^T, ..., \mathbf{H}_M^T]^T \tag{3.9b}$$

$$\mathbf{v} = [\mathbf{h}_1^{wT}, \mathbf{h}_2^{wT}, ..., \mathbf{h}_M^{wT}]^T \tag{3.9c}$$

$$\mathbf{h}_i^w = [h_i^w(1), h_i^w(2), .., h_i^w(N-1), 0, 0, .., 0]^T \tag{3.9d}$$

$$\mathbf{H}_i = \begin{bmatrix} h_i^w(0) & 0 & \cdots & 0 \\ h_i^w(1) & h_i^w(0) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & h_i^w(0) \\ h_i^w(N-1) & 0 & \cdots & \vdots \\ 0 & h_i^w(N-1) & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & h_i^w(N-1) \end{bmatrix} \tag{3.9e}$$

$$\Rightarrow \text{size } [(N+P-1) \times P]$$

3.3 Adaptive Algorithm

Vector **a** (3.9a) contains the warped common AR coefficients that are mapped back to the original domain to use them as the prefilter's fixed polynomial (3.10) as

$$A_c(q) = 1 - a_c(1)q^{-1} - ... - a_c(P)q^{-P} \tag{3.10a}$$

$$\text{or} \quad A_c(q) = \mathbf{a}_c^T \mathbf{q} \quad \text{in vector notation} \tag{3.10b}$$

The prefiltering is done on either the adaptive filter signal or the loudspeaker signal, as Figure 3.1 shows. For every microphone the numerator coefficients (3.11) are adapted using the well-known normalized least mean squares (NLMS) algorithm [6] as

$$B_i(q,t) = b_i(0,t) + b_i(1,t)q^{-1} + ... + b_i(Q,t)q^{-Q} \tag{3.11a}$$

$$\text{or} \quad B_i(q,t) = \mathbf{b}_i^T(t)\mathbf{q} \quad \text{in vector notation} \tag{3.11b}$$

NLMS is a gradient-based algorithm whose basic equations are given by

$$e_i(t) = y_i(t) - \hat{y}_i(t) \tag{3.12a}$$

$$= y_i(t) - \hat{\mathbf{b}}_i(t)\mathbf{u}^T(t) \tag{3.12b}$$

$$\hat{\mathbf{b}}_i^T(t+1) = \hat{\mathbf{b}}_i^T(t) + \mu \frac{1}{\mathbf{u}(t)\mathbf{u}^T(t)} \mathbf{u}(t)^T e_i(t) \tag{3.12c}$$

where $i = 1, ..., M$, the vector $\mathbf{u}(t) = [u(t), u(t-1), ..., u(t-Q)]$ is the input to the adaptive filter, $y_i(t)$ are the microphone signals, $e_i(t)$ are the error signals, $\hat{y}_i(t)$ are the estimated echo signals and μ is the step size. In the filter coefficients update formula (3.12c) and in (3.12b), the vectors $\hat{\mathbf{b}}_i^T(t)$ are the adaptive filter coefficients of each channel. Signals $u(t)$ and $y_i(t)$ will depend on whether the prefilter is in the adaptive filter signal path or in the loudspeaker signal path as follows.

- Prefilter in the adaptive filter signal path

$$u(t) = \frac{1}{A_c(q)} x(t) \tag{3.13}$$

$$= x(t) - a_c(1)u(t-1) - ... - a_c(P)u(t-P) \tag{3.14}$$

$$y_i(t) = H_i(q,t)x(t) \tag{3.15}$$

- Prefilter in the loudspeaker signal path

$$u(t) = x(t) \tag{3.16}$$

$$y_i(t) = A_c(q)H_i(q,t)x(t) \tag{3.17}$$

$$\approx A_c(q)\frac{B_i(q,t)}{A_i(q)}x(t) \approx B_i(q,t)x(t) \tag{3.18}$$

$\lambda = 0.0$	16.6
$\lambda = 0.7$	28.0

Table 3.1: Attenuation (dB) of AEC_1. Change of warping parameter λ.

3.4 Simulation Results

Matlab computer simulations are performed at $f_s = 16$ kHz. Five room impulse responses, (h_i with $i = 1, ..., 5$) of length $N = 2000$ samples are measured in a rectangular room of about $5 \times 3 \times 3$ m. In every simulation, $\mu = 1$ as it offered the best results. The input signal is speech (i.e. female voice) recorded at $f_s = .6$ kHz of 6.7 s of duration (i.e., length $L = 1340876$ samples). The warping parameter $\lambda = 0.7$ is found to be optimal. The performance measures are: *Attenuation*,

$$Attenuation_i = 10 \log_{10} \frac{\sum_{t=0}^{L} y_i^2(t)}{\sum_{t=0}^{L} e_i^2(t)} \quad (dB) \tag{3.19}$$

which is a scalar that measures the difference in dB between the power of the L samples length error and microphone signals; and *Echo Return Loss Enhancement* ($ERLE$),

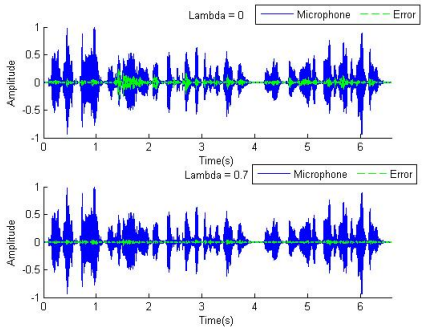
$$ERLE_i(n) = 10 \log_{10} \frac{\sum_{k=1}^{p} y_i^2\big((n-1)p + k\big)}{\sum_{k=1}^{p} e_i^2\big((n-1)p + k\big)} \quad (dB) \tag{3.20}$$

for $n = 1, ..., \frac{L}{p}$, which is (3.19) averaged over time frames of length p. The order of numerator Q and denominator P are the same in every simulation.

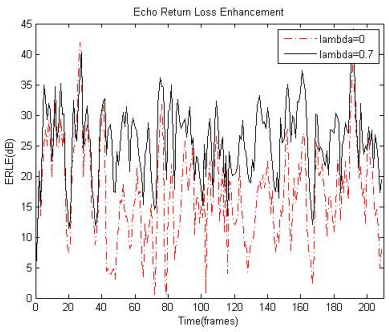
We define an echo canceler as AEC_i with $i = 1, ..., 5$. The performance in terms of $Attenuation_i$ and $ERLE_i$ of different acoustic echo cancelers is calculated in two different scenarios. In the first scenario, the AR coefficients of the prefilter are calculated from a single impulse response. In the second scenario, the AR coefficients of the prefilter are calculated using the whole set of impulse responses, thus defining the common acoustical poles of the system.

3.4.1 Comparison of ERLE with and without warping

The effect of warping an impulse response previous to calculating the AR coefficients is shown in this section. Figure 3.4 shows the differences in the AEC performance when using different values of warping parameter λ. The optimal value $\lambda = 0.7$ was applied and compared with $\lambda = 0$ which means that no warping is done to the impulse response previous to calculating the AR coefficients. The WLP (AR) order is $P = 200$ and the prefilter was then applied in the adaptive filter signal path. In such case, a pole-zero model of the RTF, with



(a) Error signal (e_1) evolution with different λ



(b) ERLE of $\text{AEC}_1$with different λ

Figure 3.4: Prefilter on the adaptive filter signal path with WLP (AR) order $P = 200$ with optimal λ=0.7 and λ=0 which means no warping.

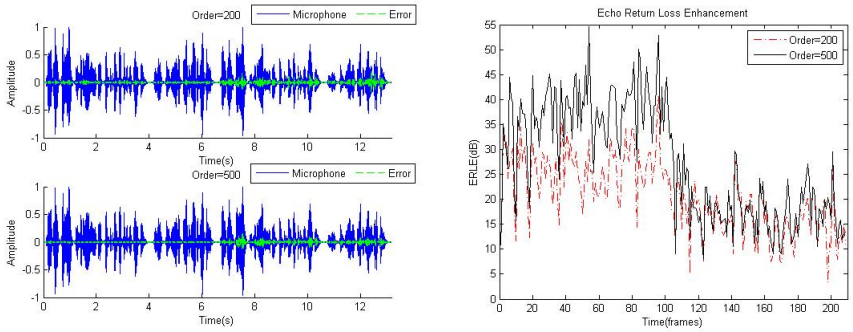
fixed denominator and variable numerator, is achieved in the AEC. The AR coefficients are extracted from impulse response h_1. Figure 3.4(a) shows the time evolution (L samples) of the microphone y_1 in blue and error signal e_1 in green and Figure 3.4(b) shows the $ERLE_1$. Table 3.1 shows the values of the attenuation of the AEC_1 where it is seen that, with warping, an improvement of more than 11 dB can be achieved.

3.4.2 Prefilter in the adaptive signal path

This section shows the performance of the AEC_i, with $i = 1, ..., 5$, when the prefilter is applied to the adaptive filter signal. In such case, a pole-zero model of the RTF, with fixed denominator and variable numerator, is achieved in the AECs. The warping parameter is set to $\lambda = 0.7$. The AR coefficients are calculated with two different WLP (AR) orders, i.e., $P = 200$ and $P = 500$.

- AR coefficients are calculated from a single impulse response (h_1)

Figure 3.5(a) and Figure 3.5(b) show the time evolution of the microphone and error signal and ERLE respectively, of both the AEC_1 (from $0 - 100$) and AEC_4 (from $100 - 200$). Table 3.2 (left) shows the values of the attenuation achieved by AEC_1 and AEC_4 where the difference between them can be of 11.7 dB ($P = 200$) and 19.5 dB ($P = 500$). The idea here is to show that the performance of AEC_4 when the AR coefficients have been calculated from h_1 is really bad. Later on, we show that with common acoustical poles the performance increases considerably.



(a) Error signal evolution with different WLP (AR) orders

(b) ERLE with different WLP (AR) orders

Figure 3.5: Prefilter in the adaptive filter signal path. AR coefficients are calculated from impulse response h_1 with optimal λ=0.7 and $P = 200$ and $P = 500$. Performance of AEC_1 and AEC_4 are shown consecutively i.e., from $0 - 100$ corresponds to AEC_1 and from $100 - 200$ to AEC_4.

- Common-acoustical-pole AR coefficients

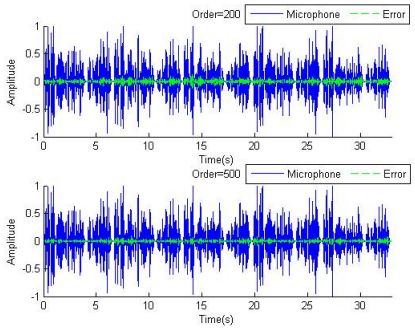
In this case, the fixed filter coefficients are calculated from the whole set of impulse responses using (3.8). Figure 3.6(a) and Figure 3.6(b) show the time evolution of the microphone and error signal and the ERLE respectively, of every AEC_i, with $i = 1, ..5$. Table 3.3 (top) shows that the difference among the *Attenuation* values with respect to Figure 3.5 is highly reduced with satisfactory individual results, i.e., average *Attenuation* = 21.6 and 25.2 dB with $P = 200$ and 500, respectively.

3.4.3 Prefilter in the loudspeaker signal path

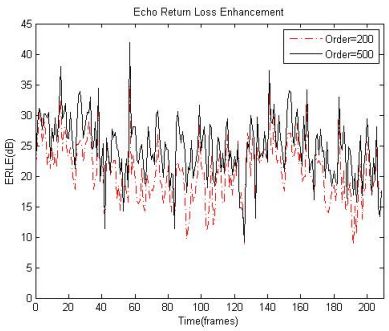
This section shows the performance of the AEC_i, with $i = 1, ..., 5$, when the prefilter is applied in the loudspeaker signal path. In such case, a dereverberating effect is achieved by canceling the main resonances. The warping parameter is set to $\lambda = 0.7$. The AR coefficients are calculated with two WLP (AR) orders,

LPC (h_1) (Adap. sig.)	AEC_1	AEC_4	LPC (h_1) (Loud. sig.)	AEC_1	AEC_4
$P = 200$	28.0	16.3	$P = 200$	19.4	14.0
$P = 500$	37.8	18.3	$P = 500$	29.9	14.2

Table 3.2: Attenuation (dB). AR Coefficients obtained from h_1. (Left) Prefilter in the adaptive filter signal path. (Right) Prefilter in loudspeaker signal path.



(a) Error signal evolution with different WLP (AR) orders



(b) ERLE with different WLP (AR) orders

Figure 3.6: Prefilter in the adaptive filter signal path. AR coefficients are calculated from common acoustical poles with optimal λ=0.7 and $P = 200$ and $P = 500$. Performance of every AEC_i are shown consecutively.

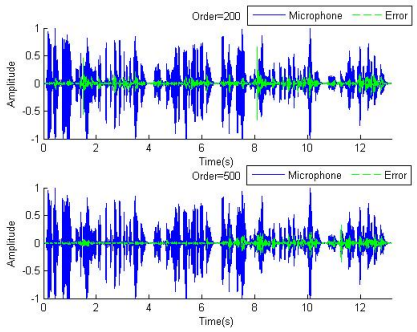
LPC_{common} (Adap. sig.)	AEC_1	AEC_2	AEC_3	AEC_4	AEC_5
$P = 200$	24.8	20.3	19.4	24.2	19.4
$P = 500$	27.9	24.3	24.4	26.0	22.5
LPC_{common} (Loud. sig.)					
$P = 200$	15.4	15.7	13.8	15.2	13.4
$P = 500$	24.0	19.9	19.7	22.5	18.3
All-zero (No prefilter)					
$Q = 2000$	36.4	31.6	33.3	28.7	32.1

Table 3.3: Attenuation(dB). AR coefficients obtained from common acoustical poles. Top) Prefilter in the Adaptive filter signal path. Middle) Prefilter in the loudspeaker signal path. Bottom) No prefilter.

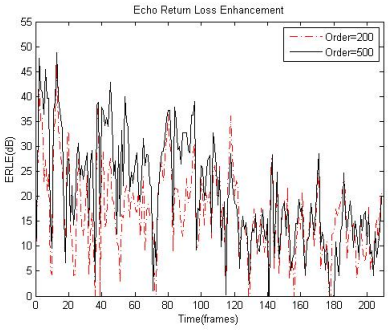
i.e., $P = 200$ and $P = 500$.

- AR coefficients calculated from one impulse response h_1

Figure 3.7(a) and Figure 3.7(b) show the time evolution of the microphone and error signal and ERLE respectively, of both the AEC_1 and AEC_4. Table 3.2.(Right) shows the values of the attenuation achieved by AEC_1 and AEC_4 where the difference between them can be of 5.4 dB ($P = 200$) and 15.7 dB ($P = 500$).



(a) Error signal evolution with different WLP (AR) orders (200 Top, 500 Bottom)



(b) ERLE with different WLP (AR) orders

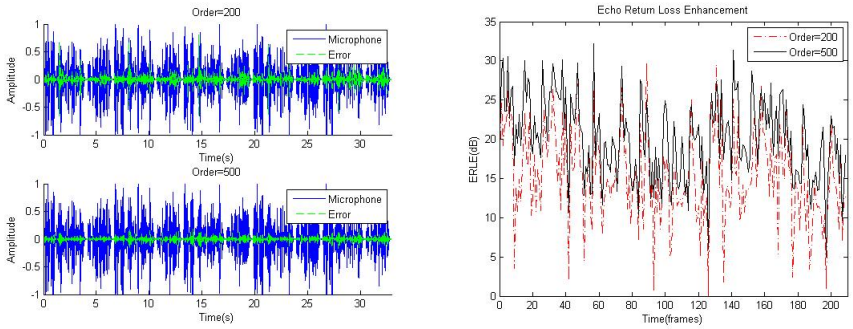
Figure 3.7: Prefilter in the loudspeaker signal path. AR coeffients are calculated from impulse response h_1 with optimal λ=0.7 and $P = 200$ and $P = 500$. Performance of AEC_1 and AEC_4 are shown consecutively.

- Common-acoustical-pole AR coefficients

In this case, the fixed filter coefficients are calculated from the whole set of impulse responses using (3.8). Figure 3.8(a) and Figure 3.8(b) show the time evolution of the microphone and error signal and the ERLE respectively, of every AEC_i, with $i = 1, ..5$. Table 3.3.Middle) shows that the difference among the *Attenuation* values is reduced with satisfactory individual results only with $P = 500$, i.e., average *Attenuation* = 14.2 and 19.9 dB with $P = 200$ and 500 respectively. Table 3.3 (bottom) shows the values of the *Attenuation* in the case no prefilter is done (i.e., all-zero case) and order $Q = 2000$ which is the total length of the impulse response. It can be seen that although the achieved attenuation is higher compared with our proposed model, the number of coefficients increases dramatically. Assuming that NLMS requires $2Q + P$ multiply-add operations for each update (3.12a)-(3.12c) [5], the number of parameters in the all-zero case is $M \cdot 2Q = 5 \cdot 4000 = 20000$ operations whereas with fixed prefiltering, $M \cdot 2Q + P = 5 \cdot 400 + 200 = 2200$ (i.e., 91% operations saving) or $M \cdot 2Q + P = 5 \cdot 1000 + 500 = 5500$ (i.e., 36% operations saving) multiply-add operations only.

3.5 Conclusion

In this chapter, we have proposed a model for multi-microphone AEC which employs the idea of common-acoustical-poles and zero modeling of RTFs using warped linear prediction of the impulse responses. Common acoustical poles



(a) Error signal evolution with different WLP (AR) orders

(b) ERLE with different WLP (AR) orders

Figure 3.8: Prefilter in the loudspeaker signal path. The AR coeffients are calculated from common acoustical poles with optimal λ=0.7 and $P = 200$ and $P = 500$. Performance of every AEC_i are shown consecutively.

are calculated off-line from a set of measured impulse responses. By frequency-warping the measured impulse responses one is able to focus the computational resources in a frequency region of interest. In RTFs the predominant spectral peaks are located in the low frequency region. Warping allows us to use the modeling effort in that frequency region to obtain better modeling results. This leads to a higher echo reduction for the same number of filter coefficients. Moreover these predominant spectral peaks are common to every RTF which allows us to have a fixed common denominator polynomial for every channel. Hence only the numerator polynomial has to be estimated recursively for adapting to changes in the room. This approach allows the number of numerator coefficients to decrease by up to one order of magnitude for each microphone compared with all-zero modeling to get satisfactory results (about 22 dB of echo attenuation). Better results are obtained in the case of prefiltering in the adaptive filter signal path. This is due to the IIR nature of this configuration which leads to a better modeling capability.

Bibliography

[1] A. Härmä, M. Karjalainen, L. Savioja, V. Välimäki, U. K. Laine, and J. Huopaniemi, "Frequency-warped signal processing for audio applications", *J. Audio Eng. Soc.*, vol. 48, no. 11, pp. 1011-1031, 2000.

[2] T. Paatero and M. Karjalainen, "Kautz Filters and Generalized Frequency

Resolution - Theory and Audio Applications", *J. Audio Eng. Soc.*,vol. 51, no. 1/2, pp. 27-44, January/February 2003.

[3] H. W. Strube, "Linear prediction on a warped frequency scale", *J. Acoust. Soc. Amer.*, vol. 68, no. 4, pp. 1071-1076, Oct. 1980.

[4] T. van Waterschoot and M. Moonen, "Adaptive feedback cancellation for audio signals using a warped all-pole near-end model," in *Proc. IEEE Int. Conf. on Acoust. Speech Signal Process. (ICASSP '08)*, Las Vegas, Nevada, USA, Apr. 2008, pp. 269-272.

[5] P. S. R. Diniz, *Adaptive Filtering: Algorithms and Practical Implementations*. Springer, Boston, MA., 2008.

[6] S. Haykin, *Adaptive Filter Theory*. Prentice Hall, 2001.

[7] Y. Haneda, S. Makino and Y. Kaneda, "Multiple-Point Equalization of Room Transfer Functions by Using Common Acoustical Poles," *IEEE Trans. Speech Audio Process.*, vol. 5, no. 4, pp. 325-333, July 1997.

[8] Y. Haneda, S. Makino and Y. Kaneda, "Common Acoustical Pole and Zero Modeling of Room Transfer Function," *IEEE Trans. Speech Audio Process.*, vol. 2, no. 2, pp. 320-328, Apr. 1994.

Part III

Linear Adaptive Filtering

Chapter 4

Regularized adaptive notch filters for acoustic howling suppression

Regularized adaptive notch filters for acoustic howling suppression

Jose Manuel Gil-Cacho, Toon van Waterschoot, Marc Moonen and Søren Holdt Jensen

Published in Proc. 17th European Signal Processing Conference 2009 (EUSIPCO'09), Glasgow, UK, Aug. 2009.

Contributions of first author

- literature study
- co-development of the RANF algorithm
- design of simulation scenarios
- software implementation and computer simulations
- co-interpretation of simulation results
- text redaction and editing

Abstract

In this chapter, a method for the suppression of acoustic howling is developed, based on adaptive notch filters (ANF) with regularization (RANF). The method features three RANFs working in parallel to achieve frequency tracking, howling detection and suppression. The ANF-based approach to howling suppression introduces minimal processing delay and minimal complexity, in contrast to non-parametric block-based methods featuring a non-parametric frequency analysis. Compared to existing ANF-based howling suppression methods, the proposed method allows for a more advanced howling detection such that tonal components in the source signal are not affected. The RANFs proposed in this chapter are implemented in direct form and are updated using a gradient-descent-type algorithm. Results show that, under certain conditions, the level of suppression and sound quality are similar to what is obtained with block-based methods.

4.1 Introduction

In some applications, such as for public address (PA) systems, notch-filter-based howling suppression (NHS) methods are widely utilized [3]. The NHS method relies in the use of notch filters in the forward path so as to suppress frequency components that produce acoustic howling [3]. In this chapter, we will focus on the NHS method. NHS methods perform a frequency analysis, howling detection and howling suppression. We may tackle these actions using either block-based techniques, i.e., using the Fast Fourier Transform (FFT) or adaptive notch filters (ANF). Howling detection is very difficult in ANF-based methods since no power spectrum information is available [4], [5]. Block-based methods, on the other hand, accomplish howling detection based on power spectra amplitude information. However, due to the FFT operations involved, block-based methods are more complex and require more processing delay than ANF-based methods. The method proposed in this chapter combines the advantages of keeping the complexity small while having an improved howling detection mechanism, by including regularization and multiple parallel ANFs.

The chapter is organized as follows. Section 4.2 reviews with notch-filter-based suppression methods. The main structure of both block-based and ANF-based methods is shown. Moreover, the equations of the direct-form ANF algorithm are given. Section 4.3 presents the proposed method, showing its operation and its block structure. Simulation results are presented in Section 4.4 and conclusions are drawn in Section 4.5.

4.2 Notch-filter-based howling suppression

To eliminate narrowband or sinusoidal signals from a broadband signal (e.g. a noise, audio or speech signal) notch filters are often used. There exist different types of notch filters, e.g. FIR or IIR filters. We will focus on second-order IIR filters with constrained poles and zeros [6]. The constraint is that the zeros lie on the unit circle and the poles will lie in the same radial direction but with a pole radius within $0 << r < 1$. The transfer function of such a notch filter, centered at a radial notch frequency $\omega_0 = 2\pi\,(f_0/f_s)$, with f_s the sampling frequency, is given by.

$$H\,(z) = \frac{1 - 2\cos(w_0)z^{-1} + z^{-2}}{1 - 2r\cos(w_0)z^{-1} + r^2 z^{-2}} \tag{4.1}$$

When using notch filters for howling suppression, one aims to have maximum attenuation at the howling frequency but a minimal effect on surrounding frequencies so as to avoid distortion of the acoustic input signal. This can be achieved by employing a very narrow bandwidth, i.e., a pole radius close to unity. Therefore, the notch filter performance strongly depends on the estimation of the howling frequency in the sense that, if very narrowband filters are to be used then very accurate frequency estimates are needed. Otherwise, there is a high risk of suppressing a signal component in the close vicinity of, but not exactly at the actual howling frequency. If the frequency estimation is known to have poor accuracy, a larger bandwidth must be used in order to ensure that the howling frequency is sufficiently attenuated. Suppressing the signal in a wider frequency band, on the other hand, leads to more distortion.

4.2.1 Non-parametric frequency estimation

Non-parametric frequency estimation methods, in which a block-based estimation is performed using the FFT [3], can yield accurate frequency estimates only when long signal blocks are used. This in turn implies that a large processing delay and a high computing power are required. The choice of the block size is hence a trade-off between processing delay and computational complexity on the one hand, and frequency estimation accuracy on the other hand. Another issue with block-based methods is that for rapid changes in the howling frequency, proper frequency tracking is insufficient if the block is too long [7].

NHS methods based on non-parametric frequency estimation are two-stage methods, where howling estimation/detection and suppression are performed separately, see Figure 4.1. As explained before, the suppression block consists of a notch filter, as in (4.1), or of a bank of notch filters in which several notch filters are cascaded. Each of those notch filters can be tuned to a different howling frequency previously estimated in the estimation/detection block. There

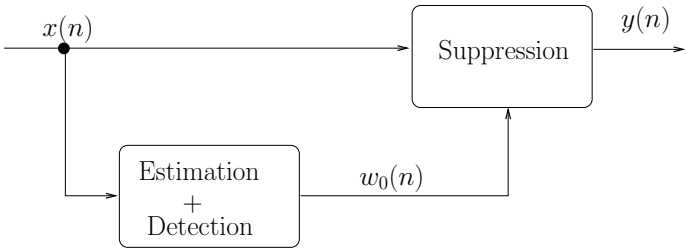


Figure 4.1: Detection and suppression block scheme in a typical two-stage FFT-based NHS system.

are different approaches to discriminate whether a tonal component is either due to undesired acoustic feedback or it is a desired source signal component. An extensive comparison of these approaches is given in [3].

4.2.2 Adaptive Notch filters

Adaptive notch filters (ANFs) perform a parametric frequency estimation, and allow for a simultaneous howling estimation/detection and suppression as shown in Figure 4.2.

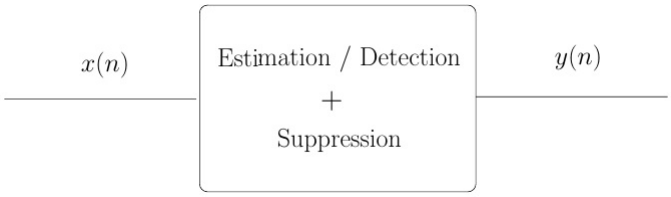


Figure 4.2: Simultaneous detection and suppression in a one-stage ANF-based system.

The advantage of ANF-based methods over FFT-based methods is fourfold: 1) the required processing delay is minimum since the ANF-based method is sample-based, 2) ANFs are able to track changes in the howling frequency in a sample-by-sample basis, 3) avoiding FFT operations strongly reduces computational complexity, and 4) the achievable frequency estimation accuracy is generally high for a limited amount of data. Different IIR-ANF implementations have been proposed in the literature, both with a Gauss-Newton-type update [8], [9], [6], and with a gradient-descent-type update [10]. In the sequel we will use a gradient-descent implementation as it is of minimal complexity and only n parameters have to be estimated, where n is the number of sinusoids. The notch filter transfer function (4.1) can be rewritten in a slightly

different form which is more suitable for coefficient updating, i.e.

$$H(q) = \frac{1 - a(n)q^{-1} + q^{-2}}{1 - a(n)rq^{-1} + r^2q^{-2}} \tag{4.2}$$

where q denotes the discrete time shift operator, i.e., $q^{-k}u(n) = u(n-k)$. The parameter $a(n)$ defines the instantaneous frequency $w_0(n)$, i.e.

$$w_0(n) = \arccos\left[\frac{a(n)}{2}\right] \tag{4.3}$$

The parameter $a(n)$ is allowed to take values bounded by $-2 < a(n) < 2$. The basic idea underlying ANF-based frequency estimation and howling suppression consists in feeding the signal into the ANF and perform a minimization w.r.t. $a(n)$, of the mean square error (MSE) of the notch filter output signal $y(n)$

$$\min_{a(n)} E[y(n)^2] \tag{4.4}$$

This will cause the notch to be centered at the frequency corresponding to the signal's narrowband or sinusoidal component. The gradient descent algorithm will adjust the coefficient $a(n)$ in the negative gradient direction $\nabla_a(n)$, until a local minimum in the cost function is attained. The gradient descent algorithm for the direct-form ANF is given by equations (4.5)-(4.9) [10]. Here the ANF input and output signals are denoted by $x(n)$ and $y(n)$ respectively, and the filter states are denoted by $u(n), u(n-1), t(n), t(n-1)$. The step-size of the gradient descent algorithm is denoted by μ.

$$\begin{aligned}
t(n+1) &= u(n) - ra(n)t(n) - r^2t(n-1) && (4.5)\\
\nabla_a(n) &= t(n+1) - rt(n-1) && (4.6)\\
u(n+1) &= x(n) - ra(n)u(n) - r^2u(n-1) && (4.7)\\
y(n) &= u(n+1) + a(n)u(n) + u(n-1) && (4.8)\\
a(n+1) &= a(n) - \mu y(n)\nabla_a(n) && (4.9)
\end{aligned}$$

4.3 Regularized Adaptive Notch Filters

The proposed NHS method employs three regularized ANFs (RANF) that run in parallel and share one decision block, see Figure 4.4. Each RANF is regularized with a term λ_i, which is chosen to have a different value for $i = 1, 2, 3$. The regularized ANF cost function is given as

$$\min_{a_i(n)} E[y_i^2(n)] + \lambda_i a_i(n)^2 \qquad i = 1, 2, 3. \tag{4.10}$$

This results in a modified gradient descent coefficient update, corresponding to a so-called Leaky LMS [11]:

$$a_i(n+1) = a_i(n) - \mu \left\{ y_i(n) \nabla_{a_i}(n) + \lambda_i a_i(n) \right\} \tag{4.11}$$

The effect of the regularization term is negligible when howling is present in the signal. This is so because the term in (4.11) with the gradient search direction $\nabla_{a_i}(n)$, i.e. $y_i(n)\nabla_{a_i}(n)$, is significantly larger than the term $\lambda_i a_i(n)$. Conversely, when howling is not present, the gradient search direction $\nabla_{a_i}(n)$ tends to zero and so the coefficient update formula approximately equals $a_i(n+1) = a_i(n) - \mu\lambda_i a_i(n) = (1 - \mu\lambda_i)a_i(n)$. The effect of this is that the regularization term is in fact penalizing the estimates $a_i(n)$. Moreover, depending on the sign of λ_i, the regularization term will introduce a leakage or an accumulation effect on the coefficient estimate. Conventionally regularized algorithms ,i.e., with a positive λ_i, produce estimates that are biased towards zero. By having negative λ_i, the proposed RANF algorithm also produces estimates that are "biased towards infinity". This is indeed what we observe in Figure 4.3: Whenever howling does not occur the RANF coefficients diverge either to their upper bound if λ_i is negative or to zero if λ_i is positive.

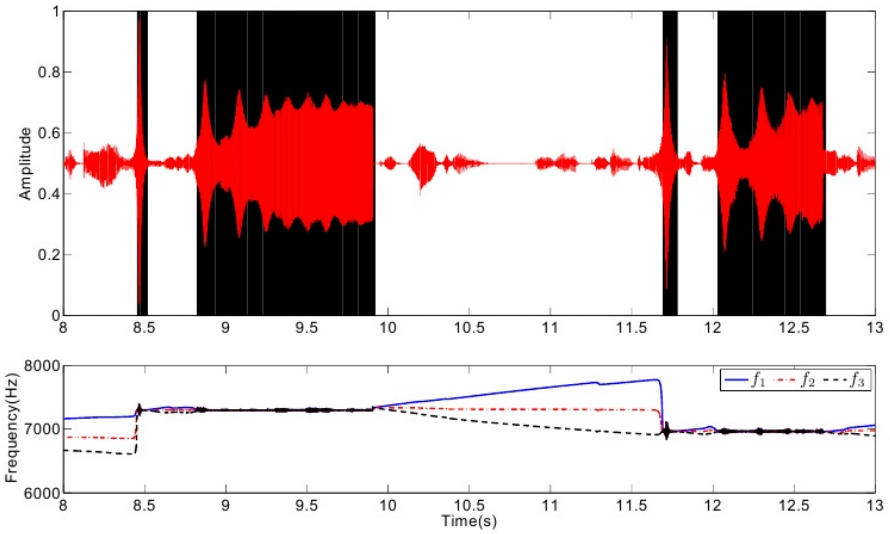


Figure 4.3: Upper: representation of the time-domain input signal $x(n)$ (upper) where the framed signal fragments correspond to howling segment. Lower: the RANF frequency estimates f_i, with $i = 1, 2, 3$.

Figure 4.3 shows an example of the evolution of the frequency tracking as a function of time with the proposed method. When howling appears in the signal, the three coefficients $a_i(n)$ converge to the same value. The boxes frame the part of the signal where howling is present. When howling disappears, the three coefficients $a_i(n)$ move away from each other as a result of having different regularization parameters λ_i.

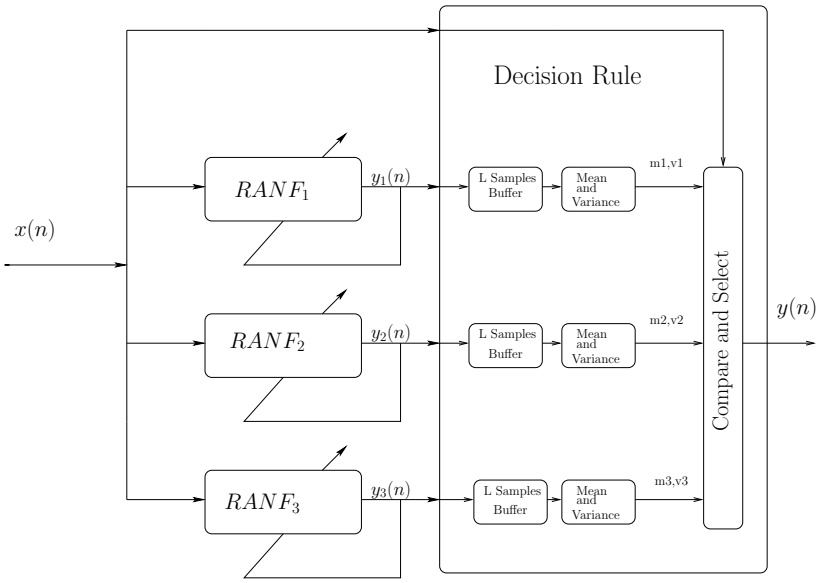


Figure 4.4: Block diagram of the proposed RANF-based NHS method.

Figure 4.4 shows a block diagram of the proposed RANF-based NHS method. The signal $x(n)$ is input into the three RANF blocks which will track a tonal component in the signal. The *Decision Rule* block monitors the coefficients $a_i(n)$ of the three RANF blocks. After L samples are buffered, where L is a small number so as to have minimum decision delay the mean and the variance of a block of L samples is calculated and compared for the three RANF blocks. If the difference between two mean values is less than a fixed threshold T, in Hz, then howling is assumed to be present. Essentially, this means that howling is detected when at least two RANF coefficients have converged to the same frequency. In this case, the output signal $y(n)$ is assigned to be one of the three RANF block output signals $y(n)_{1,2,3}$, depending on the frequency estimation variance. The smaller the variance the more reliable the result is assumed to be. Similarly, if the differences in mean value for the three RANF blocks are larger than the fixed threshold, then no howling is assumed in the signal and the output is generated directly from the input.

The values of λ_i are chosen such that the difference Δf (Hz), in a time pe-

riod of M samples, between two coefficients $a_i(n)$ corresponds to a given fixed threshold (Hz). We will set $\lambda_1 = \lambda$, $\lambda_2 = 0$ and $\lambda_3 = -\lambda$, where

$$\lambda = \frac{1}{\mu}\left[\sqrt[L]{\frac{1}{cos(\Delta w)}} - 1\right] \tag{4.12}$$

We have derived (4.12) based on the "small-angle approximation" (see Appendix 4.A) , i.e., Δf should be small compared to the sampling frequency f_s, a trigonometric relationship between the regularization parameter λ_i, and the desired divergence rate after a howling occurrence.

4.4 Results

In this section, the performance of the proposed method is evaluated for two types of signals. These signals, were generated in Matlab from clean speech and music signals. The feedback paths were synthetically generated, using an exponentially damped tone at a particular frequency, as a synthetic feedback path impulse response. The frequency and duration of the feedback path were drawn from pre-defined distributions in order to simulate changing feedback conditions (i.e., dynamic feedback path). A pre-specified loop gain was obtained by changing the forward gain appropriately. The frequency range and maximum loop gain were chosen for two test scenarios (i.e., scenarios 'a' and 'b') in order to change, besides the howling frequency, the speed at which the howling appears, the exponential slope of the increasing howling amplitude, the howling duration and how frequently howling appears in the signal, see Figure 4.5(a), 4.6(a) and Figure 4.7(a), 4.8(a) . Therefore, for each clean signal two types of howling signals were generated, namely $Speech_a$, $Speech_b$, $Music_a$ and $Music_b$.

The original signals were a speech signal (i.e., an English-speaking female voice) and a music signal (i.e. a fragment of a song) both sampled at 20 kHz. The system parameters were set to $\lambda_1 = +2.6820e-004$, $\lambda_2 = 0$, $\lambda_3 = -2.6820e-004$, $\mu = 0.023$, $r = 0.85$, $T = 5$ Hz, $L = 50$ samples.

Four objective performance measures will be used in this section, namely maximum and minimum attenuation in dB ($Att_{\max}$ and $Att_{\min}$ respectively) and maximum and mean frequency-weighted log-spectral signal distortion ($SD_{\max}$ and SD_{mean} respectively). The Att (4.13) is calculated comparing the power spectrum of the original signal without howling with the signal after howling suppression, i.e.

$$Att(f) = 10\log_{10}\frac{S_y(f)}{S_x(f)} \tag{4.13}$$

where $S_y(f)$ and $S_x(f)$ denote the short-term power spectra of the signal after suppression and the original signal around the howling frequency f, respectively. $Att_{\max}$ is the difference in dB between the frequency component with the smallest power in the signal after suppression and the original signal and so, $Att_{\min}$ is the difference with respect to the highest frequency component. Ideally, $Att_{\max}$ should be zero dB which means that perfect, distortion-free, howling suppression is achieved.

The SD is a measure of sound quality and objectively measures the distortion produced not only by applying notch filters to the signal but also due to howling. It was proposed in [2] and is given as

$$SD(t) = \sqrt{\int_0^{f_s/2} w_{\mathrm{ERB}}(f) \left(10 \log_{10} \frac{S_y(f)}{S_x(f)}\right)^2 df} \tag{4.14}$$

where $w_{\mathrm{ERB}}(f)$ is a weighting function that gives weights to each auditory critical band within the Nyquist interval, following Table II of the ANSI S3.5-1997 standard. The integration in (4.14) is approximated by a summation over the critical frequency bands. Both the mean and maximum values SD_{mean} $SD_{\max}$, will be used.

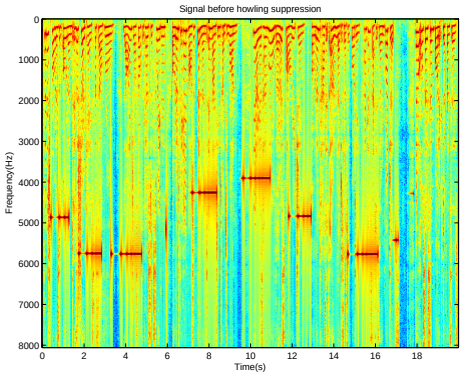
4.4.1 Speech signal

Figure 4.5 and 4.6 show the spectrogram of the speech signal before and after howling suppression by means of the proposed method. In Figure 4.5(a) and Figure 4.6(a), the howling speech signal is presented to show the howling frequency range and time evolution. From Figure 4.5(b) it is clearly observed that suppression is performed equally well over time and frequency. However, in Figure 4.6(b) we can see that in the frequency range below 1.5 kHz, no suppression is accomplished.

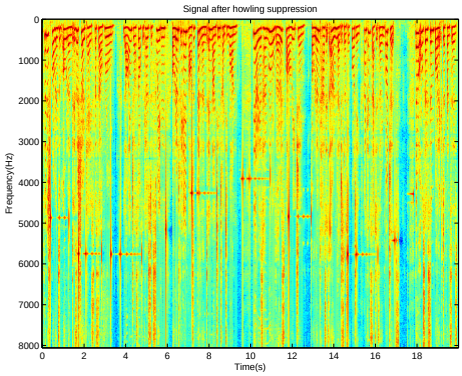
4.4.2 Music signal

The same procedure is followed in this case for a music signal. Figure 4.7 and 4.8 show the spectrograms of the music signal before and after howling suppression. The same observation as in the speech simulation can be made in this case.

In Table 4.1, the corresponding performance measures are shown. It can again be observed that scenario 'b' is more problematic in terms of both maximum and minimum attenuation (i.e., howling suppression) and spectral distortion. For frequencies below 1500 Hz, the method not only is unable to suppress the howling but also is further distorting the signal in a wider frequency range due to false howling detection. This fact has been noted in [10] where it is pointed out that direct-form adaptive notch filters are not necessarily stable when the



(a) $Speech_a$ before suppression



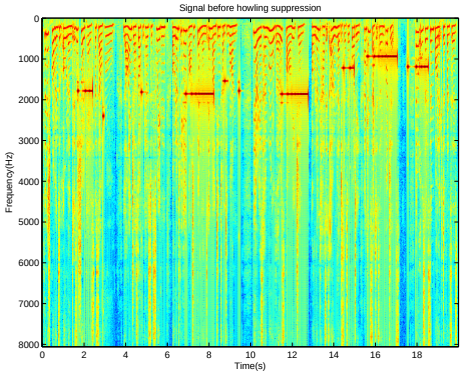
(b) $Speech_a$ after suppression

Figure 4.5: Speech$_a$ signal before and after howling suppression. The frequency range, gain and time evolution of howling is generated differently in 'a' and 'b' scenarios.

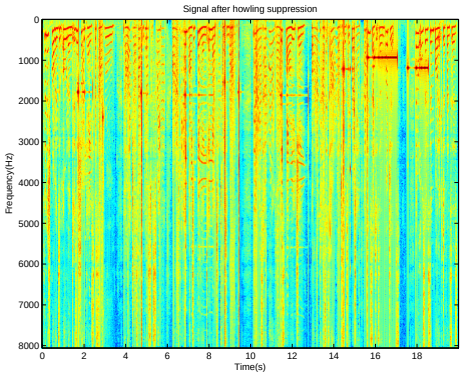
Signal	SD_{mean}	SD_{max}	Att_{max} dB	Att_{min} dB
Speech$_{\mathbf{a}}$	1.66	14.78	5	6
Speech$_{\mathbf{b}}$	4.32	28.83	5	40
Music$_{\mathbf{a}}$	1.05	13.24	3	6
Music$_{\mathbf{b}}$	3.61	17.83	1	40

Table 4.1: System performance for given howling signals.

notch frequency approaches its extreme values, 0 and $f_s/2$. Therefore, when the signal contains howling in the neighborhood of these frequencies, the method cannot be used to suppress it.

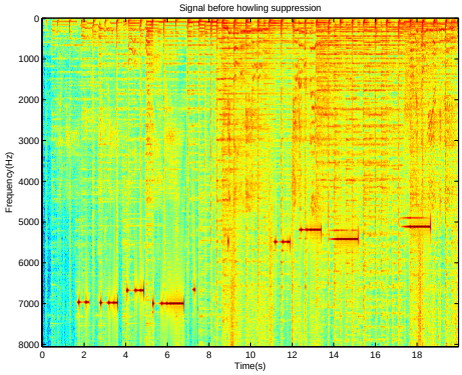


(a) $Speech_b$ before suppression

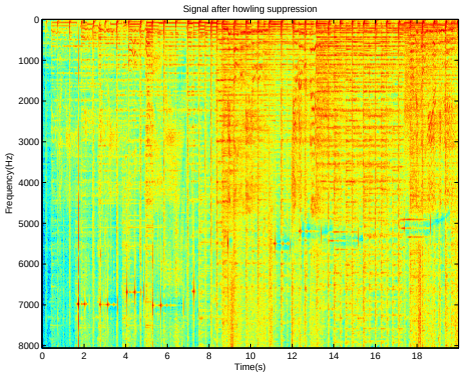


(b) $Speech_b$ after suppression

Figure 4.6: Speech$_b$ signal before and after howling suppression. The frequency range, gain and time evolution of howling is generated differently in 'a' and 'b' scenarios.

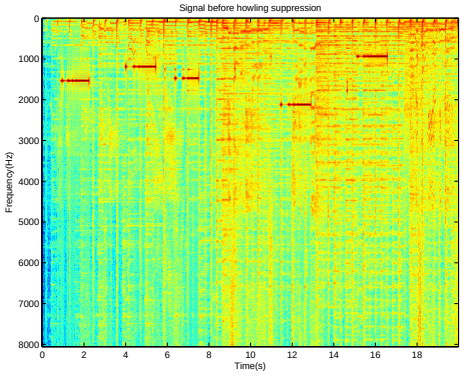


(a) $Music_a$ before suppression

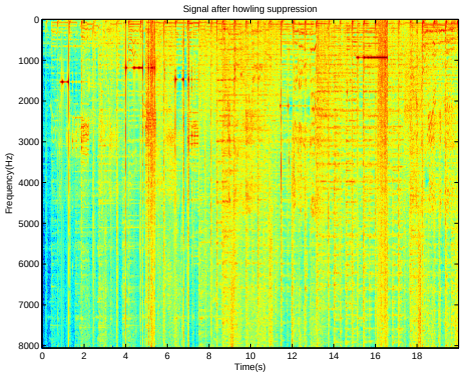


(b) $Music_a$ after suppression

Figure 4.7: Music$_a$ signal before and after howling suppression. The frequency range, gain and time evolution of howling is generated differently in 'a' and 'b' scenarios.



(a) $Music_b$ before suppression



(b) $Music_b$ after suppression

Figure 4.8: Music$_b$ signal before and after howling suppression. The frequency range, gain and time evolution of howling is generated differently in 'a' and 'b' scenarios.

4.5 Conclusion

In this chapter, a new method for acoustic howling suppression has been presented. It is based on adaptive notch filters that include a regularization term (RANF). The proposed method has the advantage over non-parametric block-based methods, e.g., FFT-based methods, that it requires a minimum processing delay and has a small computational complexity. Simulations show that the method is able to suppress and track howling frequencies in situations where the howling frequency is confined to mid-range frequencies. Compared to existing ANF-based methods, a more accurate howling detection can be achieved.

Appendix

4.A RANF regularization parameter λ

We start from the fact that

$$a(n+1) = (1+\mu\lambda)a(n) \tag{4.15}$$

$$a(n+L) = (1+\mu\lambda)^L a(n) \tag{4.16}$$

and define

$$\Delta f = f(n+L) - f(n) \tag{4.17}$$

$$\Delta w = w(n+L) - w(n) \tag{4.18}$$

$$\Delta w = \frac{2\pi\Delta f}{f_s}. \tag{4.19}$$

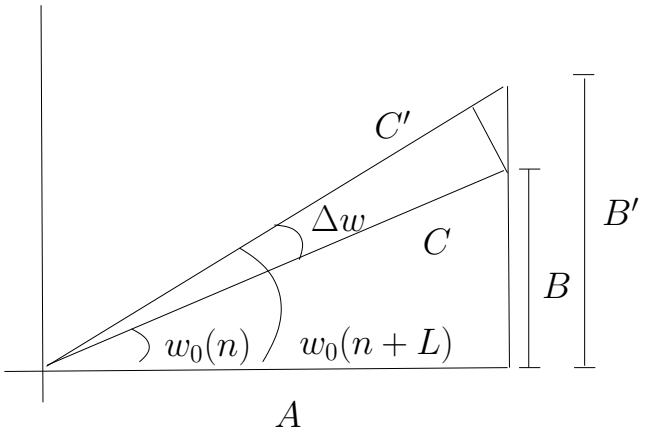


Figure 4.9: Small-angle approximation of Δw.

From (4.3) we obtain $a(n) = -2\cos(w(n))$. Next, we define the ratio R of the adaptive coefficients at different time instants as

$$R = \frac{a(n+L)}{a(n)} = \frac{2\cos\left(w[n+L]\right)}{2\cos\left(w[n]\right)} \tag{4.20}$$

$$= \frac{A/C}{A/C'} \approx \frac{C'}{C} \tag{4.21}$$

$$\approx \frac{1}{\cos(\Delta w)} \tag{4.22}$$

where in (4.21) and (4.22) the small-angle approximation is applied. From R we easily get the value of λ

$$\lambda = \frac{1}{\mu}\left[\sqrt[L]{R} - 1\right] \tag{4.23}$$

Bibliography

[1] H. Nyquist, "Regeneration theory," *Bell Syst. Tech. J.*, vol. 11, pp. 126–147, 1932.

[2] A. Spriet, S. Doclo, M. Moonen, , and J. Wouters, *Feedback control in hearing aids*. Springer, Germany, ch. 6 in "Part H. Speech Enhancement of the Springer Handbook of Speech Processing and Speech Communication (Benesty J., Huang Y. A., Sondhi M., eds.)".

[3] T. van Waterschoot and M. Moonen, "Fifty years of acoustic feedback control: state-of-the-art and future challenges," *Proc. IEEE, vol. 99, no. 2, Feb. 2011, pp. 288327.*, vol. 99, no. 2, pp. 288–327, Feb. 2011.

[4] J. B. Foley, "Adaptive periodic noise cancellation for the control of acoustic howling," in *Proc. IEEE Colloq. Adapt. Filt.*, London, UK, Mar 1989, pp. 1–4.

[5] S. M. Kuo and J. Chen, "New adaptive IIR notch filter and its application to howling control in speakerphone system," *IEEE Electr. Lett.*, vol. 28, no. 8, pp. 764–766, Apr 1992.

[6] A. Nehorai, "A minimal parameter adaptive notchfilter with constrained poles and zeros," *IEEE Trans. Acoust. Speech, Signal Process.*, vol. 33, no. 4, pp. 983–996, Aug. 1985.

[7] S. Jiao and M.H.Nagrial, "Modeling and simulation of a real time adaptive notch filter for sinusoidal frequency tracking," *Int. Conf. Power Electr. Drives Energy Syst. Indust. Growth*, vol. 2, no. 1, pp. 948–952, Dec. 1998.

[8] J. M. Travassos-Romano and M. G. Bellanger, "Fast least-squares adaptive notch filtering," *IEEE Trans. Acoust. Speech, Signal Process*, vol. 36, no. 9, pp. 1536–1540, Sept. 1984.

[9] D. V. Bhaskar and S. Kung, "Adaptive notch filtering for the retrieval of sinusoids in noise," *IEEE Trans. Acoust. Speech Signal Process*, vol. 32, no. 4, pp. 791–802, Aug. 1984.

[10] P. Regalia, *Adaptive IIR Filtering in Signal Processing and Control.* 270 Madison Avenue, NY: Marcel Dekker, 1995.

[11] K. Mayyas and T. Aboulnasr, "Leaky LMS algorithm: MSE analysis for gaussian data," *IEEE Trans. Signal Process.*, vol. 45, no. 4, pp. 927–934, Apr. 1997.

Chapter 5

A frequency-domain adaptive filter (FDAF) prediction error method (PEM) framework for double-talk-robust acoustic echo cancellation

A frequency-domain adaptive filter (FDAF) prediction error method (PEM) framework for double-talk-robust acoustic echo cancellation

Jose Manuel Gil-Cacho, Toon van Waterschoot, Marc Moonen and Søren Holdt Jensen,

Submitted to IEEE Transaction Audio Speech and Language Processing, Aug. 2013

Contributions of first author

- literature study
- co-development of the FDAF-PEM-AFROW algorithm
- design of simulation scenarios
- software implementation and computer simulations
- co-interpretation of simulation results
- text redaction and editing

Abstract

In this chapter, we propose a new framework to tackle the double-talk (DT) problem in acoustic echo cancellation (AEC). It is based on a frequency-domain adaptive filter (FDAF) implementation of the so-called PEM-AFROW algorithm (FDAF-PEM-AFROW). We show that FDAF-PEM-AFROW is by construction related to the best linear unbiased estimate (BLUE) of the echo path. We depart from this framework to show an improvement in performance with respect to other adaptive filters minimizing the BLUE criterion, namely the PEM-AFROW and the FDAF-NLMS with near-end signal normalization. One of the contributions is to propose the instantaneous pseudo-correlation (IPC) measure between the near-end signal and the loudspeaker signal. The IPC measure serves as an indication of the effect of a DT situation occurring during adaptation. We motivate the choice of FDAF-PEM-AFROW over PEM-AFROW and FDAF-NLMS with near-end signal normalization, based on performance, computational complexity and related IPC measure values. Moreover, we use the FDAF-PEM-AFROW framework to improve several state-of-the-art variable step-size (VSS) and variable regularization (VR) algorithms. The FDAF-PEM-AFROW versions significantly outperform the original versions in every simulation by at least 6 dB during DT. In terms of computational complexity, the FDAF-PEM-AFROW versions are themselves 10 times cheaper than the original versions.

5.1 Introduction

ACOUSTIC echo cancellation (AEC) is used in many speech communication applications where the existence of echoes degrades the speech intelligibility and listening comfort [1], [2]. These applications range from mobile and hands-free telephony to teleconferencing and voice over IP (VoIP), and are often integrated in smartphones, tablets, notebooks, laptops, etc. The typical set-up for an acoustic echo canceler is depicted in Figure 5.1.

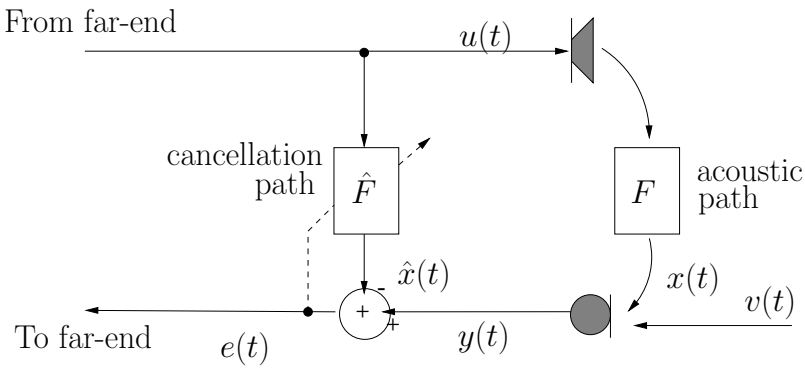


Figure 5.1: Typical set-up for AEC.

A far-end speech signal $u(t)$ is played back in an enclosure (i.e., the room) through a loudspeaker. In the room there is a microphone to record a near-end speech signal which is to be transmitted to the far-end side. An acoustic echo path between the loudspeaker and the microphone exists so that the microphone signal $y(t)$ contains an undesired echo signal $x(t)$ plus the near-end signal $v(t)$, i.e., $y(t) = x(t)+v(t)$. The echo signal $x(t)$ can be considered as the far-end speech or loudspeaker signal $u(t)$ filtered by the echo path. An acoustic echo canceler seeks to cancel the echo signal component $x(t)$ in the microphone signal $y(t)$, ideally leading to an *echo-free* error signal $e(t)$, which is then transmitted to the far-end side. This is done by subtracting an estimate of the echo signal $\hat{x}(t)$ from the microphone signal, i.e., $e(t) = y(t) - \hat{x}(t)$. Standard approaches to AEC rely on the assumption that the echo path can be modeled by a linear FIR filter [3]- [5]. The coefficients of the echo path are collected in the parameter vector $\mathbf{f}(t) = [f_0(t), f_1(t), , ..., f_{N-1}(t)]^T \in \mathbb{R}^N$ such that $x(t) = \mathbf{f}^T(t)\mathbf{u}(t) = F(q,t)u(t)$ where $\mathbf{u}(t) = [u(t), u(t-1), \ldots, u(t-N+1)]^T$ and $F(q,t) = f_0(t) + f_1(t)q^{-1} + ... + f_{N-1}(t)q^{N-1}$ with q^{-1} the unit delay operator, i.e., $q^{-1}u(t) = u(t-1)$. An adaptive filter of sufficient order is used to provide an estimate $\hat{\mathbf{f}}(t) = [\hat{f}_0(t), \hat{f}_1(t), ..., \hat{f}_{N-1}(t)]^T \in \mathbb{R}^N$ of $\mathbf{f}$, such that the echo signal estimate is $\hat{x}(t) = \hat{\mathbf{f}}^T(t)\mathbf{u}(t) = \hat{F}(q,t)u(t)$ with $\hat{F}(q,t) = \hat{f}_0(t) + \hat{f}_1(t)q^{-1} + ... + \hat{f}_{N-1}(t)q^{N-1}$.

Practical AEC implementations often rely on computationally simple time-domain stochastic gradient algorithms, such as the least mean squares (LMS) algorithm [34], the normalized LMS (NLMS) algorithm [4], or affine projection algorithm (APA) [12], which are very sensitive to the presence of a near-end signal [3]. In general, the presence of a near-end signal, in a so-called double-talk (DT) scenario, makes the AEC adaptive filter converge slowly or even diverge. There are several approaches to tackle the problem of DT in AEC. We give here a brief explanation of five different approaches: two that are based

on a *variable step size* (VSS), one based on a *variable regularization* (VR) and two based on a *best linear unbiased estimate* (BLUE) of the echo path.

The first approach is based on the so-called gradient-based VSS algorithms [6]- [11]. From this class of algorithms, the only one specifically designed for DT-robust AEC is the *projection-correlation* VSS (PCVSS) which has been proposed in [11]. PCVSS is based on the APA, and so will be referred to as PCVSS-APA. In PCVSS-APA, the adaptation rate is controlled by a measure of the correlation between instantaneous and long-term averages of the so-called *projection vectors*, i.e., gradient vectors in APA.

The second approach is based on the *non-parametric* VSS (NPVSS) algorithm proposed in [13]. Different NPVSS-based algorithms have been developed and applied for DT-robust AEC when updated with the NLMS algorithm, e.g., [14] and [15]. Their convergence, however, has been found to be slow in practice. Hence, also an APA version of these algorithms has been proposed to increase the convergence speed, resulting in the *practical* VSS affine projection algorithm (PVSS-APA) [16].

The third approach is based on equipping the adaptive filter with a VR. Several VR algorithms have been proposed in the literature based on the derivation of an optimal regularization parameter, which sometimes need quantities or models that are difficult to obtain in practice [17], [18]. Practical implementations of these algorithms have also been proposed that employ more easily measurable quantities. One example is the APA-based VR (VR-APA) algorithm that has been proposed in [19], which incorporates the statistics of the noise into the design of the VR.

The fourth approach is based on a minimum-variance echo path estimate, i.e., the BLUE [20]. This estimate depends on the near-end signal characteristics, which are in practice unknown and time-varying [3], [21]. One approach to achieve the BLUE is based on the prediction error method (PEM) [22] for jointly estimate the echo path model and an auto-regressive (AR) model of the near-end signal [31]. The algorithms in [3], [21] and [31] aim to whiten the near-end signal component in the microphone signal by using adaptive decorrelation prefilters that are estimated concurrently with the echo path. Among the PEM-based algorithms, the PEM-based adaptive filtering using row operations (PEM-AFROW) [23] is particularly interesting and it will be explained further on. Other algorithms, although not applied to DT-robust AEC, have been proposed for recursively minimizing the BLUE criterion. In fact, in [24], [25], a frequency-domain adaptive filtering (FDAF) algorithm has been obtained by first minimizing the BLUE criterion using a time-domain block stochastic gradient algorithm and then switching to the frequency domain to reduce the computational complexity. One advantage of frequency-domain adaptive filtering compared to time-domain adaptive filtering is that the step size can be normalized independently for each frequency bin. Including such a normalization

in the adaptive filter update equation results in a more uniform convergence over the entire frequency range. In the sequel, the standard frequency-domain adaptive filtering [36] is referred to as FDAF-NLMS, i.e., an FDAF with a loudspeaker signal normalization factor. The algorithm proposed in [24] is therefore referred to as FDAF-NLMS *with near-end signal normalization*. The PEM-based and FDAF-based approach to achieve the BLUE will be explained further on.

One particular assumption in all these algorithms, and in most AEC applications in general, is that the near-end signal is uncorrelated with the loudspeaker signal. This assumption can truly be exploited only for infinitely long observations of ergodic and stationary processes [4]. In real AEC applications, however, the near-end signal as well as the loudspeaker signal is a speech signal that is highly colored and non-stationary. Even when the near-end signal and the loudspeaker signal are assumed to be uncorrelated, this however does not imply that the correlation between these two signals is zero within a short-time observation window [30]. Hence, one of the contributions of this chapter is to derive and define the *instantaneous pseudo-correlation* (IPC) measure between the near-end signal $v(t)$ and the loudspeaker signal $u(t)$. The IPC measure serves as an indication of the effect of a DT situation occurring during adaptation, as it will be explained.

The aim of this chapter is to introduce a new framework for DT-robust AEC which is based on an FDAF implementation of the PEM-AFROW (FDAF-PEM-AFROW). We depart from this framework to show an improvement in performance with respect to PEM-AFROW and FDAF with near-end signal normalization. Although these three algorithms are related to the BLUE, we show that FDAF-PEM-AFROW is the preferred choice for DT-robust AEC. We motivate the choice of FDAF-PEM-AFROW over PEM-AFROW and FDAF-NLMS with near-end signal normalization, based on performance improvement, computational complexity reduction and lower IPC measure values. Moreover, we use the FDAF-PEM-AFROW framework to improve the previously introduced VR-APA, PVSS-APA and PCVSS-APA algorithm leading to the VR-FDAF-PEM-AFROW, PCVSS-FDAF-PEM-AFROW and PCVSS-FDAF-PEM-AFROW respectively. The FDAF-PEM-AFROW versions significantly reduce the computationally complexity and improve the performance of the original versions in every simulation .

The chapter is organized as follows. In Section 5.2, the BLUE is explained in two subsections: in Section 5.2.1, the basic linear regression model is reviewed, including unbiasedness constraints that are assumed in the derivation of typical algorithms for AEC. In Section 5.2.2, the generalized least squares model is reviewed, which provides a framework to explain other related estimators. In Section 5.3.1, the BLUE achieved using the PEM is considered and in Section 5.3.2, the BLUE achieved using the FDAF-NLMS with near-end signal normalization is considered. In Section 5.4, the IPC measure is

derived and defined, and several simulation results are shown evaluating the IPC measure in the NLMS and the FDAF-NLMS. The recursions that are used to compute the IPC measure are given in Appendix 5.A. In Section 5.5, the proposed FDAF-PEM-AFROW is derived. In Section 5.6, computer simulations are provided. In Section 5.6.1, we motivate the choice of the FDAF-PEM-AFROW algorithm over PEM-AFROW and FDAF-NLMS with near-end signal normalization, based on performance improvement, computational complexity reduction and lower IPC measure values. In Section 5.6.2 more detail is provided about the selected state-of-the-art VSS and VR algorithms as well as an explanation of their FDAF-PEM-AFROW versions. Simulation results are provided with a complexity analysis comparing the following algorithms: PVSS-APA, PCVSS-APA, VR-APA, PVSS-FDAF-PEM-AFROW, PCVSS-FDAF-PEM-AFROW and VR-FDAF-PEM-AFROW. Finally, Section 5.7 concludes the chapter.

5.2 Best linear unbiased estimate

5.2.1 Linear unbiased estimator

We first assume that the echo path is time-invariant, $\mathbf{f}(t) = \mathbf{f}, \forall t$ over the observation window $t = 1, 2, \ldots, L$ with $L >> N$. A regression data model may be constructed as

$$\mathbf{y} = \mathbf{X}\mathbf{f} + \mathbf{v} \tag{5.1}$$

where

$$\mathbf{y} = [y(1), y(2), ..., y(L))]^T \tag{5.2}$$

$$\mathbf{v} = [v(1), v(2), ..., v(L))]^T \tag{5.3}$$

$$\mathbf{X} = [\mathbf{u}(1), \mathbf{u}(2), ..., \mathbf{u}(L)]^T, \tag{5.4}$$

with $(\cdot)^T$ the transpose operator. The near-end signal and the loudspeaker signal are usually assumed to be uncorrelated so that $\mathcal{E}\{\mathbf{X}^T\mathbf{v}\} = \mathbf{0}_{N\times 1}$, where $\mathcal{E}\{\cdot\}$ is the expected value operator. Any *linear* estimate of parameter vector $\mathbf{f}$ can be written as a linear function of the data vector $\mathbf{y}$, i.e.,

$$\hat{\mathbf{f}} = \mathbf{D}^T\mathbf{y}. \tag{5.5}$$

For this estimate to be *unbiased*, the $L \times N$ matrix $\mathbf{D}$ should be subjected to two constraints,

$$\mathbf{D}^T\mathbf{X} = \mathbf{I}_N \tag{5.6}$$

$$\mathcal{E}\{\mathbf{D}^T\mathbf{v}\} = \mathbf{0}_{N\times 1} \tag{5.7}$$

These constraints are often implicitly assumed to hold in most of the existing AEC algorithms.

5.2.2 Generalized least squares and BLUE

Let us now consider a *generalized least squares* (GLS) estimator given as

$$\hat{\mathbf{f}}_{\text{GLS}} = \left[\mathbf{X}^T\mathbf{M}^{-1}\mathbf{X}\right]^{-1}\mathbf{X}^T\mathbf{M}^{-1}\mathbf{y} \tag{5.8}$$

This estimator is quite general since the structure of the weighting matrix $\mathbf{M}$ can result in different types of estimators [27], i.e., biased or unbiased, minimum-variance or not. As we are interested in an $\mathbf{M}$ that makes (5.8) the BLUE [3], we consider

$$\mathbf{M}_{\text{BLUE}} = \mathbf{R}_v = \mathcal{E}\{\mathbf{v}\mathbf{v}^T\}, \tag{5.9}$$

where $\mathbf{R}_v$ is the near-end signal autocorrelation matrix with symmetric Toeplitz structure, such that

$$\hat{\mathbf{f}}_{\text{BLUE}} = \left[\mathbf{X}^T\mathbf{R}_v^{-1}\mathbf{X}\right]^{-1}\mathbf{X}^T\mathbf{R}_v^{-1}\mathbf{y}. \tag{5.10}$$

Note that, when the near-end signal is a white noise with variance σ_v^2, the weighting matrix $\mathbf{R}_v = \sigma_v^2\mathbf{I}_L$, where $\mathbf{I}_L$ is the $L \times L$ identity matrix. If we substitute this into (5.10) the ordinary least squares (OLS) estimator is obtained, i.e.,

$$\hat{\mathbf{f}}_{\text{LS}} = \left[\mathbf{X}^T\mathbf{X}\right]^{-1}\mathbf{X}^T\mathbf{y} \tag{5.11}$$

This means that the OLS estimator achieves the BLUE only if the near-end signal is a white noise signal, otherwise the OLS estimate is suboptimal. Most of the existing AEC algorithms are based on the OLS estimator which is assumed to be unbiased following (5.6) and (5.7), but which is not necessarily minimum-variance.

5.3 The BLUE in adaptive filtering algorithms

It has been shown [21] that the BLUE can be considered an optimal acoustic echo path estimate during DT. However, calculating the BLUE in an AEC framework is problematic due to its dependence on the near-end signal covariance matrix $\mathbf{R}_v$. Therefore, we will seek a signal transformation that diagonalizes $\mathbf{R}_v$ and include an estimation of the resulting diagonal elements in the proposed adaptive filtering algorithm. PEM-based algorithms, which have been proposed in [3], [21], provide a signal-dependent way of diagonalizing $\mathbf{R}_v$, which requires the estimation of a near-end signal model. On the other hand, FDAF-based algorithms which have been proposed in [24], [25] are capable of diagonalizing $\mathbf{R}_v$ in a signal-independent way, after some matrix manipulation, and thus provide an attractive alternative to the PEM-based algorithms.

5.3.1 PEM-based BLUE

Note that the BLUE as in (5.10) usually cannot be calculated as such, because the autocorrelation matrix $\mathbf{R}_v$ is generally unknown. Therefore, $\mathbf{R}_v$ will be conveniently transformed. Let us first assume that the near-end signal is generated as $v(t) = H(q,t)w(t)$ where $w(t)$ is a white noise excitation signal with variance $\sigma_w(t)$, i.e., $\mathcal{E}\{w(t)w(t-i)\} = \delta(i)\sigma_w^2(t)$, and

$$H(q,t) = \frac{1}{A(q,t)} = \frac{1}{1 + a_1(t)q^{-1} + ... + a_{n_A}(t)q^{-n_A}} \tag{5.12}$$

is the near-end signal auto-regressive (AR) model, expressed as an order-n_A time-varying linear filter. The near-end signal autocorrelation matrix may then be written as

$$\mathbf{M}_{\text{PEM}} = \mathcal{E}\{\mathbf{H}(q)\mathbf{w}\mathbf{w}^T\mathbf{H}^T(q)\} \tag{5.13}$$

where $\mathbf{w} = [w(1), ..., w(L)]^T$ and the diagonal operator $\mathbf{H}(q)$ has the filters $H(q,t)$, $t = 1, ..., L$ on the main diagonal. In [3], [21] it has been shown that the BLUE can then be realized as

$$\hat{\mathbf{f}}_{\text{BLUE}} = \left[\left(\mathbf{H}^{-1}(q)\mathbf{X}\right)^T \mathbf{W}_{\text{PEM}}^{-1} \left(\mathbf{H}^{-1}(q)\mathbf{X}\right)\right]^{-1} \left(\mathbf{H}^{-1}(q)\mathbf{X}\right)^T \mathbf{W}_{\text{PEM}}^{-1} \left(\mathbf{H}^{-1}(q)\mathbf{y}\right) \tag{5.14}$$

with

$$\mathbf{W}_{\text{PEM}} = \begin{bmatrix} \sigma_w^2(1) & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_w^2(L) \end{bmatrix}, \tag{5.15}$$

which corresponds to a prefiltering and weighting of the tth row, $t = 1, ..., L$ of $\mathbf{X}$ and $\mathbf{y}$ with the inverse near-end signal model $H^{-1}(q,t) = A(q,t)$ and the inverse excitation signal variance $\sigma_w^{-1}(t)$. In practice, the AR model parameters in $A(q,t)$ and $\sigma_w(t)$ have to be estimated concurrently with the echo path model at each time instant $t = 1, ..., L$. The use of the PEM [22] has been proposed to achieve this, jointly providing estimates $\hat{f}(t)$, $\hat{A}(q,t)$ and $\hat{\sigma}_w(t)$ [31].

The PEM-based approach achieving the BLUE leads to very simple time-domain stochastic gradient algorithms that feature two components (as shown in Figure 5.2): (1) a whitening of the near-end signal component in the microphone signal by using estimated decorrelation prefilters $\hat{A}(q,t)$, (2) a single weighting scalar in the denominator of the adaptive filter update equation using the estimated excitation signal variance $\hat{\sigma}_w(t)$ [3], [21].

In [21], it has been shown that PEM-based algorithms that achieve the BLUE, are possible if three conditions are fulfilled, (1) the near-end signal $v(t)$ can at each time instant be modeled as an AR process of order n_A, (2) the prefilter $\hat{A}(q,t)$ contains at each time instant the true AR coefficients, i.e., $\hat{A}(q,t) = A(q,t)$, and (3) the weighting scalar in the denominator of the adaptive filter update equation $\hat{\sigma}_w(t)$ is at each time instant equal to the true variance of the

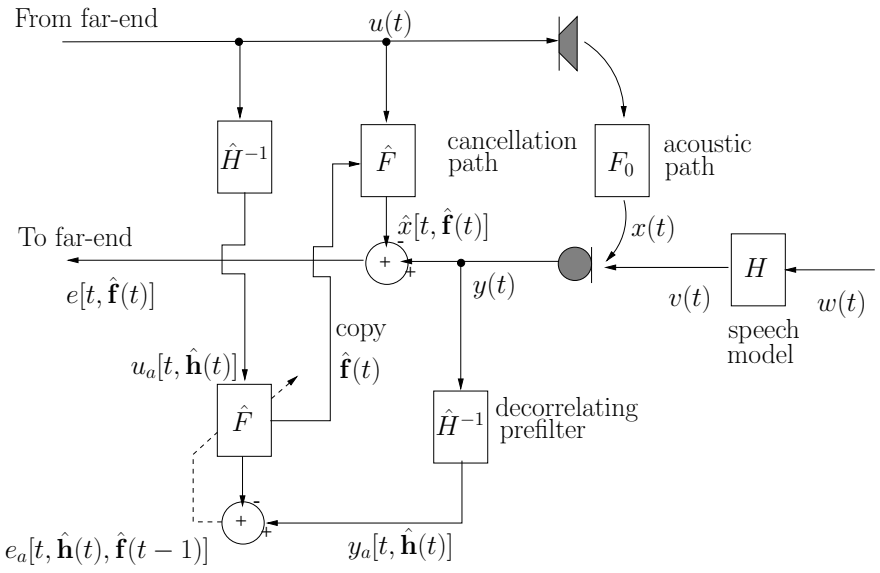


Figure 5.2: Typical set-up for AEC using a decorrelation prefilter.

near-end excitation signal $w(t)$, i.e., $\hat{\sigma}_w(t) = \sigma_w(t)$. However, it seems that some of the previous conditions are difficult to fulfill in practice. In this case, the complete whitening of the near-end signal will not be achieved (i.e., $\mathbf{W}_{\text{PEM}}$ will not be diagonal) and, therefore, the use of a single weighting scalar $\hat{\sigma}_w(t)$ in the denominator of the adaptive filter update equation will not be possible. Even if the near-end signal autocorrelation matrix $\mathbf{R}_v$ could be estimated as such, its direct inversion, as needed in a typical time-domain stochastic gradient algorithm, would be prohibitive in terms of computational complexity.

5.3.2 FDAF-based BLUE

In this section, we explain how in [24] an FDAF-based algorithm has been obtained by first minimizing the BLUE criterion using a time-domain block stochastic gradient algorithm and then switching to the frequency domain to reduce the computational complexity. To this end, consider the cost function

$$J_{\text{BLUE}}(\mathbf{f}) = (\mathbf{X}\mathbf{f} - \mathbf{y})^T \mathbf{R}_v^{-1} (\mathbf{X}\mathbf{f} - \mathbf{y}), \tag{5.16}$$

from which the BLUE in (5.10) is actually obtained. Following [24] and [25], a time-domain block stochastic gradient algorithm minimizing (5.16) can be derived. In [24]- [26] and [29] it has been shown how FDAF-based algorithms can then be derived by rewriting a time-domain block stochastic gradient algorithm in a way that Toeplitz and circulant matrices are explicitly shown. It is

important to recognize that the Toeplitz property of $\mathbf{R}_v$ is a direct consequence of the assumtion that the vector $\mathbf{v}$ is wide-sense stationary [4].

Similar to [26], [29], in [24], [25] a Toeplitz matrix is diagonalized in two steps: (1) a Toeplitz matrix is transformed into a circulant matrix and (2) a circulant matrix is transformed into a diagonal matrix using the DFT. In the resulting diagonal matrix, the different diagonal elements, correspond to the near-end signal variance in different frequency bins. In FDAF-based algorithms, the near-end signal variance in each frequency bin can be straightforwardly incorporated as a normalization factor in the adaptive filter update equation. Indeed, in [24], and [25], it has been shown that such a normalization can significantly improve the performance of an FDAF-NLMS. The algorithm proposed in [24] is therefore referred to as FDAF-NLMS with near-end signal normalization. In [24] and [25], only a stationary colored near-end noise signal is considered and the near-end noise signal variance in each frequency bin is then directly estimated from the error signal $e(t)$.

5.4 Instantaneous pseudo-correlation measure

The unbiasedness constraints in (5.6), (5.7) must be satisfied in both the PEM-based and FDAF-based BLUE. Since $\mathbf{D}$ in (5.7) is a function of the loudspeaker signal matrix $\mathbf{X}$, the constraint reduces to $\mathcal{E}\{\mathbf{X}^T\mathbf{v}\} = \mathbf{0}_{N\times 1}$. This means that the near-end signal should be uncorrelated with the loudspeaker signal. This assumption can truly be exploited only for infinitely long observations of ergodic and stationary processes [4]. In real AEC applications, however, the near-end signal as well as the loudspeaker signal is a speech signal that is highly colored and non-stationary. Even when the near-end signal and the loudspeaker signal are assumed to be uncorrelated, this however does not imply that the correlation between these two signals is zero within a short-time observation window [30]. It is known [31], [33] that the correlation between these two signals causes standard adaptive filtering algorithms to converge to a *biased* solution. This means that the adaptive filter does not only predict and cancel the echo signal component in the microphone signal, but also part of the near-end signal. To analyze this, we derive and define the *instantaneous pseudo-correlation* (IPC) measure between the near-end signal $v(t)$ and the loudspeaker signal $u(t)$. It should be clear that the IPC measure is a performance measure used in simulations hence the individual and clean signals are needed.

To this end, we first consider the time-domain LMS algorithm update equation given as

$$e(t) = y(t) - \hat{\mathbf{f}}^T(t-1)\mathbf{u}(t) \tag{5.17}$$

$$\hat{\mathbf{f}}(t) = \hat{\mathbf{f}}(t-1) + \mu e(t)\mathbf{u}(t). \tag{5.18}$$

By repeated application of (5.18), and assuming $\hat{f}(0) = 0$, it is straightforwardly shown that

$$\hat{\mathbf{f}}(t-1) = \mu \sum_{i=1}^{t-1} e(i)\mathbf{u}(i) \tag{5.19}$$

so that

$$e(t) = y(t) - \mu \sum_{i=1}^{t-1} e(i) \left(\mathbf{u}^T(i)\mathbf{u}(t)\right) \tag{5.20}$$

where $y(t) = x(t) + v(t)$.

Let us now consider a second scenario where the near-end signal is absent, i.e. $v(i) = 0$, $i = 1, ..., t$, and so the microphone signal is equal to the echo signal $y(i) = x(i)$, $i = 1, ..., t$. Applying the LMS algorithm to this scenario (denoted by means of a subscript 'x') then leads to

$$e_x(t) = x(t) - \mu \sum_{i=1}^{t-1} e_x(i) \left(\mathbf{u}^T(i)\mathbf{u}(t)\right) \tag{5.21}$$

Here, the $e_x(t)$ corresponds to the residual echo, when an 'ideal' filter adaption is run, i.e. in the absence of a near-end signal. In a DT scenario, the $e(t)$ can then be identified as the sum of $e_x(t)$ and an extra $e_v(t)$, which is due to the DT. From (5.20) and (5.21) it follows that

$$\begin{aligned} e_v(t) &= e(t) - e_x(t) \\ &= y(t) - x(t) - \mu \sum_{i=1}^{t-1} \big(e(i) - e_x(i)\big)\big(\mathbf{u}^T(i)\mathbf{u}(t)\big) \\ &= v(t) - \mu \sum_{i=1}^{t-1} e_v(i)\big(\mathbf{u}^T(i)\mathbf{u}(t)\big) \end{aligned} \tag{5.22}$$

which effectively corresponds to applying the LMS algorithm to a scenario with only the near-end signal $v(t)$ and no echo signal, i.e., $x(i) = 0$ $i = 1, ..., t$.

The aim of AEC is to have the best possible echo cancellation, i.e. the smallest possible $e_o(t)$, next to preserving the near-end signal $v(t)$. Hence, ideally $e_v(t)$ should be equal to $v(t)$. From (5.22) it follows that this implies that ideally

$$\mu \sum_{i=1}^{t-1} e_v(i) \left(\mathbf{u}^T(i)\mathbf{u}(t)\right) = 0 \tag{5.23}$$

Based on (5.23), we define a signal

$$z(t) = \frac{\mu}{N} \sum_{i=1}^{t-1} e_v(i) \left(\mathbf{u}^T(i)\mathbf{u}(t)\right) \tag{5.24}$$

$$= \frac{\mu}{N}\tilde{\mathbf{f}}^T(t-1)\mathbf{u}(t) \tag{5.25}$$

which resembles a correlation function, hence the name IPC. The signal $z(t)$ contains the factor $1/N$ in (5.24) to normalize the inner product with respect to the filter length. Consequently, we want $z(t)$ to be as small as possible, ideally $z(t) = 0$ as (5.23). Although, unfortunately, this is not the case in practice we show that several algorithms can help to reduce the signal $z(t)$.

We consider the recursions given in Appendix 5.A where it is shown how the signal $z(t)$ is computed in the NLMS and the FDAF-NLMS based on a derivation similar to the above derivation for LMS. As an IPC measure within an adaptation loop, we consider

$$\text{IPC measure} = 10\log_{10}\frac{\sum_{t=1}^{L} z(t)^2}{\sum_{t=1}^{L} v(t)^2} \quad \text{dB}, \tag{5.26}$$

i.e., the estimated variance of the signal $z(t)$ normalized w.r.t. the estimated variance of the near-end signal with L the total length of the signals. The normalization makes the IPC measure independent of the level of the near-end signal. In what follows, we show that within their adaptation loops the NLMS, and the FDAF-NLMS show different values of the IPC measure.

$N = 80$	White noise	Colored noise	Speech
NLMS	-15.3 dB	-14 dB	-12.1 dB
FDAF-NLMS	-34.5 dB	-33.2 dB	-31.7 dB

Table 5.1: IPC measure for the NLMS and the FDAF-NLMS. The loudspeaker signal is a female speech signal and the near-end signal is a white noise signal, a colored noise signal or male speech signal.

Table 5.1 shows the values of the IPC measure for the NLMS and the FDAF-NLMS. Here, the loudspeaker signal is a female speech signal and the near-end signal is either a white noise signal, a colored noise signal or male speech signal, and $N = 80$ equal to the length of the impulse response of the system under study. The smallest IPC measure is obtained using the FDAF-NLMS in each scenario. As expected, it can also be observed that the largest IPC measure is obtained with a near-end signal. In addition, the values of the IPC measure for different filter lengths are shown in Figure 5.3. It is shown that by increasing the filter length, the IPC measure in NLMS and FDAF-NLMS is reduced. This reduction is clearly higher in the FDAF-NLMS.

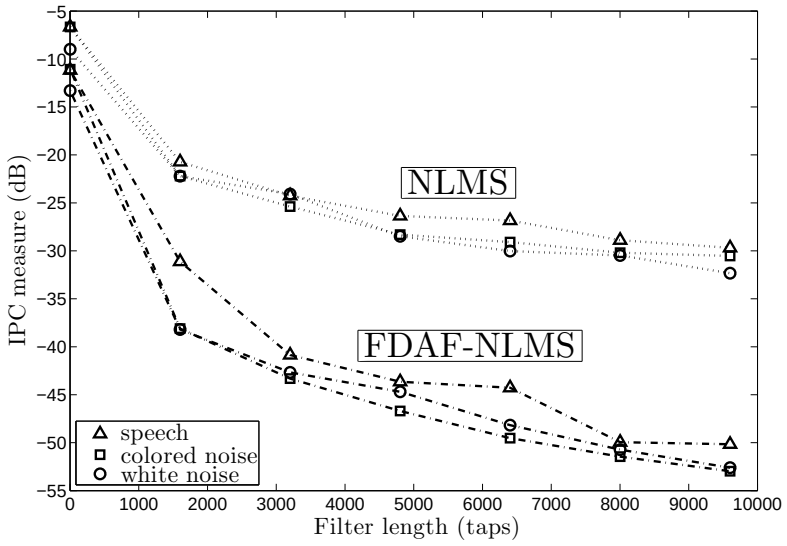


Figure 5.3: IPC measure for the NLMS and the FDAF-NLMS. The loudspeaker signal is a female speech signal and the near-end signal is a white noise signal, a colored noise signal or male speech signal.

5.5 The FDAF-PEM-AFROW algorithm

Among the PEM-based algorithms proposed for DT-robust AEC in [3] and [21], the PEM-based adaptive filtering using row operations (PEM-AFROW) [23] is particularly interesting. When the near-end signal $v(t)$ is a speech signal which is considered to be short-term stationary, the near-end signal model $A(q,t)$ does not need to be re-estimated at each time instant t. That is, instead of identifying the near-end signal model recursively, this can also be identified non-recursively on a block of loudspeaker and microphone samples. This is the idea behind the PEM-AFROW, which estimates $A(q,t)$ in a block-based manner, using a block length that approximates the stationarity interval of the near-end signal. The coefficients of the prefilter $\hat{\mathbf{a}}(t) = [\hat{a}_1(t), ..., \hat{a}_{n_A}]^T$ (5.12) and $\hat{\sigma}_w(t)$ are efficiently computed using the Levinson-Durbin recursion. The algorithm proposed in this chapter extends the PEM-AFROW using the well-known FDAF with gradient constraint, based on the overlap-save method where the size of each input block is N and so the DFT size is $M = 2N$. For a complete description of the PEM-AFROW and the FDAF the reader is referred to [23] and [36], respectively. A complete description of the algorithm proposed here, which is refereed to as FDAF-PEM-AFROW, is provided as Algorithm 1, where $\mathcal{F}(\mathcal{F}^{-1})$ represents a DFT(IDFT) operation, '$\circ$' represents an element-

Algorithm 1 FDAF-PEM-AFROW

1: Initialize: $M = 2N$, $\hat{\mathbf{F}}(0) = \mathbf{e}^T(0) = \mathbf{S}_{U_a}(0) = \mathbf{0}_{M\times 1}$.
2: Vectors $\mathbf{u}(k)$, $\mathbf{y}(k)$, and hence $\mathbf{u}_a(k)$ and $\mathbf{y}_a(k)$, are length-M vectors satisfying the overlap-save condition in FDAF, $\mathbf{u}(k) = [u(kN-N+1), ...u(kN+N)]^T$, $\mathbf{y}(k) = [y(kN-N+1), ...y(kN+N)]^T$, $\mathbf{u}_a(k) = [u_a(0), ...u_a(M-1)]^T$ and $\mathbf{y}_a(k) = [y_a(0), ...y_a(M-1)]^T$.
3: **for** $k = 1, 2, ...$ **do**
4: $\quad \mathbf{e}(k) = \left[\mathbf{y}(k) - \mathcal{F}^{-1}\left\{\mathcal{F}\{\mathbf{u}(k)\} \circ \hat{\mathbf{F}}(k-1)\right\}\right]_{N+1:M}$
5: $\quad \mathbf{r}(k) = [\mathbf{e}^T(k), \quad \mathbf{e}^T(k-1)]^T$
6: $\quad [\hat{\mathbf{a}}(k) \quad \hat{\sigma}_w(k)] = \text{Levison-Durbin}\{[\mathbf{r}(k)]_{1:P}, n_A\}$
7: $\quad$ **for** $m = 0, ..., M-1$ **do** (Decorrelation prefilter)
8: $\quad\quad u_a(m,k) = [u(kN+1+m), ..., u(kN+1+m-n_A)]\hat{\mathbf{a}}(k)$
9: $\quad\quad y_a(m,k) = [y(kN+1+m), ..., y(kN+1+m-n_A)]\hat{\mathbf{a}}(k)$
10: $\quad$ **end for**
11: $\quad \mathbf{U_a}(k) = \mathcal{F}\{\mathbf{u}_a(k)\}$
12: $\quad \mathbf{e}_a(k) = \left[\mathbf{y}_a(k) - \mathcal{F}^{-1}\left\{\mathbf{U_a}(k) \circ \hat{\mathbf{F}}(k-1)\right\}\right]_{N+1:M}$ (Prediction-error signal)
13: $\quad \mathbf{E}_a(k) = \mathcal{F}\{\begin{bmatrix}\mathbf{0}_N & \mathbf{e}_a^T(k)\end{bmatrix}^T\}$
14: $\quad$ **for** $m = 0, 1, ..., M-1$ **do**
15: $\quad\quad S_{U_a}(m,k) = \lambda_0 S_{U_a}(m,k-1) + (1-\lambda_0)|U_a(m,k)|^2$ (Recursive power estimate of loudspeaker signal)
16: $\quad\quad G(m,k) = \left(S_{U_a}(m,k) + \hat{\sigma}_w(k) + \alpha\right)^{-1}$ (Normalization factor)
17: $\quad$ **end for**
18: $\quad \hat{\mathbf{F}}(k) = \hat{\mathbf{F}}(k-1) + \mu_0 \mathcal{F}\left\{\begin{bmatrix}\left(\mathcal{F}^{-1}\{\mathbf{G}(k) \circ \mathbf{U}_a{}^*(k) \circ \mathbf{E}_a(k)\}\right)_{1:N}^T & \mathbf{0}_N\end{bmatrix}^T\right\}$
19: **end for**

wise multiplication, $[\cdot]_{a:b}$ represents a range of samples within a vector, k is the block index, capital letters represent frequency-domain variables, lowercase letters represent time-domain variables, boldface letters represent vector variables and non-boldface letters represent scalar variables.

In *line* 5, two length-N error vectors are concatenated in $\mathbf{r}(k)$ to calculate both the AR coefficients and the near-end excitation signal variance using the Levison-Durbin algorithm, where P is the block length used to estimate $A(q,t)$ with $N \leq P \leq 2N$. The order-n_A AR model coefficients are used to prefilter the loudspeaker signal $u(t)$ and the microphone signal $y(t)$ to obtain the length-$2N$ vectors $\mathbf{u}_a$ and $\mathbf{y}_a$. With these prefiltered signals, a standard FDAF algorithm with gradient constraint is then performed in *line* 11 to *line* 19.

It is worth noticing that the normalization factor in *line* 16 contains three terms: (1) the frequency-dependent loudspeaker signal power estimate $S_{U_a}(m,k)$, (2) the non-frequency-dependent near-end excitation signal variance estimated in

line 6 and (3) a small regularization term α to avoid division by zero. The latter term introduces some bias which, however, is assumed to be very small. It is noted that in FDAF-based algorithms, the near-end signal variance estimate in each frequency bin can be straightforwardly incorporated in the normalization factor of the adaptive filter update equation. In [24] and [25], the near-end noise signal variance in each frequency bin is directly estimated in the frequency domain from the error signal $e(t)$. On the other hand, the FDAF-PEM-AFROW uses the same near-end signal variance estimate in each frequency bin, i.e., the near-end excitation signal variance which is readily available from the Levison-Durbin algorithm. This is because PEM-AFROW-based algorithms feature a whitening of the near-end signal by applying the prefilters $\hat{A}(q,t)$ (as shown in Figure 5.2). In the next section, the superior performance of the FDAF-PEM-AFROW compared to PEM-AFROW and FDAF-NLMS with near-end signal normalization is demonstrated.

5.6 Simulation results

Simulations are performed using speech signals. The sampling frequency in every simulation is 8 kHz, the far-end signal is a female speech signal and the near-end signal is a male speech signal. The microphone signal consists of three concatenated segments of speech: the first and third 12.5-s segments consist of echo only, the second segment is the sum of echo and near-end signal generating a DT situation of 13 s. Notice that throughout the chapter, we assume a sufficient-order condition for the acoustic path model, i.e., $N = 80$. The tuning parameters of every algorithm are chosen to have a similar initial convergence that also leads to acceptable DT robustness. Their specific values are shown only for the FDAF-PEM-AFROW versions. The Matlab functions, and scripts with tuning parameters, that are used to generate the figures in this section are available online [1].

The *Misadjustment* (MSD) is used as performance measure to evaluate the different algorithms. The MSD between the estimated echo path $\hat{\mathbf{f}}(t)$ and the true echo path $\mathbf{f}$ represents the accuracy of the echo path estimation and is defined as,

$$\mathrm{MSD}(t) = 10\log_{10}\frac{\left\|\hat{\mathbf{f}}(t)-\mathbf{f}\right\|_2^2}{\|\mathbf{f}\|_2^2}, \quad \mathrm{dB} \tag{5.27}$$

5.6.1 Choice of the FDAF-PEM-AFROW algorithm

[1] $http://homes.esat.kuleuven.be/\sim dspuser/abstract13-13\_2.html$

Algorithm	Computation	Total
PEM-AFROW	$\left(8+\frac{4P+2n_A+1}{P}\right)N+\frac{1}{P}n_A^2+$ $\left(4+\frac{4P+2}{P}\right)n_A+\frac{P-1}{P}+10$	980
FDAF-NLMS near-end sig. norm.	$1/N(18M\log_2 M+18M+$ $3M)$	307
FDAF-PEM-AFROW	$1/N(18M\log_2 M+18M)+$ $1/N(n_A^2+n_A(4+4M))$	334

Table 5.2: Complexity comparison by the number of FLOPS per recursion. One FFT/IFFT is a $3M\log_2 M$ complexity operation. $N=80$, $n_A=1$ and $P=160$.

In this section, we compare the FDAF-PEM-AFROW to the FDAF-NLMS with near-end signal normalization [24] and to the PEM-AFROW [23]. The comparison is done based on both performance and complexity and it is structured to investigate whether or not PEM-AFROW and FDAF-NLMS with near-end signal normalization, could be a better option for DT-robust AEC than the proposed FDAF-PEM-AFROW algorithm. It is noted that these three algorithms are constructed from the BLUE framework.

- Performance

Figure 5.4 shows the MSD performance comparison between the PEM-AFROW, the FDAF-NLMS with near-end signal normalization and the FDAF-PEM-AFROW, where the upper part shows the full-length simulation and the bottom part shows a zoom-in coinciding with the DT situation. FDAF-PEM-AFROW obtains the lowest MSD values of all the other at each time instant both in single talk and in DT. In the bottom part of Figure 5.4, it is clearly seen that FDAF-PEM-AFROW generally outperforms, by $4-6$ dB, the FDAF-NLMS with near-end signal normalization. Compared to PEM-AFROW, FDAF-PEM-AFROW achieves a $7-9$ dB improvement. FDAF-NLMS with near-end signal normalization appears to outperform the PEM-AFROW.

- Computational complexity

Table 5.2 shows the complexity analysis of the three algorithms. It is shown that, for a typical choice of the algorithm parameters, FDAF-PEM-AFROW has a similar complexity as FDAF-NLMS with near-end signal normalization, for a significant performance improvement. Moreover, FDAF-PEM-AFROW is 3.2 times cheaper than PEM-AFROW, for an even more significant performance improvement.

- IPC measure in PEM-AFROW-based algorithms

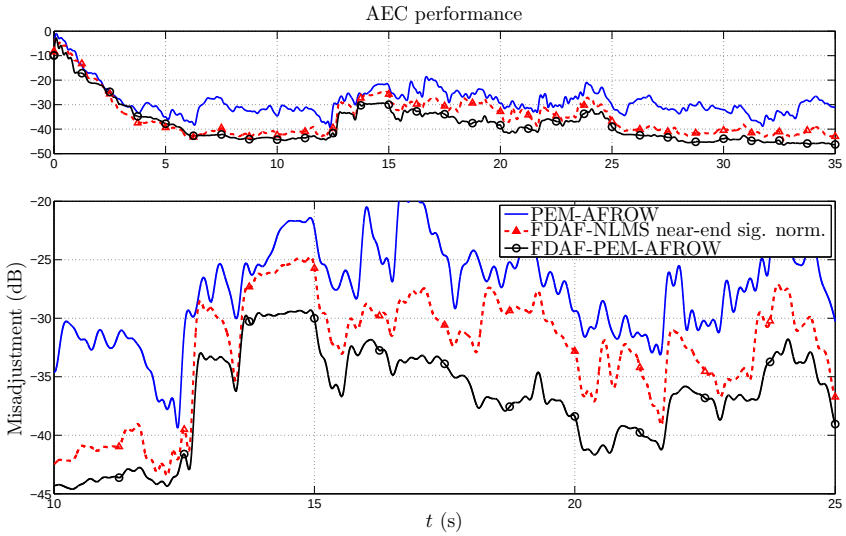


Figure 5.4: AEC performance using speech signals (far-end and near-end) between 12.5 and 25 s. Bursting DT occurs at -10 dB SER and white near-end noise at 30 dB SNR, $N = 80$, $M = 160$, $n_A = 1$, $P = 160$. The value of SNR is typical in AEC applications and the value of SER results in a moderate level. The upper part shows the full-length simulation and the bottom part shows a zoom-in coinciding with the DT. Comparison among three algorithms constructed from the BLUE framework: PEM-AFROW, FDAF-NLMS with near-end signal normalization, and FDAF-PEM-AFROW.

Table 5.3 shows the values of the IPC measure (5.26) for the PEM-AFROW and the proposed FDAF-PEM-AFROW. This is to show the impact that either the prefilter or the type of adaptation (i.e., time-domain or frequency-domain) has on the resulting IPC measure. The IPC measure is calculated without using the near-end signal variance estimate. Calculations are performed in the same scenario as before with $N = 80$ and two different AR model orders, $n_A = 1$ and $n_A = 12$. The recursions for computing $z(t)$ in PEM-based algorithms are given in Appendix 5.A.

The results when the near-end signal is a white noise signal are similar when using $n_A = 1$ and $n_A = 12$. On the other hand, in the colored noise signal case and in the speech signal case the improvement is much higher. Increasing the near-end signal model order, from $n_A = 1$ to $n_A = 12$ also reduces the IPC measure in the speech signal case. The fact that a prefilter is included in the recursions seems to significantly reduce the IPC measure in the colored noise signal case and in the speech signal case. Interestingly enough, the IPC

$N = 80$ and $n_A = 1$	White noise	Colored noise	Speech
PEM-AFROW	-23.8 dB	-26.1 dB	-28 dB
FDAF-PEM-AFROW	-34.4 dB	-41.1 dB	-42.4 dB
$N = 80$ and $n_A = 12$	White noise	Colored noise	Speech
PEM-AFROW	-23.7 dB	-27.9 dB	-27.9 dB
FDAF-PEM-AFROW	-33.9 dB	-40 dB	-44 dB

Table 5.3: IPC measure for PEM-based algorithms. The loudspeaker signal is a female speech signal and the near-end signal is a white noise signal, a colored noise signal or male speech signal. $N = 80$, $n_A = 1$ and $n_A = 12$.

measure in the white noise signal case is similar for FDAF-NLMS in Table 5.1 and for the FDAF-PEM-AFROW in Table 5.3.

To conclude, comparing Table 5.1 to Table 5.3 it seems that the IPC measure in the time-domain NLMS is higher than that of the time-domain PEM-AFROW. The reason appears to be that PEM-AFROW includes a prefiltering operation. However, the IPC measure in PEM-AFROW is higher than that of FDAF-NLMS. The reason is that FDAF-NLMS is implemented in the frequency domain which seems to reduce the IPC measure as seen in Figure 5.3. Finally and gathering both a prefilter and a frequency-domain implementation, the FDAF-PEM-AFROW has the lowest IPC measure of all. Although the three algorithms are constructed from the BLUE framework and should therefore obtain the minimum variance echo path estimate during DT, it turns out that the differences between them are clear (as shown in Figure 5.4). In the PEM-AFROW, it is clear that fulfilling the three conditions to achieve the BLUE in practice is very difficult. This fact seems to affect the time-domain PEM-AFROW much more than the FDAF-PEM-AFROW. The FDAF-NLMS with near-end signal normalization performs better than PEM-AFROW as the conditions to achieve the BLUE seem to be less restricting. It seems however that the assumption of the near-end signal being wide-sense stationary and obtaining the near-end signal variance estimate per frequency bin are also difficult to fulfill in practice. FDAF-NLMS with near-end signal normalization is clearly outperformed by FDAF-PEM-AFROW which has the benefits of being implemented in the frequency-domain, featuring a prefilter and including the non-frequency-dependent near-end signal variance estimate.

5.6.2 Results from VSS algorithms

In this section, we explain the proposed FDAF-PEM-AFROW versions of three state-of-the-art algorithms: variable regularization (VR-APA) [19], practical variable step size (PVSS-APA) [16], projection-correlation variable step size (PCVSS-APA) [11]. It is important to notice that the FDAF-PEM-AFROW

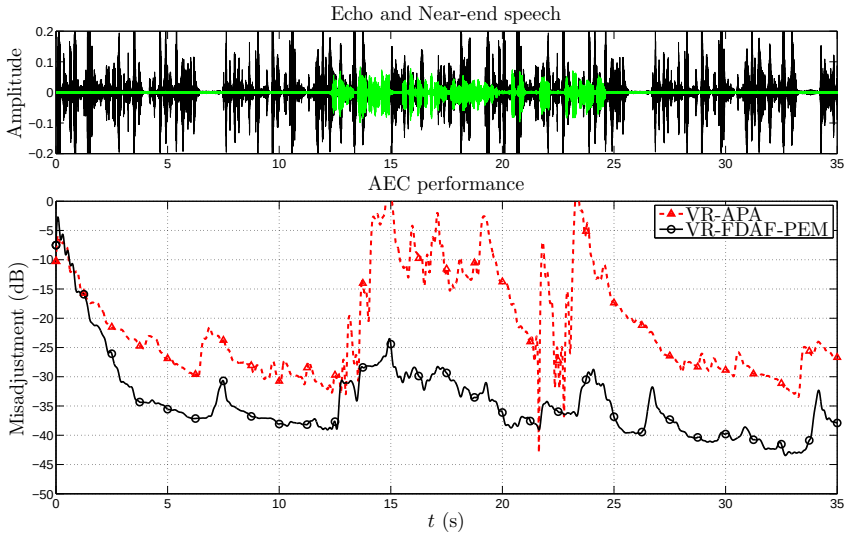


Figure 5.5: AEC performance using speech signals (far-end and near-end) between 12.5 and 25 s. The original APA and the FDAF-PEM-AFROW version of variable regularization are compared when bursting DT occurs at -10 dB SER and white near-end noise at 30 dB SNR. The upper part shows the microphone signal contributions where the dark solid line is the echo and the light green is the near-end signal.

given in Algorithm 1 uses the inverse of the estimated variance of the near-end excitation signal $w(t)$, i.e., $\hat{\sigma}_w^{-1}(t)$ to account for the variance in the estimation so as to obtain the BLUE of the echo path. $\hat{\sigma}_w(t)$ is estimated directly using the Levinson-Durbin algorithm and is subsequently used in the adaptation gain $\mathbf{G}(\omega, k)$. On the other hand, in the PVSS-APA [16] and VR-APA [19] the near-end signal variance $\sigma_v(t)$ needs to be estimated instead.

The three selected VSS algorithms are implemented using FDAF-PEM-AFROW and show different levels of improvement as compared to the original versions using APA with projection order $K = 4$. In Figure 5.5, the upper part shows the microphone signal contributions where the dark solid line is the echo and the light green is the near-end signal. In Figure 5.6(a) and Figure 5.6(b) the upper part shows the full-length simulation and the bottom part shows a zoom-in coinciding with the DT situation.

- VR-FDAF-PEM-AFROW (Algorithm 2):

In [19], a practical algorithm to design a VR factor for the APA has been

proposed. The condition to derive the VR-APA is to minimize the difference between the estimated and true filter coefficients. For this, it is assumed that the l_2 norm of the *a posteriori* error is equal to the near-end noise signal variance, similar to the condition imposed in [13], [16]. The VR-APA performance has been compared to the performance of existing techniques [17], [18] [37], demonstrating the effectiveness of VR-APA. The implementation of the VR-FDAF-PEM-AFROW is straightforward since the estimate of the near-end excitation signal variance is calculated using the Levinson-Durbin algorithm. The VR parameter is obtained in *line* 8 and included in the normalization factor in *line* 9 as shown in Algorithm 2.

In this particular case, the performance of VR-APA, shown in Figure 5.5, is very poor. The algorithm is difficult to tune such that it has a comparable initial convergence curve as the VR-FDAF-PEM-AFROW. On the other hand, it is obvious that VR-FDAF-PEM-AFROW significantly outperforms VR-APA, turning the latter into an algorithm suitable for DT situations as well.

- PVSS-FDAF-PEM-AFROW (Algorithm 3):

In PVSS-APA [16], the condition that is imposed to preserve the near-end component in the echo-compensated signal is to set the *a posteriori* error equal to the near-end signal. As satisfying this condition is not possible in practice, an estimate of the near-end signal variance is calculated and compared to an estimate of the variance of the error signal [13], [16]. The PVSS-FDAF-PEM-AFROW version of the original algorithm is given in Algorithm 3. The main steps of this algorithm are: (1) the near-end signal variance estimation which in PVSS-FDAF-PEM-AFROW is calculated using the Levinson-Durbin algorithm, and (2) the VSS calculation in *line* 7 which is frequency-dependent in PVSS-FDAF-PEM-AFROW.

The convergence of PVSS-FDAF-PEM-AFROW, shown in Figure 5.6(a), is similar to the convergence for the PVSS-APA. During DT PVSS-FDAF-PEM-AFROW generally outperforms PVSS-APA by $5-7$ dB. Moreover, it seems that PVSS-FDAF-PEM-AFROW sets a lower bound in the MSD, i.e., consistently outperforms PVSS-APA.

- PCVSS-FDAF-PEM-AFROW (Algorithm 4):

PCVSS-APA [11] belongs to the gradient-based VSS algorithms. It appears that PCVSS outperforms the algorithms given in [6] and [38] in DT situations and moreover it does not rely on any signal or system model so it is claimed to be easy to control in practice. The adaptation rate is controlled by a measure of the correlation between instantaneous and long-term averages of the so-called *projection vectors*, i.e., gradient vectors in APA. The three main features of PCVSS-FDAF-PEM-AFROW (Algorithm 4) are represented in (1) *line* 7 where the correlation of the current and past gradient estimates is calculated,

(2) *line* 8 where the current step size is generated and (3) *line* 9 where the control logic to bound its value is applied.The two algorithms, shown in Figure 5.6(b), have a similarly fast initial convergence. During DT, the improvement of PCVSS-FDAF-PEM-AFROW w.r.t. PCVSS-APA becomes apparent. PCVSS-FDAF-PEM-AFROW outperforms PCVSS-APA by $6-8$ dB and it also shows a more stable curve.

5.6.3 Complexity analysis

The complexity analysis in Table 5.2 shows that, for a typical choice of the algorithm parameters, the FDAF-PEM-AFROW algorithms are about 10 times cheaper than the corresponding original algorithms. Hence, it is concluded that the PVSS-FDAF-PEM-AFROW, PCVSS-FDAF-PEM-AFROW and VR-FDAF-PEM-AFROW algorithm are to be preferred over the original algorithms during DT situations. Based on FDAF-PEM-AFROW, a highly robust algorithm has been proposed in [39] using a Wiener variable step size (WISE) and a gradient spectral variance smoothing (GRASS) for DT-robust AEC and for acoustic feedback cancellation (AFC).

5.7 Conclusion

In this chapter, we have proposed a new framework to tackle the problem of double-talk (DT) in acoustic echo cancellation (AEC). It is based on a frequency domain adaptive filtering (FDAF) implementation of the so-called PEM-AFROW algorithm (FDAF-PEM-AFROW). It has been shown that the FDAF-PEM-AFROW minimizes the BLUE criterion and so provides an optimal acoustic echo path estimate during DT. The FDAF-PEM-AFROW algorithm shows an improved performance with respect to the PEM-AFROW and FDAF-NLMS with near-end signal normalization. Although these three algorithms are constructed from the BLUE framework and should therefore obtain the minimum variance echo path estimate during DT, in practice clear differences between them are observed. In PEM-AFROW, it is clear that fulfilling the three conditions to achieve the BLUE in practice is very difficult. This fact affects the PEM-AFROW much more than the FDAF-PEM-AFROW. The FDAF-NLMS with near-end signal normalization performs better than PEM-AFROW as the conditions to achieve the BLUE seem to be less restricting. However, the FDAF-NLMS with near-end signal normalization is still outperformed by FDAF-PEM-AFROW which has the benefits of being implemented in the frequency-domain, featuring a prefilter and including the non-frequency-dependent near-end excitation signal variance estimate. We have shown that the instantaneous pseudo-correlation (IPC) measure between the near-end signal and the loudspeaker signal is significantly reduced when using the combi-

Algorithm 2 Variable regularization [19], using FDAF-PEM-AFROW (VR-FDAF-PEM-AFROW)

1: Initialize: $\boldsymbol{S}_{U_a} = \boldsymbol{S}_{E_a} = \mathbf{0}_{M\times 1}$.
2: **for** $k = 1, 2, ...$ **do**
3: Perform lines $4 - 14$ from Algorithm 1
4: $\boldsymbol{\Phi}(k) = \mathbf{U}_a{}^*(k) \circ \mathbf{E}_a(k)$
5: **for** $m = 0, 1, ..., M-1$ **do**
6: $S_{U_a}(m,k) = \lambda_0 S_{U_a}(m,k-1) + (1-\lambda_0)|U_a(m,k)|^2$
7: $S_{E_a}(m,k) = \lambda_1 S_{E_a}(m,k) + (1-\lambda_1)\left|E_a(m,k)\right|^2$
8: $\alpha(m,k) = S_{U_a}(m,k)\dfrac{\sqrt{\hat{\sigma}_w(k)}}{\sqrt{S_{E_a}(m,k)} - \sqrt{\hat{\sigma}_w(k)}}$
9: $G(m,k) = \left(S_{U_a}(m,k) + \alpha(m,k)\right)^{-1}$
10: **end for**
11: $\hat{\mathbf{F}}(k) = \hat{\mathbf{F}}(k-1) + \mu_0 \mathcal{F}\left\{\left[\left[\mathcal{F}^{-1}\left\{\mathbf{G}(k)\circ\boldsymbol{\Phi}(k)\right\}\right]_{1:N}^T \quad \mathbf{0}_N\right]^T\right\}$
12: **end for**
13: In our simulation we used the following set of parameters: $P = 160$, $N = 80$, $n_A = 1$, $\mu_0 = 0.0325$, $\lambda_0 = 0.95$, $\lambda_1 = 0.9$.

nation of FDAF and PEM, as done in FDAF-PEM-AFROW.

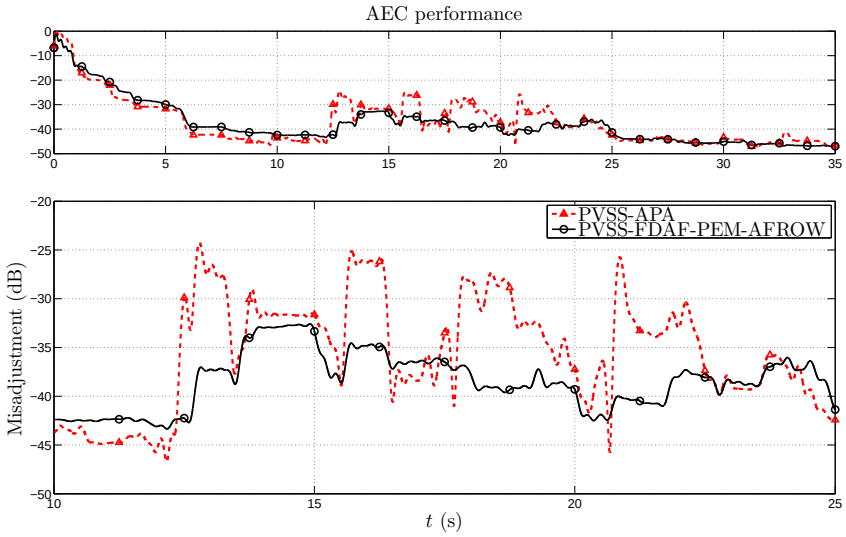
Finally, we have used the FDAF-PEM-AFROW framework to improve several state-of-the-art variable step-size (VSS) and variable regularization (VR) algorithms, in particular, the APA-based projection-correlation VSS (PCVSS-APA), the practical VSS affine projection algorithm (PVSS-APA) and the APA-based VR (VR-APA). It has been shown that the FDAF-PEM-AFROW algorithms significantly outperform the corresponding original algorithms in every simulation by at least 6 dB during DT. In terms of computational complexity the FDAF-PEM-AFROW versions are themselves 10 times cheaper than the original versions.

Algorithm 3 Practical VSS [16] using FDAF-PEM-AFROW (PVSS-FDAF-PEM-AFROW)

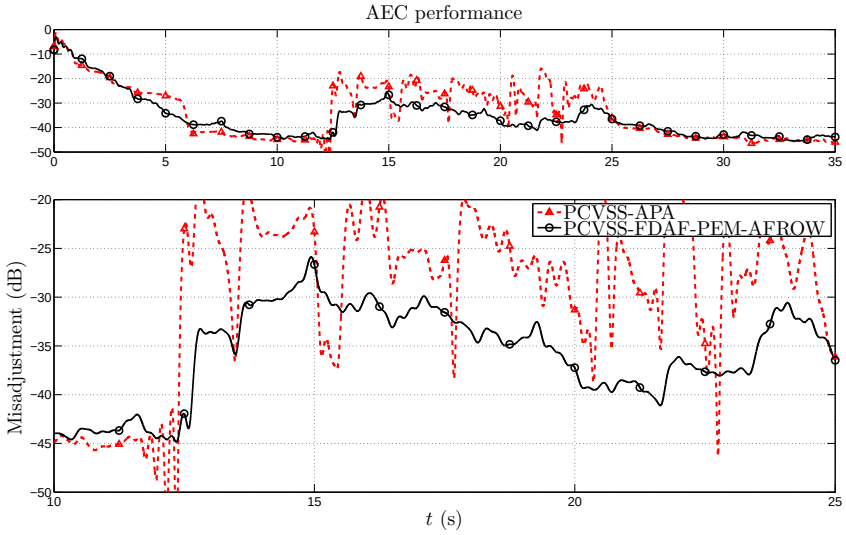
Initialize: $\boldsymbol{S}_{U_a} = \boldsymbol{S}_{E_a} = \mathbf{0}_{M\times 1}$.
for $k = 1, 2, ...$ **do**
  Perform lines $4-14$ from Algorithm 1
  for $m = 0, 1, ..., M-1$ **do**
    $S_{E_a}(m,k) = \lambda_0 S_{E_a}(m,k-1) + (1-\lambda_0)|E_a(m,k)|^2$
    $S_{U_a}(m,k) = \lambda_0 S_{U_a}(m,k-1) + (1-\lambda_0)|U_a(m,k)|^2$
    $\mu_{\text{PVSS}}(m,k) = \mu_0 \left| 1 - \dfrac{\sqrt{\hat{\sigma}_w(k)}}{\sqrt{S_{E_a}(m,k)+\alpha}} \right|$
    $G(m,k) = \left(S_{U_a}(m,k)+\alpha\right)^{-1}$
  end for
  $\hat{\mathbf{F}}(k) = \hat{\mathbf{F}}(k-1) +$
  $\mathcal{F}\left\{ \left[\left[\mathcal{F}^{-1}\left\{ \boldsymbol{\mu}_{\text{PVSS}}(k) \circ \mathbf{G}(k) \circ \mathbf{U}_a{}^*(k) \circ \mathbf{E}_a(k) \right\} \right]_{1:N}^T \quad \mathbf{0}_N \right]^T \right\}$
end for
In our simulations we used the following set of parameters: $P = 160$, $N = 80$, $n_A = 1$, $\lambda_0 = 0.99$, $\alpha = 2e^{-2}$, $\mu_0 = 0.025$, $\alpha = 10^{-8}$.

Algorithm 4 Projection-correlation VSS [11] using FDAF-PEM-AFROW (PCVSS-FDAF-PEM-AFROW)

Initialize: $\mu_{\text{PCVSS}}(0) = \mathbf{I}_{M\times 1}$, $\mathbf{S}_{U_a} = \boldsymbol{\Phi} = \mathbf{C} = \mathbf{0}_{M\times 1}$.
for $k = 1, 2, ...$ **do**
  Perform lines $4-14$ from Algorithm 1
  $\boldsymbol{\Phi}(k) = \mathbf{U}_a{}^*(k) \circ \mathbf{E}_a(k)$
  for $m = 0, 1, ..., M-1$ **do**
    $S_{U_a}(m,k) = \lambda_0 S_{U_a}(m,k-1) + (1-\lambda_0)|U_a(m,k)|^2$
    $C(m,k) = C(m,k-1) + \Phi(m,k) - \Phi(m,k-B)$
    $\mu'(m,k) = \lambda_1\ \mu_{\text{PCVSS}}(m,k-1) + \beta_0\ |C(m,k)|$
    $\mu_{\text{PCVSS}}(m,k) = \begin{cases} 0, & \mu'(m,k) < 0 \\ \mu_0, & \mu'(m,k) > \mu_0 \\ \mu'(m,k), & \text{otherwise} \end{cases}$
    $G(m,k) = \left(S_{U_a}(m,k)+\alpha\right)^{-1}$
  end for
  $\hat{\mathbf{F}}(k) = \hat{\mathbf{F}}(k-1) +$
  $\mathcal{F}\left\{ \left[\left[\mathcal{F}^{-1}\left\{ \boldsymbol{\mu}_{\text{PCVSS}}(k) \circ \mathbf{G}(k) \circ \mathbf{U}_a{}^*(k) \circ \mathbf{E}_a(k) \right\} \right]_{1:N}^T \quad \mathbf{0}_N \right]^T \right\}$
end for
In our simulation we used the following set of parameters: $P = 160$, $N = 80$, $n_A = 1$, $\lambda_0 = 0.95$, $B = 50$, $\beta = \lambda_1 = 0.9$, $\alpha = 1e^{-3}$, $\mu_0 = 0.03$.



(a)



(b)

Figure 5.6: AEC performance using speech signals (far-end and near-end) between 12.5 and 25 s. The original APA and the FDAF-PEM-AFROW versions are compared when bursting DT occurs at -10 dB SER and white near-end noise at 30 dB SNR. In (a) and (b) The upper part shows the full-length simulation and the bottom part shows a zoom-in coinciding with the DT. (a) Practical VSS. (b) Projection-correlation VSS.

Algorithm	Computation	Total
PVSS-APA	$2K^2N + 4KN + 15K + 8$	3908
PVSS-FDAF-PEM-AFROW	(FDAF) $\frac{21M\log_2 M + 34M}{N}+$ (PEM) $+\frac{6M\log_2 M + 2M + n_A^2 + (4+4M)n_A}{N}$	475
PCVSS-APA	$2K^2N + 4KN + 4K + 4N + 12$	4188
PCVSS-FDAF-PEM-AFROW	(FDAF) $\frac{15M\log_2 M + 19M + BM}{N}+$ (PEM) $+\frac{6M\log_2 M + 2M + n_A^2 + (4+4M)n_A}{N}$	367
VR-APA	$2K^2N + 4KN + 4K$	3878
VR-FDAF-PEM-AFROW	(FDAF) $\frac{15M\log_2 M + 31M}{N}+$ (PEM) $+\frac{6M\log_2 M + 2M + n_A^2 + (4+4M)n_A}{N}$	381

Table 5.4: Complexity comparison by the number of FLOPS per recursion. One FFT/IFFT is $3M\log_2 M$ FLOPS, one $\sqrt{\cdot}$ is M FLOPS and $K = 4$.

Appendix

5.A Instantaneous pseudo-correlation calculation

The following recursions show how the signal $z(t)$ for the IPC measure is calculated in different adaptive filtering algorithms, namely, in NLMS, PEM-AFROW, FDAF-NLMS and FDAF-PEM-AFROW.

We first consider the NLMS update equation given as

$$e(t) = y(t) - \hat{\mathbf{f}}(t-1)\mathbf{u}(t) \tag{5.28}$$

$$\hat{\mathbf{f}}(t) = \hat{\mathbf{f}}(t-1) + \mu \frac{e(t)}{\alpha + \mathbf{u}^T(t)\mathbf{u}(t)}\mathbf{u}(t). \tag{5.29}$$

A derivation similar to (5.22) and (5.24) then leads to

$$\begin{aligned} e_v(t) &= e(t) - e_x(t) \\ &= v(t) - \mu \sum_{i=1}^{t-1} \frac{e_v(i)}{\alpha + \mathbf{u}^T(i)\mathbf{u}(i)} \left(\mathbf{u}^T(i)\mathbf{u}(t)\right) \end{aligned} \tag{5.30}$$

and

$$z(t) = \frac{\mu}{N} \sum_{i=1}^{t-1} \frac{e_v(i)}{\alpha + \mathbf{u}^T(i)\mathbf{u}(i)} \left(\mathbf{u}^T(i)\mathbf{u}(t)\right) \tag{5.31}$$

$$= \frac{1}{N}\tilde{\mathbf{f}}^T(t-1)\mathbf{u}(t) \tag{5.32}$$

with $\alpha = 10$. The recursion to compute $z(t)$ in PEM-AFROW can be derived similarly where the NLMS is then a special case with $\hat{\mathbf{a}} = [1 \quad \mathbf{0}_{n_A}] \quad \forall t$

$$\hat{\mathbf{a}} = \text{Levison-Durbin}\left\{\mathbf{v}(t), n_A\right\} \quad \text{Computed every P samples.} \tag{5.33}$$

$$v_a(t) = \hat{\mathbf{a}}^T \bar{\mathbf{v}}(t) \tag{5.34}$$

$$u_a(t) = \hat{\mathbf{a}}^T \bar{\mathbf{u}}(t) \tag{5.35}$$

$$e_{va}(t) = v_a(t) - \tilde{\mathbf{f}}^T(t-1)\mathbf{u}_a(t) \tag{5.36}$$

$$g(t) = \frac{1}{\mathbf{u}_a^T(t)\mathbf{u}_a(t) + \alpha} \tag{5.37}$$

$$\tilde{\mathbf{f}}(t) = \tilde{\mathbf{f}}(t-1) + g(t)\mathbf{u}_a(t)e_{va}(t) \tag{5.38}$$

$$z(t) = \frac{1}{N}\tilde{\mathbf{f}}^T(t-1)\mathbf{u}_a(t), \tag{5.39}$$

where, in this case $\mathbf{v}(t) = [v(t), ..., v(t-P+1)]^T$, $\bar{\mathbf{v}}(t) = [v(t), ..., v(t-n_A+1)]^T$, $\bar{\mathbf{u}}(t) = [u(t), ..., u(t-n_A+1)]^T$ and $\mathbf{u}_a(t) = [u_a(t), ..., u_a(t-N+1)]$.

The recursion to compute $z(t)$ in FDAF-PEM-AFROW can be derived similarly where the FDAF-NLMS is then a special case with $\hat{\mathbf{a}} = [1 \quad \mathbf{0}_{n_A}] \quad \forall t$ with $\lambda = 0.99$ is written as,

$$\hat{\mathbf{a}}(k) = \text{Levison-Durbin}\left\{[\mathbf{v}(k)]_{1:P}, n_A\right\} \tag{5.40}$$

$$u_a(m,k) = [u(kN+1+m), ..., u(kN+1+m-n_A)]\hat{\mathbf{a}}(k) \quad m = 1, ..., M \tag{5.41}$$

$$v_a(m,k) = [v(kN+1+m), ..., v(kN+1+m-n_A)]\hat{\mathbf{a}}(k) \quad m = 1, ..., M \tag{5.42}$$

$$\mathbf{U}_a(k) = \frac{1}{\sqrt{M}}\mathcal{F}\left\{\mathbf{u}_a^T(k)\right\} \tag{5.43}$$

$$\tilde{\mathbf{e}}_a(k) = \left[\mathbf{v}_a(k) - \mathcal{F}^{-1}\left\{\tilde{\mathbf{F}}(k-1) \circ \mathbf{U}_a(k)\right\}\sqrt{M}\right]_{N+1:M} \tag{5.44}$$

$$\mathbf{e}_{\mathbf{v}a}(k) = \frac{1}{\sqrt{M}}\mathcal{F}\left\{[\mathbf{0}_N \quad \tilde{\mathbf{e}}_a(k)]\right\} \tag{5.45}$$

$$S_{U_a}(m,k) = \lambda S_{U_a}(m,k-1) + (1-\lambda)|U_a(m,k)|^2 \quad m = 1, ..., M \tag{5.46}$$

$$G(m,k) = \left(S_{U_a}(m,k) + \alpha\right)^{-1} \quad m = 1, ..., M \tag{5.47}$$

$$\tilde{\mathbf{F}}(k) = \tilde{\mathbf{F}}_a(k-1) \tag{5.48}$$

$$+ \frac{1}{\sqrt{M}}\mathcal{F}\left\{[\mathbf{I}_N \quad \mathbf{0}_N]^T \mathcal{F}^{-1}\left\{\mathbf{G}(k) \circ \tilde{\mathbf{E}}_a(k) \circ \mathbf{U}_a^*(k)\right\}\sqrt{M}\right\} \tag{5.49}$$

$$\mathbf{z}(k) = \frac{1}{N}\left[\mathcal{F}^{-1}\left\{\tilde{\mathbf{F}}(k-1) \circ \mathbf{U}_a(k)\right\}\sqrt{M}\right]_{N+1:M} \tag{5.50}$$

Here $(\cdot)^*$ represents a complex conjugation operation and $\mathbf{v}(k) = [v(kN-N+1), ..., v(kN+N)]^T$. It should be noted that the IPC measure is calculated using the signal $v_a(t)$ instead of $v(t)$, as given in (5.26).

Bibliography

[1] J. Benesty, T. Gänsler, D. R. Morgan, M. M. Sondhi, and S. L. Gay, *Advances in Network and Acoustic Echo Cancellation*. Berlin: Springer-Verlag, 2001.

[2] P. Gil-Cacho, T. van Waterschoot, M. Moonen, and S. H. Jensen, "Multi-microphone acoustic echo cancellation using multi-channel warped linear prediction of common acoustical poles," in *Proc. 18th European Signal Process. Conf. (EUSIPCO'10)*, Aalborg, Denmark, Aug. 2010, pp. 2121–2125.

[3] T. van Waterschoot, G. Rombouts, P. Verhoeve, and M. Moonen, "Double-talk-robust prediction error identification algorithms for acoustic echo cancellation," *IEEE Trans. Signal Process.*, vol. 55, no. 3, pp. 846–858, Mar. 2007.

[4] S. Haykin, *Adaptive Filter Theory*. Upper Saddle River, NY: Prentice Hall, 2002.

[5] P. S. R. Diniz, *Adaptive filtering: Algorithms and Practical Implementations*. Boston, MA: Springer, 2008.

[6] V. J. Mathews and Z. Xie, "A stochastic gradient adaptive filter with gradient adaptive step size," *IEEE Trans. Signal Process.*, vol. 41, no. 6, pp. 2075–2087, June 1993.

[7] Y. Zhang, J. A. Chambers, W. Wang, P. Kendrick, and T. J. Cox, "A new variable step-size LMS algorithm with robustness to nonstationary noise," in *Proc. 2007 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'07)*, vol. 3, Honolulu, Hawaii, USA, Apr. 2007, pp. 1349–1352.

[8] W.-P. Ang and B. Farhang-Boroujeny, "A new class of gradient adaptive step-size LMS algorithms," *IEEE Trans. Signal Process.*, vol. 49, no. 4, pp. 805–810, Apr. 2001.

[9] H.-C. Shin, A. H. Sayed, and W.-J. Song, "Variable step-size NLMS and affine projection algorithms," *IEEE Signal Process. Lett.*, vol. 11, no. 2, pp. 132–135, Feb. 2004.

[10] Y. Zhang, N. Li, J. A. Chambers, and Y. Hao, "New gradient-based variable step size LMS algorithms," *EURASIP J. Advances Signal Process.*, vol. 2008, no. 105, Jan. 2008.

[11] T. Creasy and T. Aboulnasr, "A projection-correlation algorithm for acoustic echo cancellation in the presence of double talk," in *Proc. 2000 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'00)*, vol. 1, Istanbul, Turkey, June 2000, pp. 436–439.

[12] K. Ozeki and T. Umeda, "An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties," *Electronics and Communication in Japan*, vol. 67, no. 5, pp. 19–27, Aug. 1984.

[13] J. Benesty, H. Rey, L. R. Vega, and S. Tressens, "A nonparametric VSS NLMS algorithm," *IEEE Signal Process. Lett.*, vol. 13, no. 10, pp. 581–584, Oct. 2006.

[14] C. Paleologu, S. Ciochină, and J. Benesty, "Double-talk robust VSS-NLMS algorithm for under-modeling acoustic echo cancellation," in *Proc. 2008 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'08)*, Las Vegas, USA, Mar. 2008.

[15] M. A. Iqbal and S. L. Grant, "Novel variable step size NLMS algorithms for echo cancellation," in *Proc. 2008 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'08)*, Las Vegas, USA, Mar. 2008, pp. 241 – 244.

[16] C. Paleologu, J. Benesty, and S. Ciochină, "A variable step-size affine projection algorithm designed for acoustic echo cancellation," *IEEE Trans. Audio Speech Lang. Process.*, vol. 16, no. 8, pp. 1466 – 1478, Nov. 2008.

[17] H. Rey, L. R. Vega, S. Tressens, and J. Benesty, "Variable explicit regularization in affine projection algorithm: Robustness issues and optimal choice," *IEEE Trans. Signal Process.*, vol. 55, no. 5, pp. 2096 – 2108, May 2007.

[18] Y.-S. Choi, H.-C. Shin, and W.-J. Song, "Adaptive regularization matrix for affine projection algorithm," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 54, no. 12, pp. 1087–1091, Dec. 2007.

[19] W. Yin and A. S. Mehr, "A variable regularization method for affine projection algorithm," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 6, pp. 476 – 480, June 2010.

[20] S. M. Kay, *Fundamentals of statistical signal processing: Estimation theory.* Upper Saddle River, New Jersey: Prentice Hall, 1993.

[21] T. van Waterschoot and M. Moonen, "Double-talk robust acoustic echo cancellation with continuous near-end activity," in *Proc. 13th European Signal Process. Conf. (EUSIPCO '05)*, Antalya, Turkey, Sep. 2005.

[22] L. Ljung, *System Identification: Theory for the user.* Englewood Cliffs, New Jersey: Prentice Hall, 1987.

[23] G. Rombouts, T. van Waterschoot, K. Struyve, and M. Moonen, "Acoustic feedback cancellation for long acoustic paths using a nonstationary source model," *IEEE Trans. Signal Process.*, vol. 54, no. 9, pp. 3426–3434, Sep. 2006.

[24] T. Trump, "A frequency domain adaptive algorithm for colored measurement noise environment," in *Proc. 1998 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'98)*, vol. 3, Seattle, Washington, USA, May 1998, pp. 1705 – 1708.

[25] ——, "Accounting for measurement noise color in frequency domain adaptive algorithms," in *Proc. Thirty-Second Asilomar Conf. Signals Syst. Comp.*, vol. 1, Pacific Grove, CA, USA, May 1998, pp. 508 – 512.

[26] G. A. Clark, S. R. Parker and S. K. Mitra, "A Unified Approach to Time- and Frequency-Domain Realization of FIR Adaptive Digital Filters," *IEEE Trans. Acoust. Speech Signal Process.*, Vol. 31, pp.1073-1083, Oct. 1983.

[27] T. Kariya, H. Kurata, *Generalised Least Squares.* John Wiley & Sons, 2004.

[28] T. Söderström and P. Stoica *System identification*, Prentice Hall, 1988.

[29] H. Buchner, J. Benesty and W. Kellermann, "Generalized multichannel frequency-domain adaptive filtering: efficient realization and application to hands-free speech communication," *Signal Process.*, vol. 85, pp. 549-570, Sep. 2005.

[30] P. Loizou, *Speech Enhancement: Theory and Practice.* Boca Raton, Florida: Taylor and Francis, 2007.

[31] A. Spriet, I. Proudler, M. Moonen and J. Wouters, "Adaptive feedback cancellation in hearing aids with linear prediction of the desired signal," *IEEE Trans. Signal Process.*, vol. 53, no. 10, pp. 3749–3763, Oct. 2005.

[32] R. M. Gray, "On the Asymptotic Eigenvalue Distribution of Toeplitz Matrices," *IEEE Trans. on Information Theory*, vol.18, no.6, pp. 725-730, Nov. 1972.

[33] T. van Waterschoot and M. Moonen, "Fifty years of acoustic feedback control: state of the art and future challenges," *Proc. IEEE*, vol. 99, no. 2, pp. 288–327, Feb. 2011.

[34] B. Widrow and M. Hoff, "Adaptive switching circuits," in *Proc. WESCON Conv. Rec.*, vol. 4, 1960, pp. 96–140.

[35] J. M. Gil-Cacho, T. van Waterschoot, M. Moonen, and S. H. Jensen, "Transform domain prediction error method for improved acoustic echo and feedback cancellation," in *Proc. 20th European Signal Process. Int. Conf. (EUSIPCO'12)*, Bucharest, Rumania, Aug. 2012, pp. 2422–2426.

[36] J. J. Shynk, "Frequency-domain and multirate adaptive filtering," *IEEE Signal Process. Mag.*, vol. 9, no. 1, pp. 14–37, Jan. 1992.

[37] V. Myllylä and G. Schmidt, "Pseudo-optimal regularization for affine projection algorithms," in *Proc. 2002 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'02)*, vol. 2, Orlando, Florida, May 2002, pp. 1917–1920.

[38] C. Rohrs and R. Younce, "Double talk detector for echo canceler and method," US Patent US 4 918 727, 04 17, 1990. [Online]. Available: http://www.patentlens.net/patentlens/patent/US4918727/

[39] J. M. Gil-Cacho, T. van Waterschoot, M. Moonen, and S. H. Jensen, "Wiener variable step size and gradient spectral variance smoothing for double-talk-robust acoustic echo cancellation and acoustic feedback cancellation," *Submitted to Elsevier Signal Process.*, June 7, 2013. pp. 1–30.

Chapter 6

Wiener variable step size and gradient spectral variance smoothing for double-talk-robust acoustic echo cancellation and acoustic feedback cancellation

Wiener variable step size and gradient spectral variance smoothing for double-talk-robust acoustic echo cancellation and acoustic feedback cancellation

Jose Manuel Gil-Cacho, Toon van Waterschoot, Marc Moonen and Søren Holdt Jensen

Submitted to Elsevier Signal Processing, May. 2013.

Contributions of first author

- literature study
- co-development of the WISE-GRASS-FDAF-PEM-AFROW algorithm
- design of simulation scenarios
- software implementation and computer simulations
- co-interpretation of simulation results
- text redaction and editing

Abstract

In this chapter, we derive a robust and computationally efficient algorithm based on the frequency-domain adaptive filter prediction error method using row operations (FDAF-PEM-AFROW) for DT-robust AEC and for AFC. The proposed algorithm features two main modifications: (a) the WIener variable Step sizE (WISE), and (b) the GRAdient Spectral variance Smoothing (GRASS). In AEC simulations, the WISE-GRASS-FDAF-PEM-AFROW algorithm obtains improved robustness and smooth adaptation in highly adverse scenarios such as in bursting DT at high levels, and in a change of acoustic path during continuous DT. Similarly, in AFC simulations, the algorithm outperforms state-of-the-art algorithms when using a low-order near-end speech signal model and in colored non-stationary noise.

6.1 Introduction

ACOUSTIC echo and acoustic feedback are two well-known problems appearing in speech communication systems, which are caused by the acoustic coupling between a loudspeaker and a microphone. On the one hand, acoustic echo cancellation (AEC) is widely used in mobile and hands-free telephony [1] where the existence of echoes degrades the intelligibility and listening comfort. On the other hand, acoustic feedback limits the maximum amplification that can be applied, e.g., in a hearing aid, before howling due to instability, appears [2], [3]. This maximum amplification may be too small to compensate for the hearing loss, which makes acoustic feedback cancellation (AFC) an important component in hearing aids. Figure 6.1 shows the typical set-up for AEC and AFC.

The goal of AEC and AFC is essentially to identify a model for the echo or feedback path, i.e., the room impulse response (RIR), and to estimate the echo or feedback signal which is then subtracted from the microphone signal. The microphone signal is given by $y(t) = x(t)+v(t)+n(t) = F(q,t)u(t)+v(t)+n(t)$ where q denotes the time shift operator, e.g., $q^{-k}u(t) = u(t-k)$, t is the discrete time variable, $x(t)$ is the echo or feedback signal, $u(t)$ is the loudspeaker signal, $v(t)$ is the near-end speech signal and $n(t)$ is the near-end noise signal. In the sequel, we will use the term near-end signal to refer to $v(t)$ and/or $n(t)$ if, according to the context, there is no need to point out a difference. $F(q,t) = f_0(t) + f_1(t)q^{-1} + ... + f_{n_F}(t)q^{-n_F}$ represents a linear time-varying model of the RIR between the loudspeaker and the microphone, where n_F is the RIR model order. Therefore, the aim of AEC or AFC is to obtain an estimate $\hat{F}(q,t)$ of the RIR model $F(q,t)$ by means of an adaptive filter, which is steered by the error signal $e(t) = \left[F(q,t) - \hat{F}(q,t)\right] u(t) + v(t) + n(t)$.

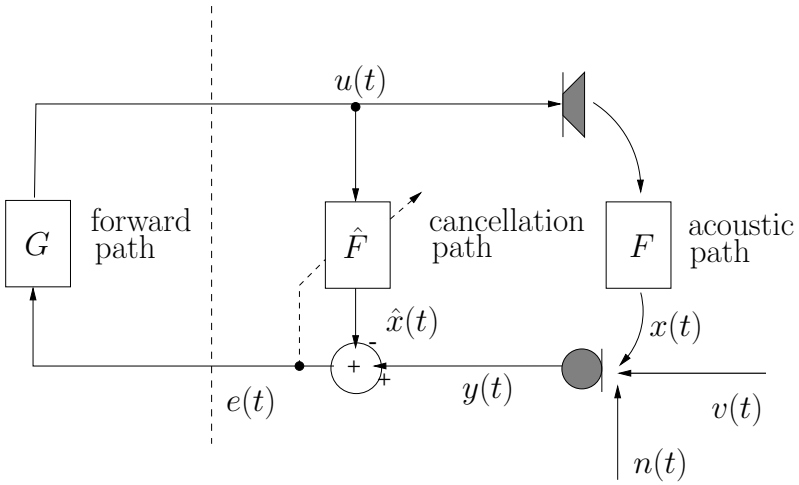


Figure 6.1: Typical set-ups for AEC/AFC. The left part (forward path) only relates to AFC. The right part relates to both AEC and AFC.

In AEC applications the loudspeaker signal $u(t)$ is considered to be the signal coming from the far-end side, i.e., the far-end signal. The *echo-compensated* error signal $e(t)$ is then transmitted to the far-end side. In AFC applications, on the other hand, the forward path $G(q,t)$ maps the *feedback-compensated* error signal $e(t)$ to the loudspeaker signal, i.e., $u(t) = G(q,t)e(t)$. Typically, $G(q,t)$ consists of an amplifier with a (possibly time-varying) gain $K(t)$ cascaded with a linear equalization filter $J(q,t)$ such that $G(q,t) = J(q,t)K(t)$. AEC and AFC in principle look the same and share many common characteristics, however different essential problems can be distinguished:

6.1.1 The problem of double-talk in acoustic echo cancellation

Practical AEC implementations rely on computationally simple stochastic gradient algorithms, such as the normalized least mean squares (NLMS) algorithms, which may be very sensitive to the presence of a near-end signal [4]. Especially near-end speech, in a so-called double-talk (DT) scenario, will affect the adaptation in the AEC context by making the adaptive filter converge slowly or even diverge.

To tackle the DT problem, adaptive filters have been equipped with DT detectors (DTDs) to switch off during DT periods. Since the Geigel algorithm [5] was proposed, several other DTD algorithms have been specifically designed for AEC applications, e.g., [6]. However, in general, DTDs take some time until an onset of a DT period is detected. Moreover, in some AEC scenarios the near-

end signal will be continuously present and then the use of a DTD becomes futile. This may be the case for example in a noisy teleconferencing or AFC application. Therefore DT-robust algorithms without the need for a DTD are called for. DT-robustness may be achieved based on three approaches namely: (1) by using a *postfilter* to suppress or enhance residual echo, (2) by using a *variable step size* (VSS) to slow down adaption during DT, (3) by prefiltering the loudspeaker and microphone signal with a decorrelation filter to achieve a *minimum-variance* RIR estimate.

The first approach consists in the use of a postfilter which interplays with the AEC to suppress residual echo (and also to reduce near-end signals) based on signal enhancement techniques [7]- [9]. On the other hand, in [10], [11], the idea behind the postfilter design is the opposite, i.e., to the enhance the residual echo in the adaptive filter loop. The postfilter design, in any case, is typically based on single-channel noise reduction techniques, which carry a trade-off between residual echo/noise reduction and signal distortion.

The second approach to DT-robust AEC is to equip the (stochastic gradient) adaptive filter with a VSS which may be derived using information about the gradient vector or information about the near-end signal power. The first type of VSS algorithms rely on two properties of the gradient vector to control the step size [12]- [16]: (1) the property that the norm of the gradient vector will be large initially and converge to a small value, ideally zero, at steady state, (2) the property that the gradient vector direction will generally show a coherent trend during initial convergence in contrast to a random trend around the optimal value during DT and at steady state. From this class of algorithms, the only one specifically designed for DT-robust AEC is the *projection-correlation* VSS (PC-VSS) which has been proposed in [17]. PC-VSS is a VSS algorithm based on the affine projection algorithm (AP) [18] where the adaptation rate is controlled by a measure of the correlation between instantaneous and long-term averages of the so-called *projection vectors*, i.e., gradient vectors in APA, which allows to achieve robustness and to distinguish between an RIR change and DT. PC-VSS is chosen as one of the competing algorithms in this chapter and hence further explanation will be given in Section 6.4.1.

Recently, more effort has been spent to steer VSS algorithm design towards DT-robust AEC. Some of these recent algorithms are based on the *non-parametric* VSS (NPVSS) algorithm proposed in [19]. The NPVSS algorithm was developed in a system identification context, aiming to recover the system noise (i.e., near-end noise) from the error signal of the adaptive filter when updated with the NLMS algorithm. Inspired by this idea, several approaches have focused on applying the NPVSS algorithm to real AEC applications where the microphone signal also contains a near-end speech signal. Consequently, different VSS-NLMS algorithms have been successfully developed for DT-robust AEC, e.g., [20], [21]. Their convergence, however, is slow in practice and hence, an APA version of the VSS-NLMS algorithm in [21] has been proposed in [22] to

increase the convergence speed. The resulting *practical* VSS affine projection algorithm (PVSS) [22] is chosen as one of the competing algorithms in this chapter and will be further explained also in Section 6.4.1.

The third approach is to search for the optimal AEC solution in a minimum-variance linear estimation framework, rather than in a traditional least squares (LS) framework. The minimum-variance RIR estimate, which is also known as the *best linear unbiased estimate* (BLUE) [23], depends on the near-end signal characteristics, which are in practice unknown and time-varying [4], [24]. The algorithms in [4], [24] aim to whiten the near-end speech signal component in the microphone signal by using adaptive decorrelation filters that are estimated concurrently with the RIR. In order to achieve the BLUE, it is also necessary to add a scaled version of the near-end speech excitation signal variance to the denominator of the stochastic gradient update equation. The use of the prediction error method (PEM) approach [25] was proposed to jointly estimate the RIR and an autoregressive (AR) model of the near-end speech signal. Among the PEM-based algorithms proposed in [4], [24], the PEM-based adaptive filtering using row operations (PEM-AFROW) [26] is particularly interesting because it efficiently uses the Levison-Durbin algorithm to estimate both the near-end speech signal AR model coefficients and the near-end speech excitation signal variance. Thus the algorithms in [4], [24] can be seen as belonging to a new family aiming at both reducing the correlation between the near-end speech signal and loudspeaker signal, and minimizing the RIR estimation variance.

6.1.2 The problem of correlation in acoustic feedback cancellation

In the AFC set-up, the near-end speech signal will be continuously present so using a DTD is pointless. However the main problem in AFC is the correlation that exists between the near-end speech signal component in the microphone signal and the loudspeaker signal itself. This correlation problem, which is caused by the closed loop, makes standard adaptive filtering algorithms to converge to a *biased* solution [2], [27]. This means that the adaptive filter does not only predict and cancel the feedback component in the microphone signal, but also part of the near-end speech signal. This generally results in a distorted *feedback-compensated* error signal. One approach to reduce the bias in the feedback path model identification is to prefilter the loudspeaker and microphone signal with the inverse near-end speech signal model, which is estimated jointly with the adaptive filter [2], [27] using the PEM [25]. For a near-end speech signal, an AR model is commonly used [2] as this yields a simple finite impulse response (FIR) prefilter. However, the AR model fails to remove the speech periodicity, which causes the prefiltered loudspeaker signal still to be correlated with the prefiltered near-end speech signal during voiced speech segments. More advanced models using different cascaded near-end

speech signal models have been proposed to remove the coloring and periodicity in voiced as well as unvoiced speech segments. The constrained pole-zero linear prediction (CPZLP) model [28], the pitch prediction model [3], and the sinusoidal model [29] are examples of alternative models used in recently proposed algorithms. However the overall algorithm complexity typically increases significantly when using cascaded near-end speech signal models [30]. In [31], a Transform-Domain PEM-AFROW (TD-PEM-AFROW) algorithm has been proposed to improve the performance of an AFC without the need for cascaded and computationally intensive near-end signal models. Significant improvement was achieved w.r.t. standard PEM-AFROW even using low-order AR models. PEM-AFROW and TD-PEM-AFROW are chosen as competing algorithms for AFC. TD-PEM-AFROW was also successfully applied to DT-robust AEC in [31] and it is, therefore, also chosen as a competing algorithm for AEC.

6.1.3 Contributions and outline

In [32], we have proposed the use of the FDAF-PEM-AFROW framework to improve several VSS and variable regularization (VR) algorithms. The improvement is basically due to two aspects, (1) instantaneous pseudo-correlation (IPC) between the near-end signal and the far-end signal is heavily reduced when using FDAF-PEM-AFROW compared to stochastic gradient and (time-domain) PEM-AFROW algorithms [32] and (2) FDAF itself may be seen to minimize a BLUE criterion if a proper normalization factor is used during adaptation [33]. In this chapter, we propose two modifications of the FDAF-PEM-AFROW algorithm for robust and smooth adaptation both in AFC and in AEC in continuous DT and bursting DT without the need of a DTD. In particular, we propose the WIener variable Step sizE (WISE) and the GRAdient Spectral variance Smoothing (GRASS) to be performed in FDAF-PEM-AFROW leading to the WISE-GRASS-FDAF-PEM-AFROW algorithm. The WISE modification is implemented as a single-channel noise reduction Wiener filter applied to the (prefiltered) microphone signal. The *Wiener filter gain* [9] is used as a VSS in the adaptive filter, rather than as a signal enhancement parameter. On the other hand, the GRASS modification aims at reducing the variance in the noisy gradient estimates based on time-recursive averaging of instantaneous gradients. The combination of the WISE and the GRASS into the FDAF-PEM-AFROW algorithm consequently gathers the best characteristics we are seeking for in an algorithm both for AEC and AFC, namely, decorrelation properties (PEM, FDAF), minimum variance (GRASS, FDAF, PEM), variable step size (WISE), and computational efficiency (FDAF).

The outline of the chapter is as follows. In Section 6.2, we briefly present the PEM, we provide a simple algorithm description and explain the choice of the near-end speech signal model. In Section 6.3, the proposed algorithm is presented with in-depth explanations about the novel algorithm modifications. The motivation for including these modifications is also justified for

DT-robust AEC and AFC. In Section 6.4, computer simulation results are provided to verify the performance of the proposed algorithm compared to three competing algorithms, in particular, the PC-VSS [17], the PVSS [22] and the TD-PEM-AFROW [31] algorithm. A description of the competing algorithms is provided together with a computational complexity analysis. The Matlab files implementing all the algorithms and those generating the figures in Section 6.3-6.4 can be found online [1]. Finally, Section 6.5 concludes the chapter.

6.2 Prediction error method

The PEM-based AEC/AFC is shown in Figure 6.2. It relies on a linear model for the near-end speech signal $v(t)$, which in Figure 6.2 is specified as

$$v(t) = H(q,t)w(t) \tag{6.1}$$

where $H(q,t)$ contains the filter coefficients of the linear model and $w(t)$ represents the excitation signal which is assumed to be a white noise signal with time-dependent variance $\sigma_w^2(t)$ i.e.,

$$\mathcal{E}\{w(t)w(t-k)\} = \sigma_w^2(t)\delta(k) \tag{6.2}$$

where $\mathcal{E}\{\cdot\}$ is the expected value operator. The near-end noise $n(t)$ is also assumed to be a white noise signal for the time being. As outlined before, a minimum-variance RIR model (in AEC) or an unbiased feedback path model (in AFC) can be identified by first prefiltering the loudspeaker signal $u(t)$ and the microphone signal $y(t)$ with the inverse near-end speech signal model $H^{-1}(q,t)$ before feeding these signals to the adaptive filtering algorithm. As $H^{-1}(q,t)$ is obviously unknown, the near-end speech signal model and the echo/feedback path model have to be jointly identified using the PEM [25]. A common approach in PEM-based AEC/AFC is to model the near-end speech signal with an AR model i.e.,

$$y(t) = x(t) + v(t) + n(t) \tag{6.3}$$

$$= F(q,t)u(t) + \frac{1}{A(q,t)}w(t) + n(t) \tag{6.4}$$

with $F(q,t)$ defined previously and $A(q,t)$ given as

$$A(q,t) = 1 + a_1(t)q^{-1} + ... + a_{n_A}(t)q^{-n_A} \tag{6.5}$$

where n_A is the AR model order.

[1] $http://homes.esat.kuleuven.be/\sim dspuser/abstract12-214\_2.html$

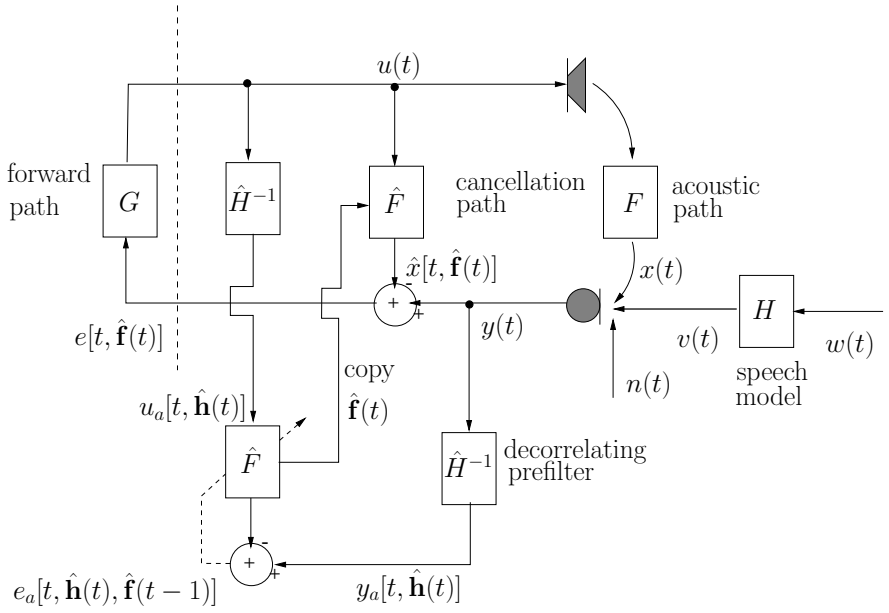


Figure 6.2: AEC/AFC with prefiltering of the loudspeaker and microphone signal using the inverse $[H^{-1}(q,t)]$ of the near-end speech signal model $[H(q,t)]$.

The PEM gives an estimate of the models $F(q,t)$ and $A(q,t)$ by minimization of the prediction error criterion

$$\hat{\vartheta}(t) = \arg\min_{\boldsymbol{\vartheta}(t)} \sum_{i=1}^{t} e_a^2 \left[i, \boldsymbol{\vartheta}(t)\right] \tag{6.6}$$

where the prediction error is defined as

$$e_a \left[i, \boldsymbol{\vartheta}(t)\right] = A(q,t) \left[y(i) - F(q,t)u(i)\right] \tag{6.7}$$

and the parameter vector $\boldsymbol{\vartheta}(t) = \left[\mathbf{f}^T(t), \mathbf{a}^T(t)\right]^T$ contains the parameters of the echo or feedback path model and the near-end speech signal model i.e.,

$$\mathbf{f}(t) = [f_0(t), f_1(t), ..., f_{n_F}(t)]^T, \tag{6.8}$$

$$\mathbf{a}(t) = [1, a_1(t), ..., a_{n_A}(t)]^T, \tag{6.9}$$

Note that throughout the chapter we assume a sufficient-order condition for the acoustic path model (i.e., $n_{\hat{F}} = n_F$). An additional assumption is that the near-end speech signal $v(t)$ is short-term stationary, which implies that the near-end speech signal model $A(q,t)$ does not need to be re-estimated at each time instant t. That is, instead of identifying the near-end speech signal model recursively, it can also be identified non-recursively on a batch of loudspeaker

Algorithm 1.1. First part of the WISE-GRASS-FDAF-PEM-AFROW algorithm for AEC/AFC.
The first part shows the FDAF-PEM-AFROW, whereas the second part shows the WISE-GRASS.

Initialize: K, $k = 0$ and $\hat{\mathbf{P}}_{U_a} = \hat{\mathbf{P}}_{X_a} = \hat{\mathbf{P}}_{D_a} = \hat{\mathbf{F}} = \boldsymbol{\nabla} = \mathbf{0}_{M\times 1}$
$[\mathbf{U}(k) = \mathcal{F}\{\mathbf{u}(k)\}]$
$\hat{\mathbf{x}}(k) = \mathcal{F}^{-1}\{\mathbf{U}(k) \circ \hat{\mathbf{F}}(k-1)\}$ (Echo estimation)
$\left[\mathbf{x}(k) = \mathcal{F}^{-1}\{\mathbf{U}(k) \circ \mathbf{F}\}\right.$ (True echo signal simulation)$\left.\right]$
$[\mathbf{y}(k) = \mathbf{x}(k) + \mathbf{v}(k) + \mathbf{n}(k)$ (Microphone signal simulation)$]$
$\mathbf{e}(k) = [\mathbf{y}(k) - \hat{\mathbf{x}}(k)]_{N+1:M}$ (Error signal)
$[\mathbf{u}(k+1) = K\ \mathbf{e}(k)$ (Loudspeaker signal simulation. Only for AFC)$]$
for $k = 1, 2, ...$ **do**
    $[\mathbf{U}(k) = \mathcal{F}\{\mathbf{u}(k)\}]$
    $\hat{\mathbf{x}}(k) = \mathcal{F}^{-1}\{\mathbf{U}(k) \circ \hat{\mathbf{F}}(k-1)\}$ (Echo estimation)
    $\left[\mathbf{x}(k) = \mathcal{F}^{-1}\{\mathbf{U}(k) \circ \mathbf{F}\}\right.$ (True echo signal simulation)$\left.\right]$
    $[\mathbf{y}(k) = \mathbf{x}(k) + \mathbf{v}(k) + \mathbf{n}(k)$ (Microphone signal simulation)$]$
    $\mathbf{e}(k) = [\mathbf{y}(k) - \hat{\mathbf{x}}(k)]_{N+1:M}$
    $[\mathbf{u}(k+1) = K\ \mathbf{e}(k)$ (Loudspeaker signal simulation. Only for AFC)$]$
    $\hat{\mathbf{a}}(k) = \mathrm{AR}\{\left[\mathbf{e}^T(k) \quad \mathbf{e}^T(k-1)\right]_{1:P}^T, n_A\}$ (Order n_A AR coefficients estimation)
    for $m = 0, ..., M-1$ **do** (Decorrelation prefilter)
        $u_a(m,k) = [u(kN+1+m), ..., u(kN+1+m-n_A)]\mathbf{a}(k)$
        $y_a(m,k) = [y(kN+1+m), ..., y(kN+1+m-n_A)]\mathbf{a}(k)$
    end for
    $\mathbf{U}_a(k) = \mathcal{F}\{\mathbf{u}_a(k)\}$
    $\hat{\mathbf{x}}_a(k) = \mathcal{F}^{-1}\{\mathbf{U}_a(k) \circ \hat{\mathbf{F}}(k-1)\}$
    $\mathbf{e}_a(k) = [\mathbf{y}_a(k) - \hat{\mathbf{x}}_a(k)]_{N+1:M}$ [(Prediction) Error signal (6.7)]
    $\mathbf{E}_a(k) = \mathcal{F}\{[\mathbf{0}_N \quad \mathbf{e}_a^T(k)]^T\}$
(... Continue in Algorithm 1.2 ...)

and microphone data. This is the idea behind the PEM-AFROW algorithm which estimates $A(q,t)$ in a block-based manner, using blocks that approximates the stationary interval of speech. The PEM-AFROW algorithm was originally developed in an AFC framework [26] and applied to a continuous-DT AEC scenario in [24]. It performs only row operations on the loudspeaker data matrix, hence the name PEM-AFROW, and both $\hat{\mathbf{a}}(t)$ and $\hat{\sigma}_w^2(t)$ are efficiently calculated using the Levinson-Durbin recursion. For a detailed description of the original PEM-AFROW algorithm the reader is referred to [26].

6.3 Proposed WISE-GRASS Modifications in FDAF-PEM-AFROW for AEC/AFC

The WISE-GRASS-FDAF-PEM-AFROW for AEC/AFC is given in Algorithm 1 (parts 1.1 and 1.2). The FDAF implementation corresponds to the overlap-save FDAF with gradient constraint and power normalization [34], where $\mathbf{u}(k)$, $\mathbf{v}(k)$ and $\mathbf{n}(k)$ are length-M vectors, with $M = 2N$ and $N = n_F + 1$, satisfying the overlap-save condition in FDAF, e.g., $\mathbf{u}(k) = [u(kN-N+1), ...u(kN+N)]^T$, where $k = 0, 1, ...\frac{L-N}{N}$ is the block-time index, L is the total length of the signals, $\circ$ represents the element-wise multiplication operator, and finally, P is the block length used to estimate $A(q,t)$ with $N \leq P \leq 2N$. The upper asterisk $(\cdot)^*$ denotes complex conjugation and $[\cdot]_{a:b}$ represents a range of samples within a vector. The index m in frequency-domain variables, e.g., $E_a(m,k)$, refers to a signal in the mth frequency bin of a block, $m = 0, ..., M-1$. On the other hand, a capital bold-face variable, e.g., $\mathbf{E}_a(k)$ denotes an M-dimensional vector of frequency components. The subscript a, e.g., $E_a(m,k)$, denotes a signal output from the decorrelating prefilter, as shown in Figure 6.2. Lines $2, 4, 5, 7$ and $9, 11, 12, 14$ correspond to the data generation and are bracketed to enphasyzed that these are not truly part of the algorithm. The specific WISE-GRASS modifications within the FDAF-PEM-AFROW algorithm are shown in Algorithm 1.2. The proposed WISE-GRASS-FDAF-PEM-AFROW weight update equation is given by

$$\boldsymbol{\phi}(k) = \left[\mathcal{F}^{-1}\{\boldsymbol{\mu}(k) \circ \mathbf{W}(k) \circ \boldsymbol{\Theta}(k)\}\right]_{1:N} \tag{6.10}$$

$$\hat{\mathbf{F}}(k) = \hat{\mathbf{F}}(k-1) + \mu_{\max}\mathcal{F}\{[\boldsymbol{\phi}^T(k) \quad \mathbf{0}_N]^T\} \tag{6.11}$$

where $\mathcal{F}\{\cdot\}$ and $\mathcal{F}^{-1}\{\cdot\}$ denote the M-point discrete Fourier transform (DFT) and inverse DFT (IDFT) respectively, $\boldsymbol{\mu}(k)$ corresponds to the power normalization step typically included in an FDAF update, μ_{max} sets the maximum allowed value of the step size, and $\mathbf{W}(k)$ together with $\boldsymbol{\Theta}(k)$ form the WISE-GRASS modifications which will be explained in detail in the following sections.

6.3.1 WIener variable Step sizE

The *WIener variable Step sizE* (WISE) modification into the FDAF-PEM-AFROW algorithm introduces a frequency-domain variable step size. Basically, the goal is to slow down the adaptation in frequency bins where the *Echo-to-Near-End-Signal Ratio* (ENR) is low and increase it in those frequency bins where the ENR is high. If we recall that the near-end signal consists of the near-end speech signal and near-end noise signal, then we can calculate $ENR(t) = \frac{\sigma_x^2(t)}{\sigma_v^2(t) + \sigma_n^2(t)}$, where $\sigma_x^2(t)$, $\sigma_v^2(t)$ and $\sigma_n^2(t)$ are the variance of the echo, the near-end speech signal and the near-end noise signal respectively. In the *near-end-signal-free* case, i.e. $v(t) = n(t) = 0$, the microphone signal consists only

Algorithm 1.2. Second part of the WISE-GRASS-FDAF-PEM-AFROW algorithm for AEC/AFC.
The second part explains the WISE-GRASS.

for $m = 0, ..., M-1$ **do**
  $\hat{P}_{U_a}(m,k) = \lambda_0 \hat{P}_{U_a}(m,k-1) + (1-\lambda_0)|U_a(m,k)|^2$ (Recursive power estimation)
  $\mu(m,k) = \left[\hat{P}_{U_a}(m,k) + \delta\right]^{-1}$ (Power normalization)
  $\theta(m,k) = E_a(m,k){U_a}^*(m,k)$ (Gradient estimation)
  $\nabla(m,k) = \lambda_3 \nabla(m,k-1) + (1-\lambda_3)|\theta(m,k)|^2$
  $\alpha(m,k) = \angle\theta(m,k)$ (Phase estimation)
  $\Theta(m,k) = \sqrt{\nabla(m,k)} e^{[j\alpha(m,k)]}$ (GRASS)
  $\hat{P}_{X_a}(m,k) = \lambda_1 \hat{P}_{X_a}(m,k-1) + (1-\lambda_1)|\hat{X}_a(m,k)|^2$
  $\hat{P}_{D_a}(m,k) = \lambda_2 \hat{P}_{D_a}(m,k-1) + (1-\lambda_2)|E_a(m,k)|^2$
  $W(m,k) = \dfrac{\sqrt{\hat{P}_{X_a}(m,k)}}{\sqrt{\hat{P}_{X_a}(m,k) + \hat{P}_{D_a}(m,k)}}$ (WISE)
end for
$\boldsymbol{\phi}(k) = \left[\mathcal{F}^{-1}\{\boldsymbol{\mu}(k) \circ \mathbf{W}(k) \circ \boldsymbol{\Theta}(k)\}\right]_{1:N}$
$\hat{\mathbf{F}}(k) = \hat{\mathbf{F}}(k-1) + \mu_{\max}\mathcal{F}\{[\boldsymbol{\phi}^T(k) \quad \mathbf{0}_N]^T\}$
end for

of the echo so one could apply the maximum step size in each frequency bin. Once a near-end signal is present in the microphone signal, and especially so if it is colored and non-stationary, the step sizes in the different frequency bins should be reduced accordingly.

The concept for deriving the WISE is that of applying a single-channel frequency-domain noise reduction Wiener filter to the microphone signal $y(t)$. This may also be seen as an *echo-enhancement* filter, however, we do not explicitly use the output of the filter itself, but we use the Wiener filter gain as a variable step size in the adaptive filer. The step size in each frequency bin is then varied by the gain of the Wiener filter at the frequency bin.

We assume that the signals are wide-sense stationary, that we have access to the full record of samples and that disjoint frequency bins can be considered uncorrelated [35]. So, without loss of generality, we may consider a single frequency bin and work with the m-dependency. The frequency-domain microphone signal (6.3) is

$$\begin{aligned} Y(m,k) &= X(m,k) + V(m,k) + N(m,k) \\ &= X(m,k) + D(m,k) \end{aligned}$$

where $X(m,k)$ is the *desired* signal and $D(m,k)$ is the *noise* signal which is to be removed from the microphone signal. An estimate of the desired signal may

be obtained as

$$\hat{X}(m,k) = W_0(m,k)Y(m,k) \tag{6.12}$$

which gives the (theoretical) frequency-domain Wiener filter gain as

$$W_0(m,k) = \frac{P_{XY}(m,k)}{P_Y(m,k)} \tag{6.13}$$

where $P_Y(m,k) = \mathcal{E}\{Y(m,k)Y^*(m,k)\}$ and $P_{XY} = \mathcal{E}\{X(m,k)Y^*(m,k)\}$ are the power spectral density (PSD) of $Y(m,k)$, and the cross-power spectral density (CPSD) of $X(m,k)$ and $Y(m,k)$ respectively. A common assumption in single-channel noise reduction is that the desired signal $X(m,k)$ is uncorrelated with the noise component $D(m,k)$, so that the numerator of the Wiener filter results in $P_{XY}(m,k) = P_X(m,k)$ and the denominator becomes $P_Y(m,k) = P_X(m,k) + P_D(m,k)$ which gives the (theoretical) frequency-domain Wiener filter gain as

$$W_0(m,k) = \frac{P_X(m,k)}{P_X(m,k) + P_D(m,k)}. \tag{6.14}$$

In order to obtain (6.14), we have neglected the terms corresponding to the expected value of the cross-product of $X(m,k)$ and $D(m,k)$ assuming they are uncorrelated. This assumption can truly be exploited only for infinitely long observations of ergodic and stationary processes [1]. In real AEC applications, however, the near-end signal as well as the loudspeaker signal is a speech signal which is highly colored and non-stationary. Although the near-end speech signal and the loudspeaker signal are assumed to be uncorrelated, this however does not imply that the correlation between these two signals is zero within a short-time observation window [9]. In AFC applications, on the other hand, this assumption is clearly violated as explained in Section 6.1.2. In [32], we have shown that the IPC between the near-end speech signal and the loudspeaker signal may be very large. Besides, the practical computation of (6.14) is generally based on time-recursive estimates of the PSD using short-time Fourier transform (STFT) of $X(m,k)$ and $D(m,k)$. All this means that, in real AEC and AFC applications, the practical computation of (6.14) would clearly lead to misleading gain values.

In [32], however, we have shown that the IPC between the far-end speech signal and near-end speech signal can be significantly reduced by using the FDAF-PEM-AFROW framework. Consequently, we will consider deriving the Wiener filter gains (6.14) using the filtered signals (see Figure 6.2) so as to apply them in both real AEC and AFC applications. Then the *desired* signal is the filtered echo signal $x_a(t)$ and the filtered near-end signals, grouped in $d_a(t) = v_a(t) + n_a(t)$, are considered as the *noise* component in the microphone signal $y_a(t)$. The (theoretical) frequency-domain Wiener filter using whitened signals will moreover result in a similar filter as the theoretical one in (6.14),

since

$$P_{X_aY_a}(m,k) = \mathcal{E}\{A(m,k)X(m,k)Y^*(m,k)A^*(m,k)\} = |A(m,k)|^2 P_{XY} \tag{6.15}$$

$$P_{Y_a}(m,k) = \mathcal{E}\{A(m,k)Y(m,k)Y^*(m,k)A^*(m,k)\} = |A(m,k)|^2 P_Y, \tag{6.16}$$

so that

$$W_0(m,k) = \frac{P_{X_aY_a}(m,k)}{P_{Y_a}(m,k)} = \frac{P_{X_a}(m,k)}{P_{X_a}(m,k) + P_{D_a}(m,k)}. \tag{6.17}$$

We will compute time-recursive estimates of the PSDs instead and use available estimates of the desired signal, i.e., $\hat{X}_a(m,k)$, and of the noise component in the microphone signal, i.e., $E_a(m,k)$. The Wiener filter gains are thus efficiently estimated as follows

$$m = 0, ..., M-1$$

$$\hat{P}_{X_a}(m,k) = \lambda_1 \hat{P}_{X_a}(m,k-1) + (1-\lambda_1)|\hat{X}_a(m,k)|^2 \tag{6.18}$$

$$\hat{P}_{D_a}(m,k) = \lambda_2 \hat{P}_{D_a}(m,k-1) + (1-\lambda_2)|E_a(m,k)|^2 \tag{6.19}$$

Finally, using (6.18)-(6.19), we can write

$$W(m,k) = \frac{\hat{P}_{X_a}(m,k)}{\hat{P}_{X_a}(m,k) + \hat{P}_{D_a}(m,k)} \tag{6.20}$$

where $\hat{P}_{X_a}(m,k)$ is an estimate of the PSD of $X_a(m,k)$ which is calculated in (6.18) using the output of the adaptive filter $\hat{X}_a(m,k) = U_a(m,k)\hat{F}(m,k-1)$, and where $\hat{P}_{D_a}(m,k)$ is an estimate of the PSD of $D_a(m,k)$ which is calculated in (6.19) using the prediction-error signal $E_a(m,k)$.

An interpretation of the Wiener filter gain $W(m,k)$ in terms of the $E\hat{N}R(m,k)$ can be given by writing (6.20) as

$$W(m,k) = \frac{E\hat{N}R(m,k)}{E\hat{N}R(m,k)+1} \tag{6.21}$$

where

$$E\hat{N}R(m,k) = \frac{\hat{P}_{X_a}(m,k)}{\hat{P}_{D_a}(m,k)} \tag{6.22}$$

$$= \frac{\hat{P}_{X_a}(m,k)}{\hat{P}_{V_a}(m,k) + \hat{P}_{N_a}(m,k)}. \tag{6.23}$$

The Wiener filter gain is a real positive number in the range $0 \leq W(m,k) \leq 1$ which is used as a variable step size in (6.10). A maximum value for the step size is also adopted, so that the effective step size is $\mu_{\max} W(m,k)$. Let

us now consider the two limiting cases of (1) an 'echo-only' microphone signal, i.e., $E\hat{N}R(m,k) = \infty$ and (2) a 'near-end-only' microphone signal, i.e., $E\hat{N}R(m,k) = 0$. In the first case the Wiener filter gain $W(m,k) = 1$, so the filter would apply the maximum step size in the mth frequency bin. In the second case, the Wiener filter gain $W(m,k) = 0$ so the adaptation is suspended in the mth frequency bin. In between these two extreme cases, the Wiener filter gain reduces the step size in proportion to an estimate of the ENR in each frequency bin.

6.3.2 GRAdient Spectral variance Smoothing

The *GRAdient Spectral variance Smoothing* (GRASS) is included to reduce the variance in the gradient estimation, in particular, the effect on the gradient estimation of sudden high-amplitude near-end signal samples. Although the step size control has been considered in Section 6.3.1, it may not be sufficiently fast to react in certain occasions, e.g., in DT bursts. In FDAF(-PEM-AFROW), an estimate of the gradient $\theta(m,k) = E_a(m,k){U_a}^*(m,k)$ is calculated for every block of samples. This noisy estimate may be separated into two components as

$$\theta(m,k) = \theta_0(m,k) + \theta_{D_a}(m,k) \tag{6.24}$$

where $\theta_0(m,k) = \left[X(m,k) - \hat{X}(m,k)\right] {U_a}^*(m,k)$ is the *true gradient* and $\theta_{D_a}(m,k) = D_a(m,k){U_a}^*(m,k)$ is the *gradient noise*.

The concept for deriving the GRASS is that of applying averaged periodograms, which are typically used for estimating the PSD of a signal [36]. The periodogram is defined here as the squared magnitude of the gradient. An estimate of the true gradient PSD, at a given frequency m, is obtained as

$$\hat{P}_{\hat{\theta}_0}(m,k) = \frac{1}{K}\sum_{j=0}^{K-1} |\theta(m,k-j)|^2. \tag{6.25}$$

It is shown in [36] that the variance of (6.25) is

$$\mathrm{var}\left\{\hat{P}_{\hat{\theta}_0}(m,k)\right\} \approx \frac{1}{K}\left|\theta_0(m,k)\right|^2 \tag{6.26}$$

so that it is proportional to the square of the true gradient's PSD divided by K and hence, as $K \to \infty$, (6.26) approaches zero. This concept would be sufficiently justified if the RIR does not change and the algorithm has converged to an optimum, therefore $\theta_0(m,k) = 0$ by definition. In practice, of course, the true gradient $\theta_0(m,k)$ is time-varying. Therefore the concept of averaging starts to lose its sense. However we will assume that the true gradient varies slowly, therefore, a low-pass filter (LPF) with a low cut-off frequency would be more beneficial than (6.25).

The realization of the GRASS is based on a time-recursive averaging of the gradient estimate (6.27)-(6.28). This is another way to obtain a first order infinite impulse response (IIR) filtering where λ_3 is the pole of the low-pass IIR filter, and it is given as

$$\theta(m,k) = E_a(m,k)U_a{}^*(m,k) \tag{6.27}$$

$$\nabla(m,k) = \lambda_3\nabla(m,k-1) + (1-\lambda_3)|\theta(m,k)|^2 \tag{6.28}$$

In fact, a LPF with a low cut-off frequency, will effectively filter out the gradient noise and reduce the variance in the gradient estimation. Note that (6.25), is a special case of a low-pass FIR filter, i.e., $\hat{P}_{\hat{\theta}_0}(m,k) = \sum_{j=0}^{K-1} b_k|\theta(m,k-j)|^2$ with weights $b_j = 1/K, \forall j$.

Finally, the phase $\alpha(m,k)$, that is obtained from the gradient estimate in (6.29), is applied in (6.30) to form the GRASS, i.e., $\Theta(m,k)$, as

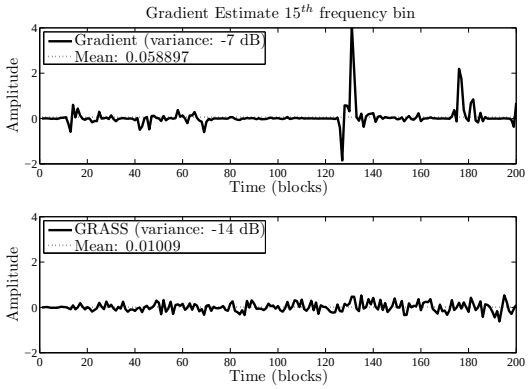
$$\alpha(m,k) = \angle\theta(m,k) \tag{6.29}$$

$$\Theta(m,k) = \sqrt{\nabla(m,k)}e^{[j\alpha(m,k)]} \tag{6.30}$$

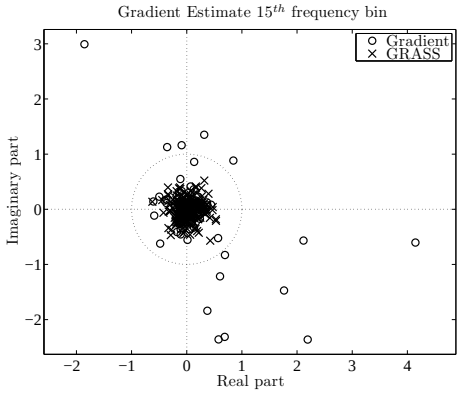
A practical simulation of the GRASS for $m = 15$, i.e., the 15-th frequency bin, with $k = 1, ..., 200$ and $\lambda_3 = 0.95$ is shown in Figure 6.3. For this simulation we have used two speech signals, $u(t)$ and $v(t)$, and a noise signal $n(t)$ sampled at 8 kHz. The AR coefficients of the near-end signal model are estimated using $d(t) = v(t) + n(t)$ and $n_A = 55$. The signals, $u(t)$ and $d(t)$ are then filtered, as in Algorithm 1.1 lines $16 - 20$, before feeding them to the overlap-save-type recursion. The gradient-constraint-type of calculations are performed, as in Algorithm 1.2 lines $34 - 35$, to obtain both the noisy gradient and the GRASS gradient. We have assumed that the true gradient has converged to an optimum, and thus it is, by definition, equal to zero. In Figure 6.3(a) a time-domain representation is shown where the upper and lower figures correspond to the real part of $\theta(m,k)$ and $\Theta(m,k)$, respectively. Besides, in Figure 6.3(b) a complex representation is shown where the circles ($\circ$) represent the complex value of $\theta(m,k)$ and the crosses ($\times$) represent the complex value of $\Theta(m,k)$. In both representations, the variance in the estimation is shown to be reduced by 7 dB, the mean value is closer to zero, and high-amplitude samples are clearly smoothed.

6.4 Evaluation

Simulations are performed using speech signals sampled at 8 kHz. Two types of near-end noise $n(t)$ are used in the simulations namely, a white Gaussian noise



(a)



(b)

Figure 6.3: Instantaneous gradient and GRASS estimates with $m = 15$ and $k = 1, ..., 200$. (a) Time representation: Upper and lower figures correspond to the real part of $\theta(m, k)$ and $\Theta(m, k)$, respectively. The solid line $(-)$ represents the gradient estimate and the dotted line $(\cdots)$ represents its mean value. (b) Complex representation: the circles $(\circ)$ represent the complex value of $\theta(m, k)$ and the crosses $(\times)$ represent the complex value of $\Theta(m, k)$.

(WGN) and a *speech babble* noise which is one of the most challenging noise types [37] in signal enhancement applications. We define two measures which determine the relative signal levels in the simulations: the *Signal-to-Echo Ratio* (SER)$= 10 \log_{10} \frac{\sigma_x^2}{\sigma_v^2}$ and the *Signal-to-Noise Ratio* (SNR)$= 10 \log_{10} \frac{\sigma_x^2}{\sigma_n^2}$, where σ_x^2, σ_v^2 and σ_n^2 are the variances of the echo signal, the near-end speech signal and near-end noise signal.

In the AEC simulations, the far-end (or loudspeaker) signal $u(t)$ is a female speech signal and the near-end speech is a male speech signal. The microphone signal consists of three concatenated segments of speech: the first and third 12 s segment correspond to a single-talk situation, i.e., $y(t) = x(t)+n(t)$, while the second 13 s segment corresponds to a DT situation, i.e, $y(t) = x(t)+n(t)+v(t)$. The AR model order in the AEC is $n_A = 12$, following the indications given in [32], and the APA order is $Q = 4$. There are two RIRs used in the AEC simulations: an 80-taps impulse response $\mathbf{f}_1$ measured from a real mobile phone, and an artificial RIR $\mathbf{f}_2 = \mathbf{f}_1 + \bar{\mathbf{n}} \circ \mathbf{f}_1$ simulating an abrupt RIR change where $\bar{\mathbf{n}}$ is one realization of a Gaussian random process. In the AEC simulations, the WGN and speech babble noise types are set at different SNRs: 30 and 20 dB respectively. Several SER values are used for the simulations: from mild (15 dB) to highly adverse (-5 dB) DT conditions.

In the AFC simulations, the near-end speech is the same female speech signal as in the AEC simulations. Two different AR model orders are chosen as in [4], [31]: $n_A = 12$ which is common in speech coding for formant prediction and $n_A = 55$ which is high enough to capture all near-end signal dynamics. The forward path gain $K(t)$ is set 3 dB below the *maxinum stable gain* (MSG) without feedback cancellation i.e., see (6.32) in Section 6.4.2. A measured 100-tap acoustic impulse response was obtained from a real hearing aid and used to simulate the feedback path. In both AEC and AFC, the window length P was chosen to be 20 ms (160 samples), which corresponds to the average time interval in which speech is considered stationary.

6.4.1 Competing algorithms and tuning parameters

AEC simulations are performed comparing four algorithms namely: the Projection Correlation VSS (PC-VSS) algorithm [17], the Practical VSS affine projection algorithm (PVSS) [22], the Transform-Domain PEM-AFROW (TD-PEM-AFROW) algorithm [31] and the proposed WISE-GRASS-FDAF-PEM-AFROW algorithm. PC-VSS belongs to the gradient-based VSS algorithms and it is claimed to feature the appealing characteristics of being able to distinguish between RIR changes and DT. PC-VSS is the result of improving the algorithm given in [12] to be specifically suited for AEC in DT situations. In PC-VSS, the adaptation rate is controlled by a measure of the correlation between instantaneous and long-term averages of the so-called projection vectors, i.e., gradient vectors in APA. It appears that PC-VSS outperforms the algorithms given in [12] and [38] in DT situations. Moreover, it does not rely on any signal or system model so it is easy to control in practice. The second competing algorithm is the practical VSS affine projection algorithm (PVSS) [22] which stems from the so-called NPVSS proposed in [19]. PVSS takes into account the near-end signal power variations, it is effectively used in DT situations and, it is claimed to be easy to control in practice. PVSS has been shown to outperform the algorithms proposed in [21], [39], and [40].

Algorithm	Computation	Total
PC-VSS	$2NQ + 7Q^2 + 4N + 12$	1084
PVSS	$2NQ + 7Q^2 + 3Q + 6 + Q + 4Q + 2$	792
PEM-AFROW	$\left(8 + \frac{4P + 2n_A + 1}{P}\right) N + \frac{1}{P} n_A^2 +$ $\left(4 + \frac{4P + 2}{P}\right) n_A + \frac{P - 1}{P} + 10$	980
TD-PEM-AFROW	Same as PEM-AFROW = 980 $+6N \log_2 N$	4015
WISE-GRASS FDAF-PEM-AFROW	$1/N(18M \log_2 M + 25M + 3N)$ $+1/N(n_A^2 + 4Mn_A + 4 + M)$	416

Table 6.1: Complexity comparison by the number of FLOPS per recursion. One FFT/IFFT is $3M \log_2 M$ and the phase calculation is an M complexity operation.

The third competing algorithms, TD-PEM-AFROW, has been investigated in terms of DT robustness in AEC and general improvement in AFC. It has been shown that the combination of a prewhitening of the input and microphone signal together with transform-domain filter adaptation, successfully leads to an algorithm that solves the problem of decorrelation in a very efficient manner. TD-PEM-AFROW is very robust in DT situations and boosts the performance of the simplest AFC (i.e., using only an AR model for the near-end signal).

All the algorithms presented in this chapter can be downloaded [2] along with the scripts generating every figure in this section. The tuning parameters in both PVSS and PC-VSS are chosen according to the specifications given in [22] and [17] respectively. The parameters of TD-PEM-AFROW are chosen to have similar initial convergence curves as PVSS and PC-VSS. In the proposed WISE-GRASS-FDAF-PEM-AFROW algorithm, the following parameters are chosen to have similar initial convergence properties as the other algorithms: $\mu_{\max} = 0.03$, $\delta = 2.5e^{-6}$, $\lambda_0 = 0.99$, $\lambda_1 = 0.1$, $\lambda_2 = 0.9$. The different values of λ_1 and λ_2 in the time-recursive averaging for power estimation, basically aim for having a longer averaging window for the near-end signal and a shorter averaging window for the echo signal. Note that for higher robustness to near-end signals a higher value of λ_2 may be chosen; however, convergence would be slower.

AFC simulations are, on the other hand, performed comparing the original PEM-AFROW [26], TD-PEM-AFROW [31] and the proposed WISE-GRASS-FDAF-PEM-AFROW. The parameters are tuned to have similar initial performance curves. In all three algorithms the following common values are applied: the forward path $G(q, t)$ consist of a delay of 80 samples and a fixed

[2] *http : //homes.esat.kuleuven.be/ ∼ dspuser/abstract*12_214_2.*html*

gain $K(t) = K$, $\forall t$, resulting in a 3 dB MSG without AFC, and a window of $P = 160$ samples is used for estimating the AR model. For WISE-GRASS-FDAF-PEM-AFROW, $\lambda_0 = \lambda_2 = 0.99$, $\lambda_1 = 0.1$, and $\mu_{\max} = 0.025$.

The computational complexity of the different algorithms used in the AEC simulations is given in Table 6.1 using $N = 80$ and APA order $Q = 4$. The computational complexity of WISE-GRASS-FDAF-PEM-AFROW is the lowest and that of TD-PEM-AFROWis the highest. The drawback of most of the FDAF-based algorithms is, however, their inherent delay of N samples. In AFC applications, this delay, or part of it, may be 'absorbed' by the forward path.

6.4.2 Performance measures

The performance measure for AEC simulations is the *misadjustment* (MSD). The MSD between the estimated RIR $\hat{\mathbf{f}}(t)$ and the true RIR $\mathbf{f}_1$ or $\mathbf{f}_2$ represents the accuracy of the estimation and is defined as,

$$\mathrm{MSD}(t) = 10\log_{10}\frac{\left\|\hat{\mathbf{f}}(t) - \mathbf{f}_{1,2}\right\|_2^2}{\|\mathbf{f}_{1,2}\|_2^2} \tag{6.31}$$

where $\mathbf{f}_{1,2}$ is either $\mathbf{f}_1$ or $\mathbf{f}_2$. The performance measure for AFC is the *maximum stable gain* (MSG). The achievable amplification before instability occurs is measured by the MSG, which is derived from the Nyquist stability criterion [27] and it is defined as

$$\mathrm{MSG}(t) = -20\log_{10}[\max_{\omega\in\phi}|J(\omega,t)[F(\omega) - \hat{F}(\omega,t)]|] \tag{6.32}$$

where ϕ denotes the set of frequencies at which the loop phase is a multiple of 2π (i.e., the feedback signal $x(t)$ is in phase with the near-end speech signal $v(t)$) and $J(\omega,t)$ denotes the forward path processing before the amplifier, i.e., $G(\omega,t) = J(\omega,t)K(t)$.

6.4.3 Simulation results for DT-robust AEC

In the first set of simulations, the noise $n(t)$ is a WGN at 30 dB SNR. PVSS and PC-VSS are first tuned, as [22] and [17] respectively suggested, to obtain the best performance both in terms of convergence rate and final MSD. TD-PEM-AFROW and WISE-GRASS-FDAF-PEM-AFROW are tuned to have similar initial convergence rate as PVSS and PC-VSS. This set of parameters remains unchanged throughout the simulations. All figures in this section consist of an upper part showing the microphone signals and a bottom part showing the AEC performance. The upper part shows the echo (dark blue) and near-end (light green) signals, and the bottom part shows the MSD on the same time

scale. When a change of RIR occurs, the first part of the echo signal (generated by $\mathbf{f}_1$) will be in a lighter color than the second part (generated by $\mathbf{f}_2$). Plotting the time domain representation of these signals allows us to distinguish between the amplitude and start/end points of both the echo and near-end signals.

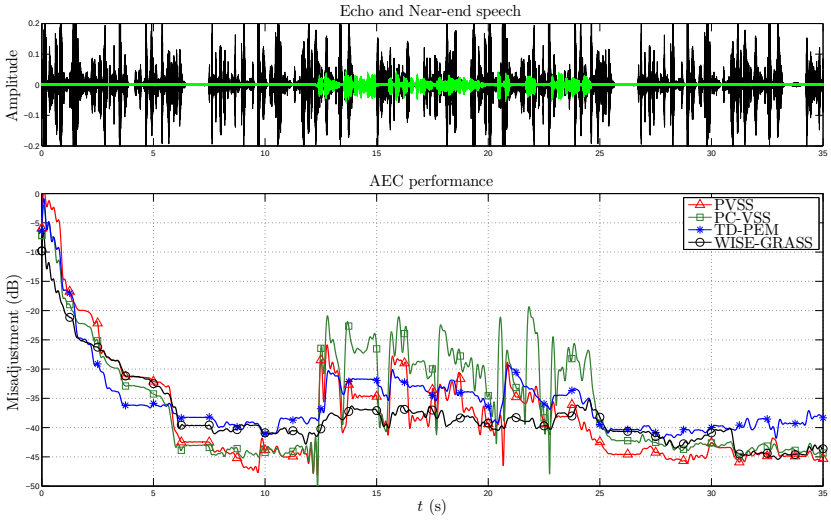
Results for bursting DT

Figures 6.4 and 6.5 show the AEC performance when a bursting DT (from 12.5 s during 12.5 s) occurs at two different SERs (15 dB and 5 dB) immersed in WGN at 30 dB SNR (Figure 6.4) and immersed in speech babble noise at 20 dB SNR (Figure 6.5). More specifically, in Figure 6.4(a) and 6.4(b) a DT burst is shown at 15 and 5 dB SER respectively both immersed in WGN at 30 dB SNR. It can be seen that in WGN, PC-VSS and PVSS achieve 5 dB improvement in final MSD compared to the other two algorithms during single talk, i.e., -45 to -40 dB respectively. However, during DT, WISE-GRASS-FDAF-PEM-AFROW outperforms the other three algorithms: in Figure 6.4(a) by 5 dB in the case of PVSS and TD-PEM-AFROW, and by around 10 dB in the PC-VSS case; in Figure 6.4(b) these differences are increased since WISE-GRASS-FDAF-PEM-AFROW outperforms TD-PEM-AFROW by $8-10$ dB, PVSS by $5-8$ dB and PC-VSS by $10-20$ dB.

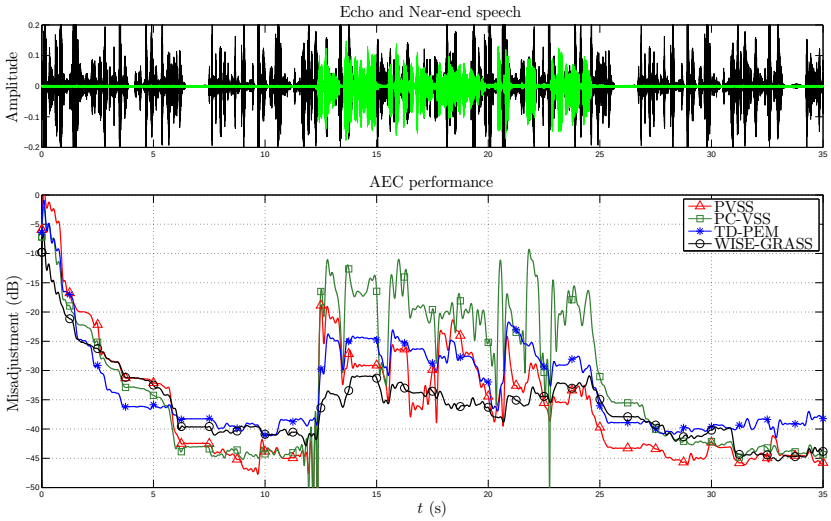
Figure 6.5(a) and 6.5(b) show two scenarios where the near-end noise is speech babble noise at 20 dB SNR and two DT bursts at 15 and 5 dB SER occur after 12.5 s In these adverse scenarios, the improved performance of WISE-GRASS-FDAF-PEM-AFROW compared to the other three algorithms is demonstrated. The convergence of PVSS is seriously damaged in this scenario. As for the PC-VSS convergence, although it is the same as with WGN during the first second, the MSD during single-talk is much higher than with WGN. On the other hand, TD-PEM-AFROW and WISE-GRASS-FDAF-PEM-AFROW obtain a smoother and faster convergence. During DT, the MSD of WISE-GRASS-FDAF-PEM-AFROW remains at -40 dB, insensitive to DT. The other three algorithms perform as in the WGN case during DT, which evidences the great difference with respect to the WISE-GRASS-FDAF-PEM-AFROW performance. These differences are clearly visible in the case of DT at 5 dB SER. After DT, WISE-GRASS-FDAF-PEM-AFROW restores the low MSD value (-40 dB) although it is outperformed by TD-PEM-AFROW followed by PVSS. PC-VSS seems to have serious problems in speech babble noise since its MSD is almost 10 dB larger than for the other algorithms.

Results of RIR change during bursting DT

Figure 6.6 shows a scenario with an abrupt change of the RIR when the near-end noise is speech babble noise at 20 dB SNR. More specifically, in Figure **??** the AEC performance is shown where the change of RIR occurs during DT at 15 dB SER and in Figure 6.6(b) the SER is 5 dB. The poorest performance of

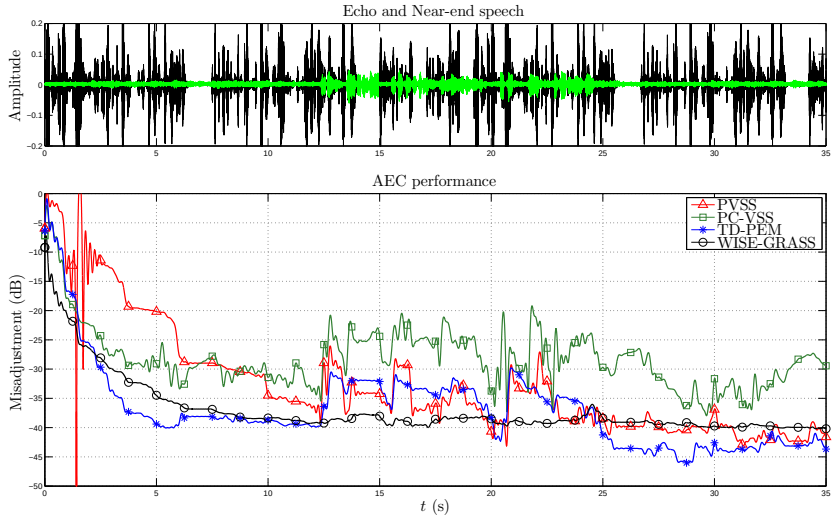


(a)

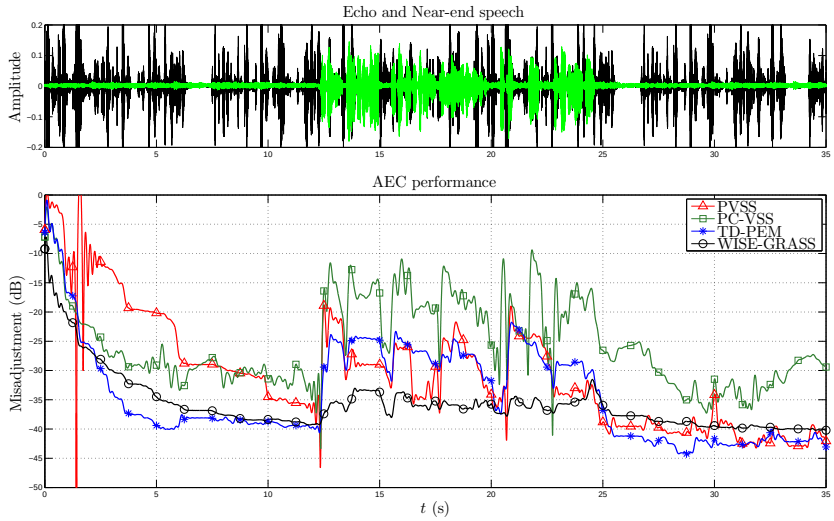


(b)

Figure 6.4: AEC performance in WGN at 30 dB SNR. Bursting DT occurs at different SER: (a) DT at 15 dB SER. (b) DT at 5 dB SER.



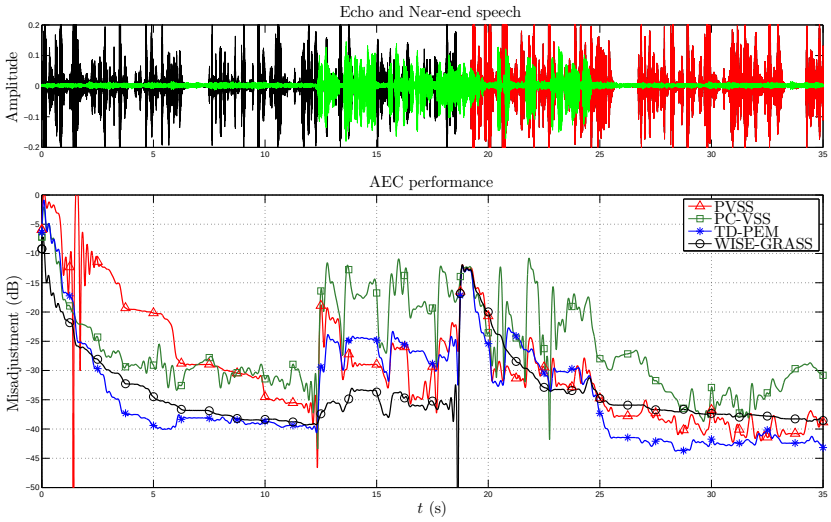
(a)



(b)

Figure 6.5: AEC performance in speech babble noise at 20 dB SNR. Bursting DT occurs at different SER: (a) DT at 15 dB SER. (b) DT at 5 dB SER.

(a)



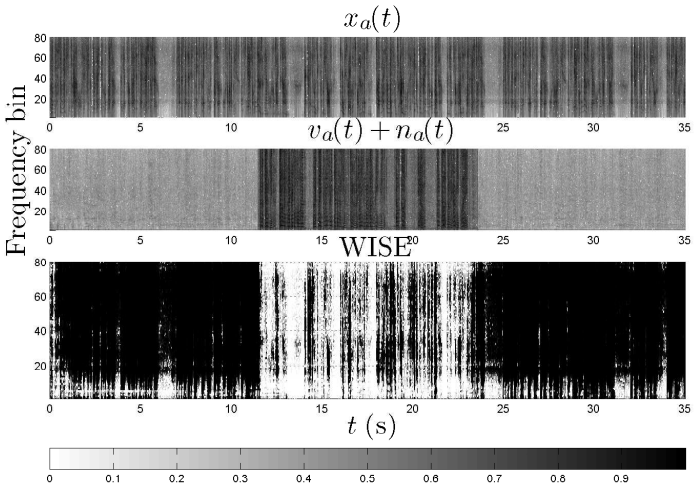
(b)

Figure 6.6: AEC performance in terms of MSD(t) in speech babble noise at 20 dB SNR and an abrupt change of RIR after 17 s occurs during bursting DT (a) DT at 15 dB SER. (b) DT at 5 dB SER.

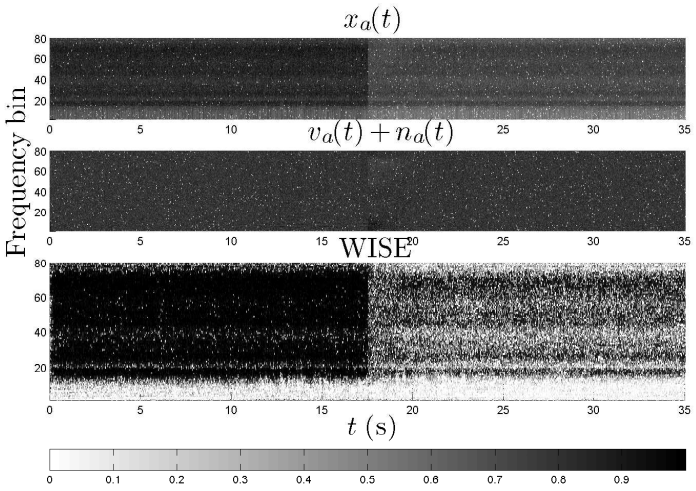
PC-VSS becomes apparent now since the DT contaminates its convergence and hence the MSD is still 15 dB. On the other hand, PVSS and TD-PEM-AFROW, although quite affected by DT, still show a descending trend in their MSD curve which implies that they are converging during DT. In both Figure **??** and 6.6(b), it is observed that the performance of WISE-GRASS-FDAF-PEM-AFROW is surprisingly almost constant. This fact highlights the robustness of WISE-GRASS-FDAF-PEM-AFROW in an adverse scenario. In fact, its performance is barely degraded which demonstrates the robustness of WISE-GRASS-FDAF-PEM-AFROW as compared to the other three algorithms.

WISE evolution

Figure 6.7 shows the WISE evolution, i.e., $W(m,k)$ for $m = 0, ..., 79$ and $k = 1, ..., \frac{L-N}{N}$, in two different scenarios: (1) in Figure 6.7(a) speech babble noise at 40 dB SNR and bursting DT at 5 dB SER is used, and (2) in Figure 6.7(b) both the loudspeaker signal and the near-end noise are WGN at 20 dB SNR, and an abrupt change of the RIR ($\mathbf{f}_2 = 0.5\mathbf{f}_1$) occurs at the 17.5 s mark. Figure 6.7(a) displays the (dense) echo signal spectrogram and the near-end signal



(a)



(b)

Figure 6.7: WISE evolution in different scenarios together with the spectrogram of the echo signal and the near-end signals: (a) speech babble noise at 40 dB SNR and bursting DT at 5 dB SER. (b) Both the the near-end noise and the loudspeaker signal are WGN at 20 dB SNR and abrupt change of RIR ($\mathbf{f}_2 = 1/2\mathbf{f}_1$) occurs at the 17.5 s mark.

spectrogram which clearly shows the near-end activity between 12.5 and 25 s.

In the bottom figure, the WISE evolution shows a drastic step size reduction during DT. At $t = 20$ s where the PSD of the near-end signal is low in every frequency bin, the step size is increased in those frequency bins where the SER and SNR is appropriate: step sizes in frequency bins $0 - 10$ are not increased because the echo PSD is also low in the frequency bins and, thus, the Wiener filter gain should be low. The step sizes in frequency bins from 75 and on, are also smaller due to the fact that the echo PSD is lower. In Figure 6.7(b) it is shown how the step size follows the echo spectrum 'weighted' by the near-end signal PSD.

Results in highly adverse scenarios

The performance of the algorithms is challenged in Figure 6.8 where two highly adverse scenarios are proposed: In this DT situation WISE-GRASS-FDAF-PEM-AFROW significantly outperforms PC-VSS which appears to be strongly affected by DT, and moreover shows a significant improvement with respect to PVSS which itself outperforms TD-PEM-AFROW. In Figure 6.8(a) a change of the RIR occurs which decreases the PVSS performance, while WISE-GRASS-FDAF-PEM-AFROW performance is only decreased with respect to Figure 6.6(b). WISE-GRASS-FDAF-PEM-AFROW shows a robust convergence after the RIR change even in a highly adverse DT level obtaining more than 8 dB improvement compared to PVSS after the RIR change. On the other hand Figure 6.8(b) shows another highly adverse situation where a continuous DT occurs. The simulation is done using four equal-length segments of speech babble noise at four different SERs $(20, 10, 5, 30$ dB). It shows that, in these noise levels, the convergence of PVSS decreases considerably and PC-VSS diverges as the SER increases. After convergence, WISE-GRASS-FDAF-PEM-AFROW shows the lowest MSD, followed by that of TD-PEM-AFROW ($3 - 5$ dB above) and then PVSS ($6 - 7$ dB above). Finally, Figure 6.8(b) shows that, after a change of the RIR in continuous DT, the only two algorithms that are able to converge are TD-PEM-AFROW and WISE-GRASS-FDAF-PEM-AFROW.

Results of WISE-only, GRASS-only and WISE-GRASS in FDAF-PEM-AFROW

Figure 6.9 can shed some light on why WISE-GRASS-FDAF-PEM-AFROW is so robust and yet performs so well. A simulation of FDAF-PEM-AFROW using WISE-only, GRASS-only and the proposed combination WISE-GRASS is shown in two adverse scenarios: speech babble noise at 10 dB and 20 dB SNR both with DT at -5 dB SER and with an abrupt change of the RIR in Figure 6.9(a) and 6.9(b) respectively. In FDAF and stochastic gradient algorithms in general, the excess MSE depends on the step size and noise variance [1], [41]. GRASS-only FDAF-PEM-AFROW, although robust and smooth, has a fixed step size and thus, the final MSD is much higher. However, it is shown how

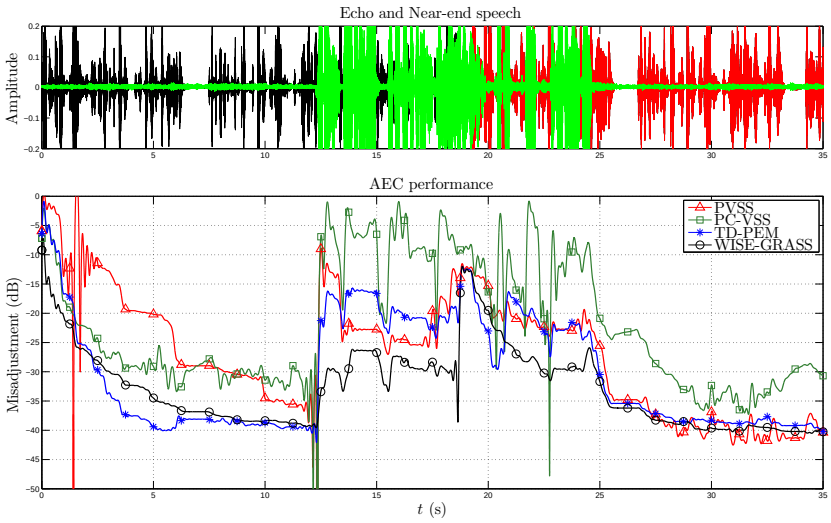
WISE-GRASS-FDAF-PEM-AFROW always obtains the best result compared to WISE-only and GRASS-only. It is interesting to note how WISE-GRASS-FDAF-PEM-AFROW takes over GRASS-only FDAF-PEM-AFROW to WISE-only FDAF-PEM-AFROW at around 3 s in Figure 6.9(a) and around 5 s in Figure 6.9(b).

6.4.4 Simulation results for AFC

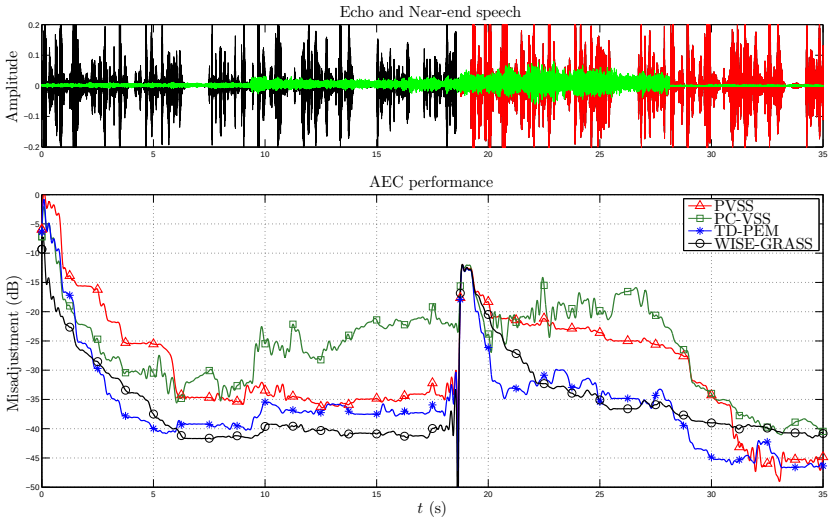
Three AFC scenarios are shown to compare the performance of PEM-AFROW, TD-PEM-AFROW and WISE-GRASS-FDAF-PEM-AFROW. The performance is given in terms of the MSG. The value of $W(m,k) = 1$ for $k = 1, ..., 20$ because at start-up the estimated $\hat{P}_{X_a}(m,k)$, and therefore $W(m,k)$, are very low. WISE-GRASS-FDAF-PEM-AFROW needs some initial iterations until a significant feedback signal PSD is estimated. In Figure 6.10 the circles represent points of PEM-AFROW, stars represent points of TD-PEM-AFROW and squares represent points of WISE-GRASS-FDAF-PEM-AFROW. More specifically, in Figure 6.10(a) the MSG is shown when using a near-end signal model of $n_A = 55$ and WGN at 40 dB SNR. It can be seen that the PEM-AFROW achieves $5 - 7.5$ dB MSG improvement and TD-PEM-AFROW achieves $3 - 5$ dB MSG improvement w.r.t to PEM-AFROW. On the other hand, WISE-GRASS-FDAF-PEM-AFROW outperforms the other two algorithms by $8 - 10$ and $5 - 3$ dB respectively. The superior performance of WISE-GRASS-FDAF-PEM-AFROW appears more clearly in the case of a near-end signal model of $n_A = 12$ and WGN at 40 dB SNR shown in Figure 6.10(b). It can be seen that PEM-AFROW goes into the instability region and that TD-PEM-AFROW outperforms PEM-AFROW by $6 - 7$ dB when using low n_A orders. WISE-GRASS-FDAF-PEM-AFROW outperforms by far the other two algorithms, e.g., $10 - 15$-dB improvement compared to PEM-AFROW. Moreover, it obtains almost the same performance as in the $n_A = 55$ case. In Figure 6.11, a $n_A = 55$ AR model order is used but, in this case, a speech babble noise at 20 dB SNR is chosen. In this case both PEM-AFROW and TD-PEM-AFROW remain at around 5 dB above the instability region. On the other hand, WISE-GRASS-FDAF-PEM-AFROW maintains its superior performance of $6 - 9$ dB with respect to the other two algorithms. In this last scenario, Figure 6.12 shows the WISE evolution in terms of its instantaneous values (in each frequency bin) as time evolves. It is clearly seen that some frequency bins usually have a smaller step size (e.g., frequency bin $5 - 35$ and $85 - 100$) than others (e.g., $35 - 60$ and from $70 - 85$). It is also interesting to notice that, during the first 7 s, frequency bins $5 - 30$ have smaller step sizes than after 7 s.

6.5 Conclusion

In this chapter, we have derived a practical yet highly robust algorithm based on the frequency-domain adaptive filter prediction error method using row operations (FDAF-PEM-AFROW) for DT-robust AEC and AFC. The proposed algorithm contains two modifications, namely (a) the WIener variable Step sizE (WISE) and (b) the GRAdient Spectral variance Smoothing (GRASS) to be performed in FDAF-PEM-AFROW, leading to the WISE-GRASS-FDAF-PEM-AFROW algorithm. The WISE is implemented as a single-channel noise reduction Wiener filter applied to the (prefiltered) microphone signal, where the Wiener filter gain is used as a VSS in the adaptive filter. On the other hand, the GRASS aims at reducing the variance in the noisy gradient estimates based on time-recursive averaging of gradient estimates. WISE-GRASS-FDAF-PEM-AFROW obtains improved robustness and smooth adaptation in highly adverse scenarios such as in bursting DT at high levels, and with a change of the acoustic path during continuous DT. Simulations show that WISE-GRASS-FDAF-PEM-AFROW outperforms other competing algorithms in adverse scenarios both in AEC and AFC applications when the near-end signals (i.e., speech and/or background noise) strongly affect the microphone signal. WISE-GRASS-FDAF-PEM-AFROW combines the best characteristics of robust adaptation both in continuous DT and bursting DT without the need for a DTD. WISE-GRASS-FDAF-PEM-AFROW consequently gathers all the characteristic we are seeking for namely, decorrelation properties (PEM, FDAF), minimum variance (GRASS, FDAF, PEM), variable step size (WISE), and computational efficiency (FDAF).

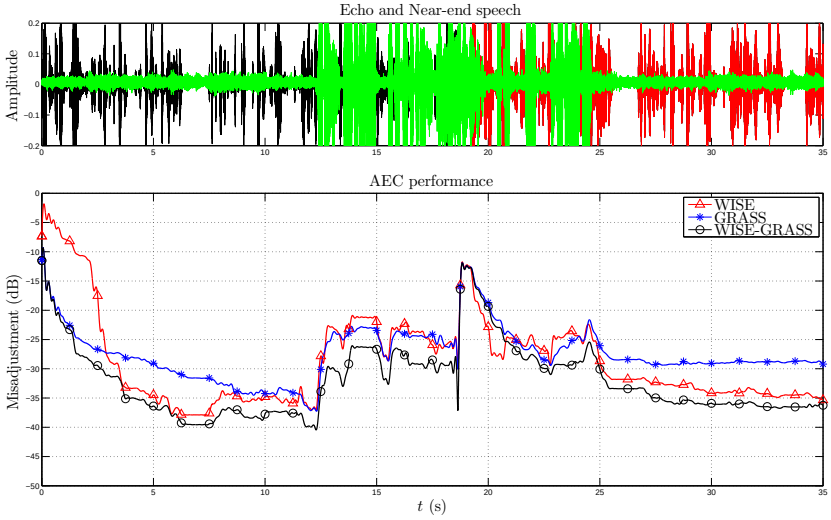


(a)

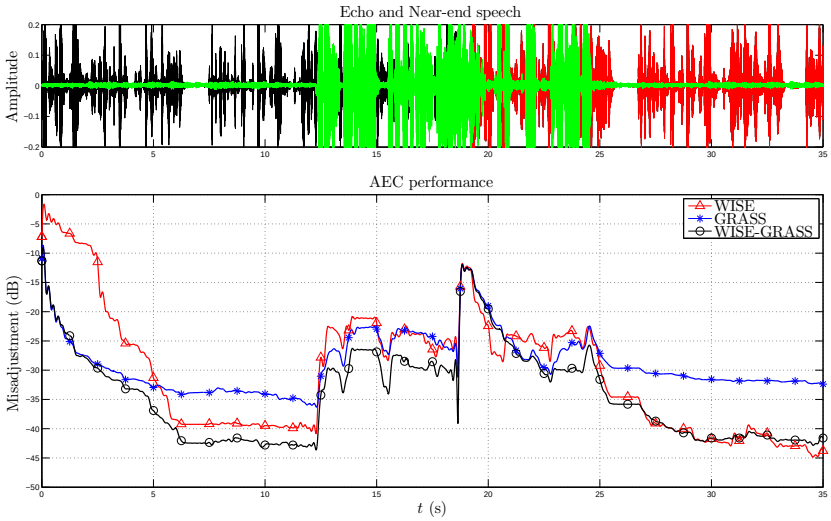


(b)

Figure 6.8: AEC performance comparison in highly adverse scenarios: (a) speech babble noise at 20 dB SNR and a change of the RIR after 17 s during bursting DT at -5 dB SER. (b) speech babble noise (continuous DT) at $20, 10, 5, 30$ dB SNR and change of RIR after 17 s.

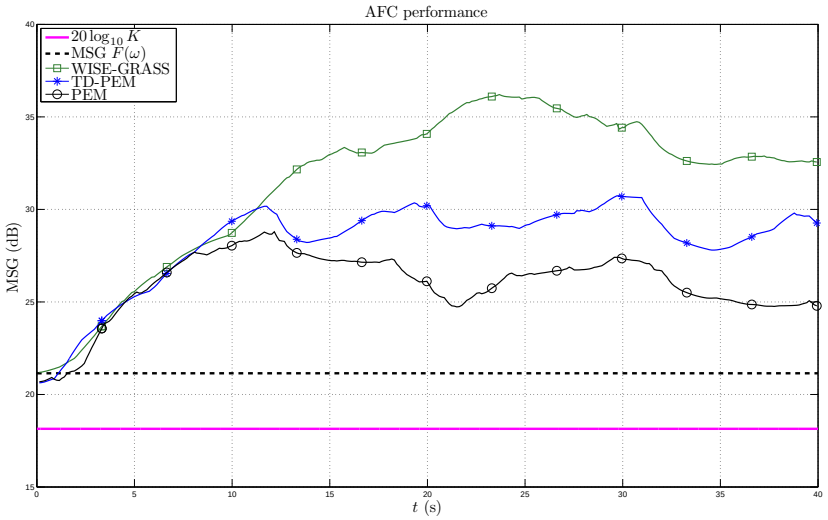


(a)

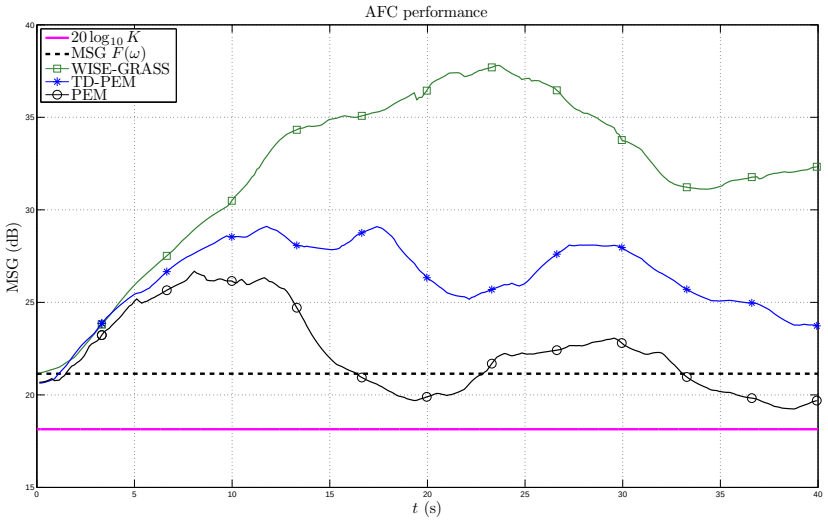


(b)

Figure 6.9: Comparison of WISE-only, GRASS-only and the combination of the two WISE-GRASS. Bursting DT at -5 dB SER and RIR change with continuous speech babble noise at various SNRs: (a) speech babble noise at 10 dB SNR. (b) speech babble noise at 20 dB SNR.



(a)



(b)

Figure 6.10: AFC performance in terms of MSG(t) of the three algorithms: PEM-AFROW, TD-PEM-AFROW and WISE-GRASS-FDAF-PEM-AFROW: (a) $n_A = 55$ and WGN at 40 dB SNR. (b) $n_A = 12$ and WGN at 40 dB SNR.

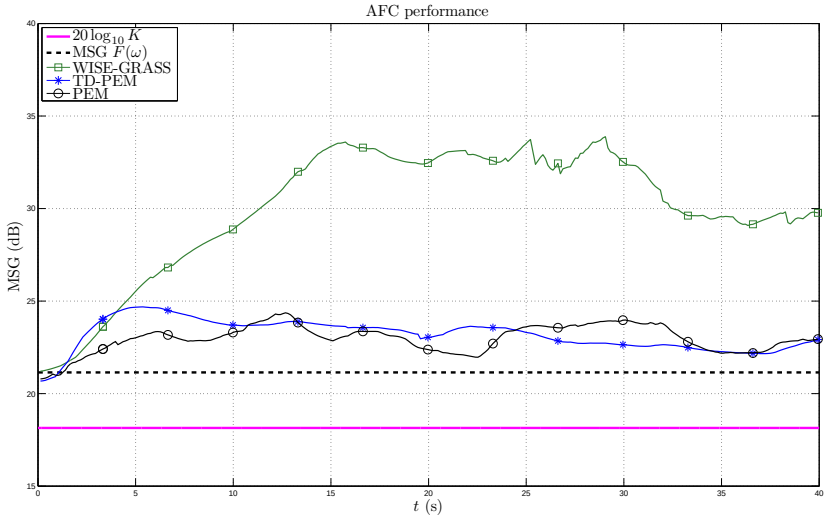


Figure 6.11: AFC performance in terms of MSG(t) of the three algorithms: PEM-AFROW, TD-PEM-AFROW and WISE-GRASS-FDAF-PEM-AFROW: $n_A = 55$ and speech babble noise at 20 dB SNR.

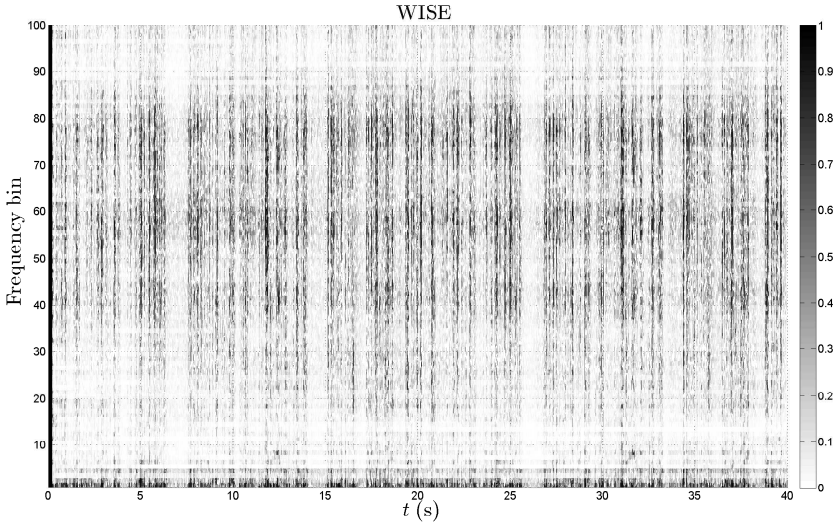


Figure 6.12: AFC WISE evolution in WISE-GRASS-FDAF-PEM-AFROW with $n_A = 55$ and speech babble noise at 20 dB SNR.

Bibliography

[1] S. Haykin, Adaptive Filter Theory, Upper Saddle River, New Jersey: Prentice Hall, 2002.

[2] A. Spriet, I. Proudler, M. Moonen, J. Wouters, Adaptive feedback cancellation in hearing aids with linear prediction of the desired signal, IEEE Trans. Signal Process. 53, no. 10 (2005) 3749–3763.

[3] K. Ngo, T. van Waterschoot, M. G. Christensen, S. H. J. M. Moonen, J. Wouters, Prediction-error-method-based adaptive feedback cancellation in hearing aids using pitch estimation, in: Proc. 18th European Signal Process. Conf. (EUSIPCO'10), Aalborg, Denmark, 2010.

[4] T. van Waterschoot, G. Rombouts, P. Verhoeve, M. Moonen, Double-talk-robust prediction error identification algorithms for acoustic echo cancellation, IEEE Trans. Signal Process. 55, no. 3 (2007) 846–858.

[5] D. L. Duttweiler, A twelve-channel digital echo canceler, IEEE Trans. Commun. 26, no. 5 (1978) 647–653.

[6] H. K. Jung, N. S. Kim, T. Kim, A new double-talk detector using echo path estimation, Speech Commun. 45, no. 1 (2005) 41–48.

[7] G. Enzner, P. Vary, Frequency-domain adaptive Kalman filter for acoustic echo control in hands-free telephones, Signal Process. 86, no. 6 (2006) 1140–1156.

[8] S. Gustafsson, F. Schwarz, A postfilter for improved stereo acoustic echo cancellation, in: 1999 Int. Workshop Acoustic Echo Noise Control (IWAENC'99), Pocono Manor,Pennsylvania, Sep. 1999, pp. 32–35.

[9] P. Loizou, Speech Enhancement: Theory and Practice, Boca Raton, Florida: Taylor and Francis., 2007.

[10] T. S. Wada, B.-H. Juang, Towards robust acoustic echo cancellation during double-talk and near-end background noise via enhancement of residual echo, in: Proc. 2008 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'08), Las Vegas, USA, Mar. 2008, pp. 253–256.

[11] T. S. Wada, B.-H. Juang, Enhancement of residual echo cancellation for improved acoustic echo cancellation, in: Proc. 15th European Signal Process. Int. Conf. (EUSIPCO'07), Poznan, Poland, Sep. 2007, pp. 1620–1624.

[12] V. J. Mathews, Z. Xie, A stochastic gradient adaptive filter with gradient adaptive step size, IEEE Trans. Signal Process. 41, no. 6 (1993) 2075–2087.

[13] Y. Zhang, J. A. Chambers, W. Wang, P. Kendrick, T. J. Cox, A new variable step-size lms algorithm with robustness to nonstationary noise, in: Proc. 2007 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'07), Vol. 3, Honolulu, Hawaii, USA, Apr. 2007, pp. 1349–1352.

[14] W.-P. Ang, B. Farhang-Boroujeny, A new class of gradient adaptive step-size LMS algorithms, IEEE Trans. Signal Process. 49, no. 4 (2001) 805–810.

[15] H.-C. Shin, A. H. Sayed, W.-J. Song, Variable step-size NLMS and affine projection algorithms, IEEE Signal Process. Lett. 11, no. 2 (2004) 132–135.

[16] Y. Zhang, N. Li, J. A. Chambers, Y. Hao, New gradient-based variable step size lms algorithms, EURASIP J. Advances Signal Process. 2008, no. 105.

[17] T. Creasy, T. Aboulnasr, A projection-correlation algorithm for acoustic echo cancellation in the presence of double talk, in: Proc. 2000 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'00), Vol. 1, Istanbul, Turkey, June 2008, pp. 436–439.

[18] K. Ozeki, T. Umeda, An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties, Electronics and Communication in Japan 67, no. 5 (1984) 19–27.

[19] J. Benesty, H. Rey, L. R. Vega, S. Tressens, A nonparametric VSS NLMS algorithm, IEEE Signal Process. Lett. 13, no. 10 (2006) 581–584.

[20] M. A. Iqbal, S. L. Grant, Novel variable step size NLMS algorithms for echo cancellation, in: Proc. 2008 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'08), Las Vegas, USA, Mar. 2008, pp. 241 – 244.

[21] C. Paleologu, S. Ciochină, J. Benesty, Double-talk robust VSS-NLMS algorithm for under-modeling acoustic echo cancellation, in: Proc. 2008 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'08), Las Vegas, USA, Mar. 2008.

[22] C. Paleologu, J. Benesty, S. Ciochină, A variable step-size affine projection algorithm designed for acoustic echo cancellation, IEEE Trans. Audio Speech Lang. Process. 16, no. 8 (2008) 1466 – 1478.

[23] S. M. Kay, Fundamentals of statistical signal processing: estimation theory, Upper Saddle River, New Jersey: Prentice Hall, 1993.

[24] T. van Waterschoot, M. Moonen, Double-talk robust acoustic echo cancellation with continuous near-end activity, in: Proc. 13th European Signal Process. Conf. (EUSIPCO'05), Antalya, Turkey, 2005.

[25] L. Ljung, System Identification: Theory for the user, Englewood Cliffs, New Jersey: Prentice Hall, 1987.

[26] G. Rombouts, T. van Waterschoot, K. Struyve, M. Moonen, Acoustic feedback cancellation for long acoustic paths using a nonstationary source model, IEEE Trans. Signal Process. 54, no. 9 (2006) 3426–3434.

[27] T. van Waterschoot, M. Moonen, Fifty years of acoustic feedback control: state of the art and future challenges, Proc. IEEE 99, no. 2 (2011) 288–327.

[28] T. van Waterschoot, M. Moonen, Adaptive feedback cancellation for audio applications, Signal Process. 89, no. 11 (2009) 2185–2201.

[29] K. Ngo, T. van Waterschoot, M. G. Christensen, M. Moonen, S. H. Jensen, J. Wouters, Adaptive feedback cancellation in hearing aids using a sinusoidal near-end signal model, in: Proc. 2010 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'10), Dallas, USA, 2010.

[30] K. Ngo, T. van Waterschoot, M. G. Christensen, M. Moonen, S. H. Jensen, Improved prediction error filters for adaptive feedback cancellation in hearing aids, Elsevier Signal Processing.

[31] J. M. Gil-Cacho, T. van Waterschoot, M. Moonen, S. H. Jensen, Transform domain prediction error method for improved acoustic echo and feedback cancellation, in: Proc. 20th European Signal Process. Int. Conf. (EUSIPCO'12), Bucharest, Rumania, Aug. 2012, pp. 2422–2426.

[32] J. M. Gil-Cacho, T. van Waterschoot, M. Moonen, S. H. Jensen, A frequency-domain adaptive filtering (FDAF) prediction error method (PEM) for double-talk-robust acoustic echo cancellation, in: Technical Report KULuven, ESAT-SCD, Aug. 2013, (online $http : //homes.esat.kuleuven.be/\ dspuser/abstract13 - 13_2.html$).

[33] T. Trump, A frequency domain adaptive algorithm for colored measurement noise environment, in: Proc. 1998 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP'98), Seattle, USA, Mar. 1998, pp. 1705–1708.

[34] J. J. Shynk, Frequency-domain and multirate adaptive filtering, IEEE Signal Process. Mag. 9, no. 1 (1992) 14–37.

[35] N. J. Bershad, P. L. Feintuch, Analysis of the frequency domain adaptive filter, Proc. IEEE 67, no. 12 (1979) 1658–1659.

[36] M. H. Hayes, Statistical Digital Signal Processing and Modeling, New York, NY: John Wiley & Sons, Inc., 1996.

[37] N. Krishnamurthy, J. H. L. Hansen, Babble noise: Modeling, analysis, and applications, IEEE Trans. Audio Speech Lang. Process. 17, no. 7 (2009) 1394–1407.

[38] C. Rohrs, R. Younce, Double talk detector for echo canceler and method (04 1990).
URL `http://www.patentlens.net/patentlens/patent/US4918727/`

[39] T. Gänsler, S. L. Gay, M. M. Sondhi, J. Benesty, Double-talk robust fast converging algorithms for network echo cancellation, IEEE Trans. Speech Audio Process. 8, no. 6 (2000) 656–663.

[40] H. Rey, L. R. Vega, S. Tressens, J. Benesty, Variable explicit regularization in affine projection algorithm: Robustness issues and optimal choice, IEEE Trans. Signal Process. 55, no. 5 (2007) 2096–2108.

[41] B. Farhang-Boroujeny, Adaptive filters: Theory and applications, Chichster, UK: Wiley, 1998.

Part IV

Nonlinear Adaptive Filters

Chapter 7

Linear-in-the-parameters nonlinear adaptive filters for acoustic echo cancellation

Study and characterization of odd and even nonlinearities in electrodynamic loudspeakers by means of periodic random-phase multisine
Published in Proc. of the 127th Audio Engineering Society (AES) convention, NY, USA, Oct. 2009.

Linear-in-the-parameters nonlinear adaptive filters for acoustic echo cancellation
Accepted with major revision in Journal of the Audio Engineering Society (JAES), July 2013.

Jose Manuel Gil-Cacho, Toon van Waterschoot, Marc Moonen and Søren Holdt Jensen

Contributions of first author

- literature study
- co-development of the nonlinear algorithms
- design of simulation scenarios
- software implementation and computer simulations
- co-interpretation of simulation results
- text redaction and editing

Abstract

In this chapter, we consider the description of odd and even nonlinearities in electrodynamic loudspeakers by means of periodic random-phase multisine signals. It is shown that the odd nonlinear contributions are more predominant than the even ones. This fact implies that *at least* a 3rd-order nonlinear model of the loudspeaker should be used, which is in clear conflict with the extensive, almost unique, use of second-order (quadratic) Volterra filters. We, therefore, consider the identification and validation of a model of the loudspeaker using several linear-in-the-parameters nonlinear adaptive filters, namely, Hammerstein and Legendre polynomial filters of various orders, and a simplified 3rd-order Volterra filter of various lengths. In our measurement set-up, the obtained results imply however, that a 3rd-order nonlinear filter can not capture all the nonlinearities, meaning that odd and even nonlinear contributions are produced by higher-order nonlinearities. Legendre polynomial filters are shown to improve performance monotonically with increasing order whereas Hammerstein filters do not. In the simplified 3rd-order Volterra filter case, the performance is remarkably poorer than in the Legendre polynomial case for the same filter length. The differences between identification and validation appear to be very small in the Legendre polynomial filter case. However, the simplified 3rd-order Volterra filter presents clear signs of overfitting.

7.1 Introduction

ACOUSTIC echo cancellation (AEC) is used in many speech communication systems where the existence of echoes degrades the speech intelligibility and listening comfort [1], [2]. Applications range from mobile or hands-free telephony to teleconferencing or voice over IP (VoIP) services, which are often integrated in smartphones, tablets, notebooks, laptops, etc. While mobile devices are becoming smaller, the demand for quality, performance, and special features in audio products is increasing. Moreover, the need for higher sound pressure levels, provided by smaller devices, presents a huge challenge to engineers who have to deal with (usually cheap) loudspeakers working close to saturation.

The general set-up for an AEC is depicted in Figure 7.1. A far-end speech signal $u(t)$ is played back into an enclosure (i.e., the room) through a loudspeaker. In the room, there is a microphone to record a near-end speech signal, which is to be transmitted to the far-end side. An acoustic echo path H between the loudspeaker and the microphone exists, so that the microphone (i.e., desired) signal $d(t) = y(t) + n(t)$ contains an undesired echo $y(t)$ plus the background noise $n(t)$. The echo signal $y(t)$ can be considered as the far-end signal $u(t)$ filtered by the loudspeaker-enclosure-microphone (LEM) system's response. An

acoustic echo canceler seeks to cancel the echo signal component $y(t)$ in the microphone signal $d(t)$, ideally leading to an *echo-free* error (or residual) signal $e(t)$, which is then transmitted to the far-end side. This is done by subtracting an estimate of the echo signal $\hat{y}(t)$ from the microphone signal, i.e., $e(t) = d(t) - \hat{y}(t) = y(t) - \hat{y}(t) + n(t)$. Thus, the main objective of an echo canceler is to identify a model that represents a best fit to the LEM system.

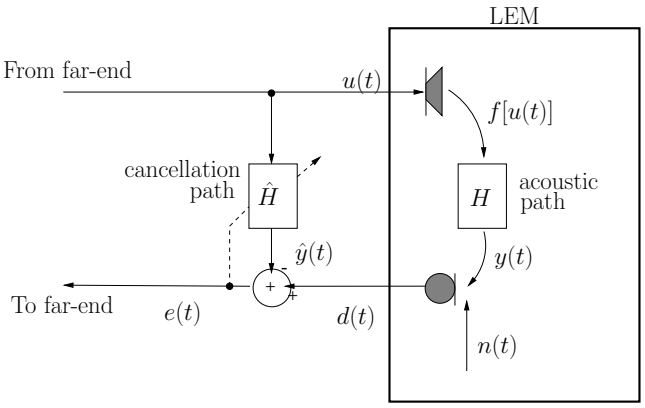


Figure 7.1: Typical set-up for an AEC.

Typically, a linear finite-impulse-response (FIR) filter is assumed to provide a sufficiently accurate model for the LEM system, leading to a class of linear-adaptive-filter-based AEC systems. This assumption may be valid for some components in the LEM system. In fact, even in nonlinear AEC (NLAEC), the acoustic path and microphone response may be modeled as a linear system. However, loudspeakers, as well as amplifiers, DACs, coders, etc., included in the LEM system introduce nonlinear distortion and must be modeled as a nonlinear system [3]. If the output $f[u(t)]$, in Figure 7.1, is a highly nonlinear function of $u(t)$, poor performance may be expected from any linear-adaptive-filter-based AEC system [4]. The error signal is contaminated by the nonlinear contribution that is not modeled by the linear adaptive filter making the adaptive filter to converge slowly or even diverge [5]. It is known that the nonlinearity also changes with time [6]- [8]. Therefore, a nonlinear adaptive filter is needed. Several nonlinear adaptive filters, intended to overcome the limitations of linear filters, have been used for NLAEC with more or less success.

Linear-in-the-parameters nonlinear adaptive filters, which require nonlinear expansions of the input vector, are appealing from an adaptive filtering perspective since standard linear adaptive filtering techniques can be applied and global optimality can be guaranteed [9]- [11]. The main problem, however, is that many more filter coefficients are needed than in the linear case, resulting in a large computational complexity together with inherently slow con-

vergence [12]- [15]. Volterra [16]- [18] and simplified Volterra [9] filters are commonly used linear-in-the-parameters nonlinear adaptive filters, although typically only very low nonlinear orders are considered due to computational complexity constraints.

The answer to 'what model represents a best fit to the LEM system?' is not easily given. Indeed, a nonlinear model may exhibit an excellent performance for one particular system but may be useless for a second one. A number of contributions found in literature appear to select a nonlinear model based on faith more than on a true examination of the LEM system. For instance, 2nd-order Volterra filter are widely used where most of the times only a single reference [13] is used to justify this choice. This nonlinear model assumption may be accurate in some cases, of course, but in other cases without previous study of the system it seems reasonably doubtful. Another problem found in the NLAEC literature is that only the attenuation of the residual echo is considered without taking any prevention for overfitting. This clearly leads to some misinterpretation of the obtained results and a lack of robustness of the adaptive filter.

The contribution of this chapter is twofold, namely, (1) to propose a method for analyzing and describing the nonlinear response of LEM systems, and (2) to compare and analyze several models adopted in linear-in-the-parameters nonlinear adaptive filters that are typically used in NLAEC.

The outline of the chapter is as follows. In Section 7.2, we present the type of excitation signal used in the considered analysis method, and explain the nonlinear response of a nonlinear system based on this type of signal. In Section 7.5, the analysis method for classification of nonlinearities is presented and one example is provided based on a simulated Hammerstein system. In Section 7.6, the structure of the class of linear-in-the-parameters nonlinear adaptive filters is explained and a brief review of the normalized least mean squares (NLMS) algorithm [19] is also provided. The more specific sub-class of nonlinear filters without cross terms is explained in Section 7.7, where Hammerstein filters and Legendre polynomial filters are elaborated on. Volterra filters and simplified Volterra filters which are a sub-class of nonlinear filters with cross terms are elaborated on in Section 7.8. In Section 7.9, the identification and model validation of a simulated 5th-order Hammerstein system using both a Hammerstein filter and a Legendre polynomial filter is performed. In Section 7.10, the identification and model validation of a real loudspeaker is performed using Hammerstein filters of different orders, Legendre polynomial filter of different orders, and a simplified 3rd-order Volterra of different sizes. Finally, Section 7.11 concludes the chapter.

7.2 Excitation signal and response of nonlinear systems

A signal $u(t)$ is a *random-phase multisine* [20] excitation when

$$u(t) = \sum_{k=1}^{F} b_k \cos\left(i_k \omega_0 t + \phi_k\right) \tag{7.1}$$

where $\omega_0 = 2\pi f_0$, and f_0 is the frequency of the first harmonic, i.e., the fundamental frequency. The harmonics are classified as *odd harmonics* and *even harmonics* which refer to odd multiples or even multiples of the fundamental frequency, respectively. F is the number of harmonics, b_k is the deterministic amplitude of the kth harmonic and $i_k \in \mathbb{R}$. If $i_k \in \{k \mid k = 1, 2, ..., F\}$, then all odd and even harmonics are considered. The phases ϕ_k are a realization of independent uniformly distributed random processes on [0,2π) such that $\mathcal{E}\left\{e^{j\phi_k}\right\} = 0$ where $\mathcal{E}\{\cdot\}$ represents the expected value operation. Periodic random-phase multisine signals combine the advantage of broad-band signals and periodicity (i.e., discrete spectrum), while providing total control over the amplitude of each harmonic. In the frequency domain, the signal in (7.1) is given as

$$U(j\omega) = \sum_{k=-F}^{F} a_k \delta(\omega - i_k \omega_0) e^{j\phi_k} \tag{7.2}$$

where $a_{-k} = a_k = b_k/2$, $i_{-k} = -i_k$, $\phi_k = -\phi_k$ and $\delta(\cdot)$ represents the Dirac delta function defined as

$$\delta(\omega - i_k\omega_0) \begin{cases} = 0 & \omega - i_k\omega_0 \neq 0 \\ = 1 & \omega - i_k\omega_0 = 0. \end{cases} \tag{7.3}$$

In order to understand the effect of nonlinearities on the output spectrum, let us consider the output signal from a static nonlinearity $y_3(t) = f[u(t)] = u^3(t)$, where $u(t)$ is a random-phase multisine. The operation of the nonlinearity corresponds to a time-domain multiplication of the input samples and, hence, a convolution in the frequency domain, so that

$$Y_3(j\omega) = \sum_{p=-F}^{F} \sum_{m=-F}^{F} \sum_{k=-F}^{F} a_p a_m a_k \delta(\omega - [i_p + i_m + i_k]\omega_0) e^{j(\phi_p + \phi_m + \phi_k)}. \tag{7.4}$$

In general, the output from a nonlinearity of order n consists on $(2F)^n$ contributions, i.e., of all possible combinations, with permutations, of n input harmonics. It is then clear that a 3rd-order nonlinearity generates harmonics at frequencies that are sums and differences of the frequency of three excited input harmonics. To clarify the effect of static nonlinearities we may consider two types of contributions, namely, *harmonic* contributions and *interharmonic* contributions.

- Harmonic contributions:

These contributions are generated by pairs of equal positive and negative harmonics, and also by one harmonic combined with pairs of equal positive and negative harmonics. For example, for a 2nd-order nonlinearity, a combination such as $(+3\omega_0 - 3\omega_0)$ results in a harmonic contribution at $0\omega_0$ (i.e., the so-called *direct current* (DC) term). For a 3rd-order nonlinearity, combinations such as $(+3\omega_0 + 3\omega_0 - 3\omega_0\ )$ and $(\ +3\omega_0 + \omega_0 - \omega_0)$ results in harmonic contributions at $+3\omega_0$. The phase of the harmonic contribution equals the phase of the corresponding input harmonic.

- Interharmonic contributions:

These contributions are generated by combinations of harmonics that follow a different pattern. For example, for a 2nd-order nonlinearity, the harmonic combination $(+3\omega_0 - 2\omega_0)$ results in an interharmonic contribution at ω_0, and, for a 3rd-order nonlinearity, the frequency combination $(+3\omega_0 + \omega_0 + \omega_0)$ generates an interharmonic contribution at $+5\omega_0$. The phases of these contributions vary depending on the phases of the specific input harmonics giving rise to them, i.e., $(+3\omega_0 + 3\omega_0 - \omega_0)$ gives rise to $+5\omega_0$ but with different phase compared to the contribution generated by $(+3\omega_0 + \omega_0 + \omega_0)$.

The specific influence of these two types of contributions will depend on the order of the nonlinearity, namely, *odd-order nonlinearities* and *even-order nonlinearities*. This is explained as follows.

- Odd-order nonlinearities:

The **harmonic contributions** coincide with the excited input harmonics. They always have the same phase as their coinciding excited input harmonic since they are generated by the combination of this input harmonic with pairs of equal positive and negative harmonics, where the phases cancel out. However, it should be noted that, although the phases cancel out, this is not true for the amplitudes which are multiplied together, and hence the resulting amplitudes depend on the specific frequency combinations. Moreover, if the input signal contains only odd harmonics then the **interharmonic contributions** occur only at odd harmonic frequencies.

- Even-order nonlinearities:

The **harmonic contributions** all occur at DC in this case, since they are generated by pairs of equal positive and negative frequencies. If the input signal contains only odd harmonics, then the **interharmonic contributions** occur at even harmonic frequencies.

		Harmonic content	
		Odd	Even
Nonlinearity	Odd	**Odd**	**Even**
	Even	**Even**	**Even**

Table 7.1: Odd / Even responses

In other words, as it is summarized in Table 7.1, a combination of *odd* harmonics generates *odd* harmonics only when input to an *odd* nonlinearity (i.e., an *odd* combination of *odd* harmonics generates *odd* harmonics only); however, a combination of *odd* harmonics generates *even* harmonics only when input to an *even* nonlinearity (i.e., an *even* combination of *odd* harmonics generates *even* harmonics only). On the other hand, a combination of *even* harmonics only generates *even* harmonics at the output of both an *even* and *odd* nonlinearity (i.e., there is no *even* combination of *even* harmonics that generates an *odd* harmonic).

7.3 Detection of nonlinear distortions

In [21], it is shown that, for random excitations, a nonlinear system can be represented by a linear system followed by an additive noise source (see Figure 7.2). For the considered class of excitation signals, the linear system provides the best linear approximation (BLA) of the output signal. The noise source represents the nonlinear distortions not represented by the linear system.

The measured frequency response function (FRF) $G(jw_k) = \dfrac{Y(jw_k)}{U(jw_k)}$ of the nonlinear system may be written as

$$\begin{aligned} G(jw_k) &= G_{\text{BLA}}(jw_k) + G_S(jw_k) \\ &= G_0(jw_k) + G_B(jw_k) + G_S(jw_k) \end{aligned} \tag{7.5}$$

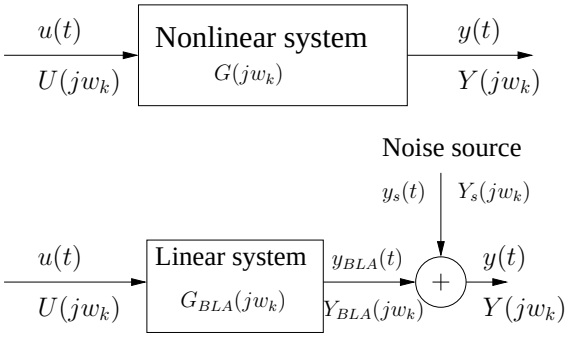


Figure 7.2: Representation of a nonlinear system as the combination of a linear system plus a noise source

Each term in expression (7.3) contributes to the echo path in different ways:

- $G_0(jw_k)$ is the true underlying linear system and it forms together with $G_B(jw)$ the BLA.
- $G_B(jw_k)$ is the systematic nonlinear contribution and it acts as a bias term on the estimated model. Only odd nonlinearities contribute to this term.
- $G_S(jw_k)$ is the stochastic nonlinear contribution and it acts as a noise source (as represented in Figure 7.2 where $Y_S(jw_k) = G_S(jw_k)U(jw_k)$). Odd and even nonlinearities contribute to this term.

In the next two sections, a method is reviewed that has been developed in [20] in the context of characterizing operational amplifiers, and which will be the basis for our analysis. This method is suitable for nonlinear (dynamic) systems that can be approximated arbitrarily well (in a least-squares sense) by a Volterra series. The method allows for a description of nonlinearities such as saturation (i.e., amplifiers, loudspeakers) and discontinuities (i.e., relays, quantizers), but excludes chaotic systems and systems producing subharmonics [20].

7.4 Estimating the level of the nonlinear noise source

The use of this class of signals allows for separating the nonlinear distortion and the disturbing noise levels. This is done by analyzing the variations over consecutive periods and over different realizations of the input, as will be explained.

7.4.1 Analysis method

To estimate the level of the noise source $G_S(jw_k)$ several phase realizations of a full multisine are used. A full multisine is a signal as in (2) where harmonics $k_i \in \{k \mid k = 1, 2, ..., F\}$ (i.e. even and odd harmonics) are excited. This will allow to have the odd and even responses together, and will provide a complete picture of the total level of the stochastic nonlinear distortions. The method starts from measured FRF data $G(jw_k) = \dfrac{Y(jw_k)}{U(jw_k)}$. The level of the stochastic nonlinear distortion is obtained through averaging over different random phase realizations of the multisine excitation as shown in Figure 7.3.

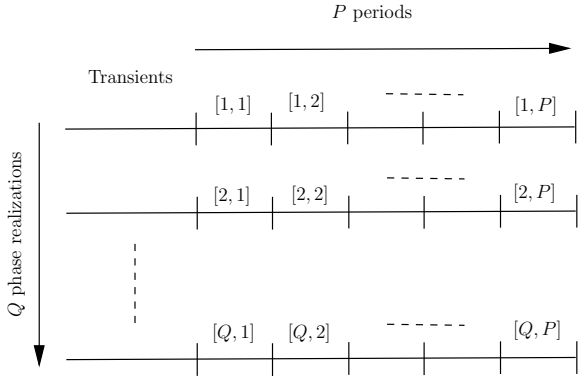


Figure 7.3: Measurement scheme where P periods of the steady state response to a random phase multisine is measured and repeated for Q different random phase realization

The nonparametric estimation of the BLA, the variance of the stochastic nonlinear distortions and the output noise variance, is based on the analysis of the sample mean and sample variance of the FRF over different multisine periods and phase realizations as shown in Figure 7.3. Since $N_y(jw_k)$ is a stochastic process and $Y_S(jw_k)$ is a periodic signal depending on the phase realization of the input signal, the FRF of the qth phase realization and pth period is related to the BLA, the stochastic nonlinear distortions and the output noise as

$$\begin{aligned} G^{[q,p]}(jw_k) &= \frac{Y^{[q,p]}}{U^{[q]}} \\ &= G_{\mathrm{BLA}}(jw_k) + \frac{Y_S^{[q]}}{U^{[q]}} + \frac{N_Y^{[q,p]}}{U^{[q]}} \end{aligned} \tag{7.6}$$

This shows that the sample variance over the P periods only depends on the

output noise, while the sample variance over the M realizations depends on both the stochastic nonlinear distortions and the output noise. From the $Q \times P$ noisy FRFs $G^{[q,p]}(jw_k)$, $q = 1, 2, ..., Q$ and $p = 1, 2, ..., P$, one calculates for each multisine phase realization the average FRF, $\hat{G}^{[q]}(jw_k)$, and its sample variance, $\hat{\sigma}^2_{\hat{G}^{[q]}}(jw_k)$, over the P periods as

$$\hat{G}^{[q]}(jw_k) = \frac{1}{P}\sum_{p=1}^{P} G^{[q,p]}(jw_k) \tag{7.7}$$

$$\hat{\sigma}^2_{\hat{G}^{[q]}}(jw_k) = \sum_{p=1}^{P} \frac{|G^{[q,p]}(jw_k) - \hat{G}^{[q]}(jw_k)|^2}{P(P-1)} \tag{7.8}$$

Additional averaging over the Q realizations gives the BLA $\hat{G}_{\text{BLA}}(jw_k)$ and the total variance $\hat{\sigma}^2_{\hat{G}_{\text{BLA}}}(jw_k)$ as

$$\hat{G}_{\text{BLA}}(jw_k) = \sum_{m=1}^{M} \frac{\hat{G}^{[q]}(jw_k)}{Q} \tag{7.9}$$

$$\hat{\sigma}^2_{\hat{G}_{\text{BLA}}}(jw_k) = \sum_{m=1}^{Q} \frac{|\hat{G}^{[q]}(jw_k) - \hat{G}_{\text{BLA}}(jw_k)|^2}{Q(Q-1)} \tag{7.10}$$

and an improved estimate of the noise variance

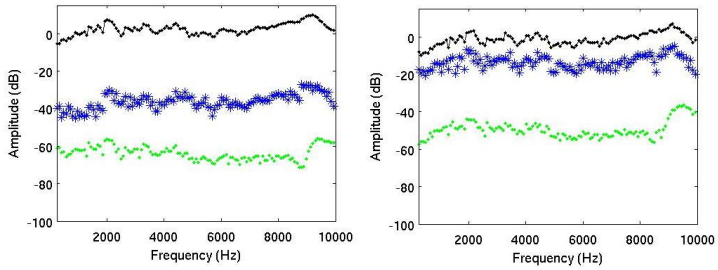
$$\hat{\sigma}^2_{\hat{G}_{\text{BLA}},n}(jw_k) = \frac{1}{Q^2}\sum_{q=1}^{Q} \hat{\sigma}^2_{\hat{G}^{[q]}}(jw_k) \tag{7.11}$$

Finally, the estimated variance of the stochastic nonlinear contributions is calculated as the difference between the total variance and the noise variance multiplied by the number of realizations (i.e. to account for the averaging process).

$$var\left[G_S(jw_k)\right] \approx Q\left[\hat{\sigma}^2_{\hat{G}_{\text{BLA}}(jw_k)} - \hat{\sigma}^2_{\hat{G}_{\text{BLA}},n}(jw_k)\right] \tag{7.12}$$

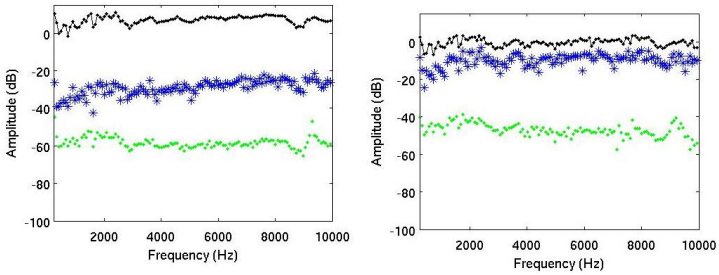
7.4.2 Results and discussion

Measurements are conducted in a room that is acoustically conditioned and prepared to have low reverberation times and listening comfort but that is



(a) (b)

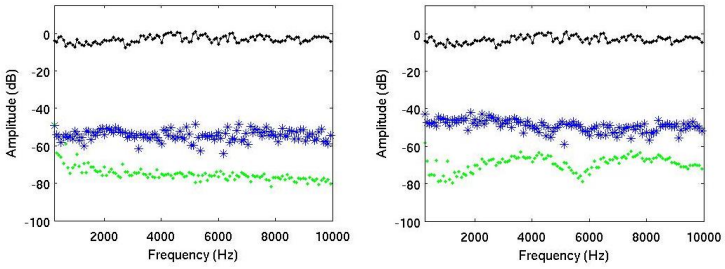
Figure 7.4: BLA, nonlinear distortion and noise level of the multimedia loudspeaker at RMS input signals: (a) 0.02 (b) 0.16.



(a) (b)

Figure 7.5: BLA, nonlinear distortion and noise level of the teleconferencing monitor at RMS input signals: (a) 0.02 (b) 0.16.

not anechoic. Hence, it is expected that some (linear) dynamics are added to the measured frequency response. The microphone is a condenser microphone, AKG CK97-C with an AKG SE300-B pre-amplifier, connected to a PC running Cool Edit pro 2.0 through a RME Multiface II sound card. It is expected that none of these elements will cause any significant harmonic distortion. Three different active loudspeakers are measured following the scheme of Figure 7.3, namely, a low-quality multimedia loudspeaker, a medium-quality Boss MA-12 monitor typically used in small teleconferencing rooms, and a high-quality Soundcraft Spirit Absolute 4P recording studio monitor. The quality of the loudspeakers and their uses differ greatly. The input signal relative input power RMS = 0.02 and 0.16. The low RMS value is set so that a non disturbing background noise is perceived and the high RMS value is set 18 dB higher. The first frequency with non-zero amplitude is $f_{\min} = 40$ Hz, and the maximum excited frequency $f_{\max} = 8$ kHz



(a) (b)

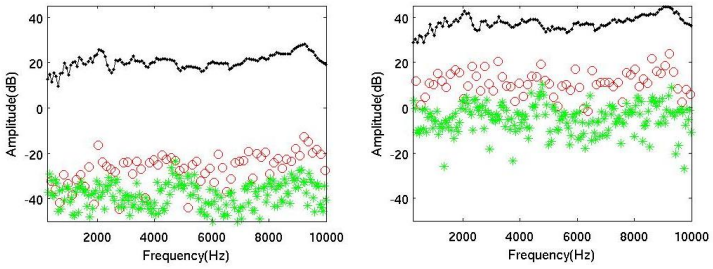
Figure 7.6: BLA, nonlinear distortion and noise level of the studio monitor at RMS input signals:: (a) 0.02 (b) 0.16.

The level of the stochastic nonlinear contribution $var\left[G_S(jw_k)\right]$ is calculated for different RMS values of a full multisine as explained in Section 7.4 with $P = 2$ and $Q = 7$. In the following figures the upper line $(-.-)$ represent the $\hat{G}_{\text{BLA}}$, stars $(\star)$ represent the stochastic nonlinear contribution and dots $(.)$ represent the output noise variance. It can be seen from Figure 7.4 and 7.5 that for the multimedia loudspeaker and the teleconferencing monitor the level of the stochastic nonlinear contribution increases steadily reaching significant levels quite easy. On the other hand, the nonlinear distortion in the studio monitor remains at low levels (50 dB below $\hat{G}_{\text{BLA}}$) for every input RMS value (see Figure 7.6). Active loudspeakers typically have a protection device (i.e. saturation curve) in the amplifier to avoid diaphragm damage at too high input signals. This saturation curve may also be a cause of nonlinear distortion at high amplitudes. It is clear from this section that nonlinearities are present in low quality loudspeaker and eventually in high quality loudspeakers as well.

7.5 Classification of nonlinearities

7.5.1 Analysis method

An *odd multisine* is a signal as in (7.1), where only odd harmonics are excited, i.e., $i_k \in \{(2k-1) \mid k = 1, 2, ..., F\}$. In the analysis that follows, a particular type of odd multisine will be used as excitation signal, which is referred to as an *odd-random multisine*. An odd-random multisine is an odd multisine where in every set of K consecutive odd harmonics one randomly-chosen odd harmonic is not excited. The complete set of non-excited odd harmonics is called the *harmonic grid*. The non-excited odd harmonics together with the even harmonics are called the *detection lines*.



(a) (b)

Figure 7.7: Output spectrum of the multimedia loudspeaker where $(-.-)$ is the output level at the excited lines, circles (o) represent the output level at non-excited odd harmonics and stars ($\star$) represent the level at non-excited even harmonics. RMS input signals: (a) 0.02 (b) 0.16.

As the input detection lines have zero magnitude, the presence of signal energy at the corresponding output detection lines is due to the nonlinear distortion of the system plus, probably, some measurement noise. The information about the nonlinear contribution (excluding the harmonic contribution generated by an odd-order nonlinearity) is obtained from the output DFT spectrum. The average output spectrum is calculated at all frequencies (excited and non-excited harmonics) [21], based on

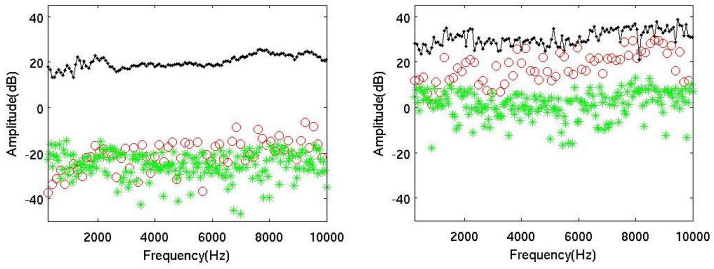
$$\hat{Y}(j\omega_k) = \frac{1}{P}\sum_{p=1}^{P} Y^{[p]}(j\omega_k) \quad k = 1, ..., F \tag{7.13}$$

where $\omega_k = i_k\omega_0$, and $Y^{[p]}(j\omega_k)$ is the DFT spectrum of one period p of the output signal. It follows the measurement scheme illustrated in Figure 7.3 only with one phase realization, i.e., $Q = 1$.

7.5.2 Results and discussion

The frequency range is the same as previously, but now only odd harmonics are excited with $K = 6$, $Q = 1$, and $P = 40$. The response of the high-quality studio monitor has not been considered here since the previous analysis shows a linear behavior. By inspecting the odd and even detection lines in the output DFT spectrum we can see the loudspeaker odd and even nonlinear response.

From Figure 7.7 and 7.8, it can be seen that at relatively high input signal levels, nonlinear distortions should be taken into account. In this case, odd nonlinearities are significantly more predominant that even nonlinearities in



(a) (b)

Figure 7.8: Output spectrum of the teleconferencing monitor where $(-.-)$ is the output level at the excited lines, circles (o) represent the output level at non-excited odd harmonics and stars ($\star$) represent the level at non-excited even harmonics. RMS input signals: (a) 0.02 (b) 0.16.

the full frequency range. This implies that the dominant nonlinearity must be modeled at least as a 3rd-order system.

7.6 Linear-in-the-parameters nonlinear adaptive filters

In this section, so-called linear-in-the-parameters nonlinear filters are reviewed, which are characterized by a linear dependence of the filter output on the filter coefficients. These filters are particulary suited to AEC, where adaptive filters are typically used to adapt to changes in the acoustic echo path, now including also the nonlinearities. Linear-in-the-parameters nonlinear adaptive filters then have derisable characteristics, namely, that they are inherently stable, and that they can converge to a globally minimum solution (in contrast to other types of nonlinear filters whose cost function may exhibit many local minima) avoiding the undesired possibility of getting stuck in a local minimum during adaptation.

In the general case, a linear-in-the-parameters nonlinear filter is described by the input-output relationship

$$y_F(t) = \mathbf{h}_F^T \mathbf{u}_F(t) \tag{7.14}$$

where $\mathbf{h}_F$ is a vector containing L filter coefficients and $\mathbf{u}_F(t)$ is the corresponding vector whose L elements are nonlinear combinations and/or expansions of the input samples in $\mathbf{u}(t) = [u(t), u(t-1), \cdots, u(t-N+1)]^T$. In fact, (7.14) can be interpreted as an FIR filter with L coefficients in which $\mathbf{u}_F(t)$ is used instead of the usual $\mathbf{u}(t)$. If $\mathbf{u}_F(t)$ is chosen equal to $\mathbf{u}(t)$, the filter is truly a

linear FIR filter with $L = N$ coefficients. In the nonlinear case, we may define $\mathbf{u}_F(t)$ and $\mathbf{h}_F$ as a set of sub-vectors $\mathbf{u}_r(t)$ and $\mathbf{h}_r$, with $r = 1, \ldots, M$. Thus, we may define the input vector as

$$\mathbf{u}_F(t) = \left[\mathbf{u}_1^T(t)\ \mathbf{u}_2^T(t)\ \cdots\ \mathbf{u}_M^T(t)\right]^T, \tag{7.15}$$

and then the output is given as

$$y_F(t) = \mathbf{h}_1^T\mathbf{u}_1(t) + \mathbf{h}_2^T\mathbf{u}_2(t) + \cdots + \mathbf{h}_M^T\mathbf{u}_M(t) \tag{7.16}$$

where the vector $\mathbf{h}_F$ has been divided in sub-vectors of matching lengths.

The next sections present a collection of linear-in-the-parameters nonlinear filters that have been successfully applied in NLAEC [9]- [13]. A complete review of linear-in-the-parameters nonlinear filters with and without cross terms can be found in [22].

7.6.1 Adaptive algorithm

Adaptive filter algorithms developed for the identification and tracking of linear FIR systems can also be applied to the nonlinear filter structure of (7.14). The popular NLMS algorithm [19], for instance, can be adopted, of which the update equations are given as

$$e(t) = d(t) - \hat{\mathbf{h}}_F^T(t)\mathbf{u}_F(t) \tag{7.17}$$

$$\hat{\mathbf{h}}_F(t+1) = \hat{\mathbf{h}}_F(t) + \mu\frac{\mathbf{u}_F(t)e(t)}{\mathbf{u}_F^T(t)\mathbf{u}_F(t) + \epsilon} \tag{7.18}$$

where ϵ is a small positive constant to avoid overflow.

7.7 Nonlinear filters *without* cross terms

This section introduces linear-in-the-parameters nonlinear filters involving only instantaneous nonlinear expansions of the input samples. This sub-class of filters includes Hammerstein filters [23] as well as filters involving nonlinear expansions based on orthogonal polynomials [22], [23].

7.7.1 Hammerstein filters

A Hammerstein filter is formed by a static nonlinearity followed by a linear filter $h(n)$. Since the nonlinearity is usually expressed as a polynomial function and the linear filter has a finite impulse response of N samples, it is considered

as a simple example of a truncated Volterra filter. Thus, for the Hammerstein filter, the input-output relation is given as

$$y_F(t) = \sum_{n=0}^{N-1} h(n) \sum_{m=1}^{M} g_m u^m(t-n) \tag{7.19}$$

including a linear term, and higher-order polynomial terms up to order M. This model, which is linear-in-the-parameters $h(n)g_m$, has been studied and used for many applications including NLAEC [12], [22]. The vector $\mathbf{u}_F(t)$ is a collection of instantaneous nonlinear expansions, using powers, of the input samples, as shown in the following expression

$$\begin{aligned} \mathbf{u}_F(t) = [&u(t),\ u(t-1),\ \cdots, u(t-N+1) \\ &u^2(t), u^2(t-1),\ \cdots, u^2(t-N+1) \\ &\vdots \\ &u^M(t), u^M(t-1), \cdots, u^M(t-N+1)]. \end{aligned} \tag{7.20}$$

It is worth noting that Hammerstein filters have been called "power filters" in [12]. As observed, the terms in (7.20) are not mutually orthogonal. In [12] a Gram-Schmidt orthogonalization of the input data matrix has been proposed for improved convergence. However the computational load of the orthogonalization is very high even when considering a low-order filter.

7.7.2 Filters based on orthogonal polynomials

Filters using nonlinear expansions within the family of orthogonal polynomials have been mentioned in [24]. While Hammerstein filters are based on a power-series expansion of the input signal, a better approximation can be achieved with signal-independent polynomials such as orthogonal polynomials in the interval $[-1, 1]$, as for example Legendre, Chebyshev, and Hermite polynomials.

In Legendre polynomial filters, the sub-vectors are formed with

$$L_1[u(t)] = u(t) \tag{7.21}$$

$$L_2[u(t)] = u^2(t) \tag{7.22}$$

$$\begin{aligned} L_{m+1}[u(t)] &= \frac{1}{m+1}\left[L_m[u(t)](2m+1)u(t) - mL_{m-1}[u(t)]\right] \\ &m = 3, ..., M \end{aligned} \tag{7.23}$$

Thus, a Legendre polynomial filter is described by the following input-output relationship

$$y_F(t) = \sum_{n=0}^{N-1} h(n) \sum_{m=1}^{M} g_m L_m\left[u(t-n)\right] \tag{7.24}$$

including a linear term, and higher-order polynomial terms up to order M. This model is again linear-in-the-parameters $h(n)g_m$. The vector $\mathbf{u}_F(t)$ is a collection of instantaneous nonlinear expansions, using Legendre polynomials, as shown in the following expression

$$\begin{aligned}\mathbf{u}_F(t) = &\big[L_1[u(t)],\ L_1[u(t-1)],\ \ \cdots, L_1[u(t-N+1)] \\ &L_2[u(t)],\ L_2[u(t-1)],\ \ \cdots, L_2[u(t-N+1)] \\ &\ \vdots \quad\ \vdots \\ &L_M[u(t)],\ L_M[u(t-1)],\ \ \cdots, L_M[u(t-N+1)]\big]\end{aligned} \tag{7.25}$$

The number of filter parameters in the Hammerstein filters as well as in the Legendre filters is $L_{\mathrm{H}} = L_{\mathrm{L}} = MN$.

7.8 Nonlinear filters *with* cross terms

The class of filters described by (7.14) includes also more general nonlinear filters involving cross terms, i.e., products of input samples with different time shifts, such as truncated Volterra filters [23] and simplified Volterra filters [9].

7.8.1 Volterra filters

Volterra filters are described by input-output relationships that result from two truncations of the discrete Volterra series [23], namely, a memory truncation $(N-1)$ of the Volterra kernel orders, and a nonlinearity order truncation (M) to limit the number of Volterra kernels. The Volterra kernels can be assumed to be symmetric and, thus, the Volterra filter input/output relationship can be cast in the compact triangular form as

$$\begin{aligned}y(t) = &\sum_{n_1=0}^{N-1} h_1(n_1)u(t-n_1) \\ &+ \sum_{n_1=0}^{N-1}\sum_{n_2=n_1}^{N-1} h_2(n_1,n_2)u(t-n_1)u(t-n_2) + \cdots \\ &+ \sum_{n_1=0}^{N-1}\cdots\sum_{n_M=n_{M-1}}^{N-1} h_M(n_1...n_M)u(t-n_1)...u(t-n_M)\end{aligned} \tag{7.26}$$

The model in (7.26) includes an FIR filter $h_1(n_1)$ and higher-order kernels $h_m(n_1,\cdots,n_m)$ with $m = 2,\ldots,M$. For instance, a 3rd-order Volterra filter, which includes a 2nd-order and a 3rd-order kernel, involves the following nonlinear expansions

$$\mathbf{u}_2(t) = [u^2(t), u(t)u(t-1), ..., u^2(t-1), \\ u(t-1)u(t-2), ..., u^2(t-N+1)]^T$$

with dimension $N(N+1)/2$ and

$$\mathbf{u}_3(t) = [u^3(t), u(t)u(t)u(t-1), ..., u(t)u(t-1)u(t-2), \\ ..., u^3(t-N+1)]^T$$

with dimension $N(N+1)(N+2)/6$. Thus, the total number of filter coefficients is $L_{\mathrm{V}} = N + N(N+1)/2 + N(N+1)(N+2)/6$.

7.8.2 Simplified 3rd-order Volterra kernel

Based on the simplified 2nd-order Volterra kernel presented in [9], we propose a simplified 3rd-order Volterra kernel. In [9], a simplification is made where only a reduced number of terms is considered depending on the desired complexity. By analyzing the 2nd-order Volterra kernel obtained from a loudspeaker, it has been observed that the coefficients with the most significant amplitude correspond to small values of $N_d = n_2 - n_1$. For this reason, a good approximation of a 2nd-order Volterra filter may be achieved when coefficients corresponding to large values of $n_2 - n_1$ are not selected, i.e.,

$$V_{2K}(t) = \sum_{n_1=n_2=0}^{N-1} h_2(n_1, n_2)u(t-n_1)u(t-n_2) \tag{7.27}$$

$$+ \sum_{n_1=0}^{N-2} \sum_{n_2=n_1+1}^{\min(n_1+N_d, N-1)} h_2(n_1, n_2)u(t-n_1)u(t-n_2). \tag{7.28}$$

In the 3rd-order case, there is no evidence a similar approximation is justified. However, with the idea of reducing computation, we propose the following simplification for the 3rd-order Volterra kernel,

$$V_{3K}(t) = \sum_{n_1=n_2=n_3=0}^{N-1} h_3(n_1, n_2, n_3)u(t-n_1)u(t-n_2)u(t-n_3) \tag{7.29}$$

$$+ \sum_{n_1=0}^{N-2} \sum_{n_2=n_1+1}^{\min(n_1+N_d, N-1)} \sum_{n_3=n_2+1}^{\min(n_2+N_d, N-1)} \\ h_3(n_1, n_2, n_3)u(t-n_1)u(t-n_2)u(t-n_3) \tag{7.30}$$

The vector $\mathbf{u}_2(t)$ is then a collection of 2nd-order combinations of input samples with different time shifts, i.e.,

$$\begin{aligned}\mathbf{u}_2(t) = [&u^2(t), u^2(t-1), ..., u^2(t-N+1)\\ &u(t)u(t-1), ..., u(t-N)u(t-N+1)\\ &\cdots\\ &u(t)u(t-N+1)]^T\end{aligned} \tag{7.31}$$

with dimension $N(N+1)/2$. Similary, the vector $\mathbf{u}_3(t)$ is a collection of 3rd-order combinations of input samples with different time shifts, i.e.,

$$\begin{aligned}\mathbf{u}_3(t) = [&u^3(t), u^3(t-1), ..., u^3(t-N+1)\\ &u(t)u(t-1)u(t-2), ..., u(t-N-1)u(t-N)u(t-N+1)\\ &\cdots\\ &u(t)u(t-N)u(t-N+1)]^T\end{aligned}$$

with dimension $N(N_d+1)(N_d+2)/6$. Thus, the total number of filter coefficients is $L_{\rm SV} = L + N(N_d+1)/2 + N(N_d+1)(N_d+2)/6$.

7.9 Simulated Hammerstein system identification

We consider the Hammerstein system illustrated in Figure 7.9, where a 5th-order power-series nonlinearity, i.e., $f[u(t)] = u(t) + \alpha u^2(t) + \beta u^3(t) + \gamma u^4(t) + \delta u^5(t)$ is used. The linear filter is a 16-coefficient finite impulse response (FIR) low-pass filter (LPF) with cut-off frequency 1/4 of the sampling frequency. Figure 7.10 shows the simulated Hammerstein system input and output signals. The input signal is an odd-random multisine, the harmonic grid parameter $K = 6$, and the analysis method is based on (7.13). Figure 7.10(a) shows the spectrum of the input odd-random multisine after conversion to a 16-bit pulse code modulation (PCM) representation. The round-off noise appears as a low-level (-60 dB) odd and even harmonics contribution. Figure 7.10(b) shows the spectrum of the Hammerstein system output signal. The spectrum of the nonlinear contribution of $f[u(t)]$ is clearly shown, where α, β, γ, and δ are chosen such that the nonlinear contributions are scaled to -20 dB compared to the linear contribution.

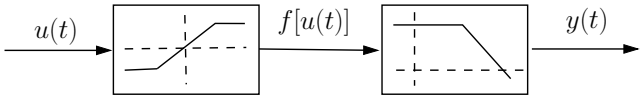
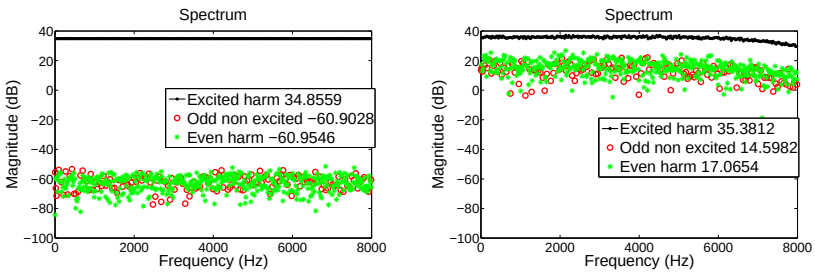


Figure 7.9: Hammerstein system: Static nonlinearity followed by a linear filter.

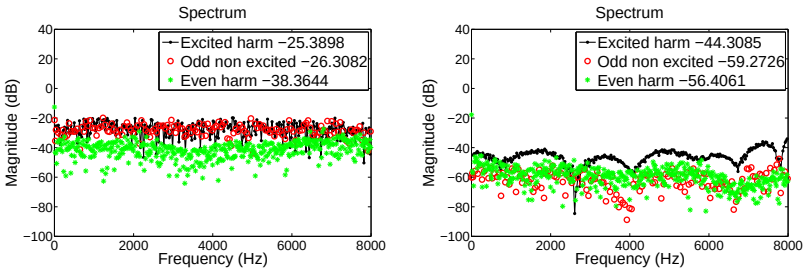


(a) (b)

Figure 7.10: Hammerstein system analysis using odd-random multisine. Spectrum estimated using (7.13). (a) Input spectrum (16-bit precision PCM represenation). (b) Output spectrum.

The identification of this simulated Hammerstein system is performed using NLMS and two different nonlinear filters, a 5th-order Hammerstein filter, and a 5th-order Legendre polynomial filter. The input signal is an odd-random multisine, the fundamental frequency is 2 Hz, the first frequency with non-zero amplitude is $f_{\min} = 40$ Hz, the maximum excited frequency $f_{\max} = 8$ kHz, the sampling frequency is 32 kHz and the harmonic grid parameter is $K = 6$. From the total number of periods, i.e., $P = 40$, we use the first 35 periods for identification and the last 5 periods for validation. In identification, the analysis method is applied to the residual signal after the adaptive filters have sufficiently converged. In validation, we fix the filter parameters obtained previously in the identification and analyze the residual signal. A very useful feature of the analysis method is that one can easily calculate the power of the different contributions independently. The values we show correspond to the averaged power of the excited odd, non-excited odd and even harmonics independently.

Figure 7.11 shows that the Legendre polynomial filter achieves significantly better performance than the Hammerstein filter. Interestingly, the Legendre polynomial filter achieves -59 dB in the non-excited odd harmonics and -57 dB in the even harmonics. These values are nearly the same as the level of the nonlinear distortion due to 16-bit format, see Figure 7.10(b). These two facts, shed some light on the benefits of using orthogonal polynomial expansions (e.g., Legendre) if high orders are considered. It is known that non-orthogonal polynomials (e.g., power series) are not well behaved when using high orders. Indeed, even when the underlying nonlinear and linear systems are exactly known, the (high order) Hammerstein filter does not provide a good approximation.



(a) (b)

Figure 7.11: Results of simulated **Hammerstein system** identification using NLMS adaptation and where the nonlinearity is a 5th-order power-series expansion. (a) Validation of the 5th-order **Hammerstein filter**. (b) Validation of the 5th-order **Legendre polynomial filter**

7.10 Loudspeaker identification

The loudspeaker is a Boss MA-12 active monitor, which is measured using the same input signal as in the previous section, shown in Figure 7.10(a). The lower level is set so that a non disturbing background noise is perceived and the high level is set 10 dB higher.

Figure 7.12(a) shows the spectrum of the output signal of the loudspeaker at low input level where the non-excited odd and even harmonics levels are -24 dB and -22 dB respectively. Figure 7.12(b) shows the output spectrum of the loudspeaker at high input level ($+10$ dB) where the even harmonics lay at -9 dB and the non-excited odd harmonics are at much higher level $+8$ dB. The loudspeaker clearly presents an odd-order nonlinearity that is more predominant than the even-order nonlinearity, which means that at least a 3rd-order nonlinear model must be considered for identification.

Figure 7.12(c) shows the estimated impulse response using a linear least-squares (LS) estimate based on the low level signal. Similarly, Figure 7.12(d) shows the estimated impulse response based on the high level signal. The estimated impulse response is clearly less noisy in the low level case, as the nonlinear distortion is around 47 dB lower compared to the high level case, as shown in Figures 7.12(a) and 7.12(b). From the estimated impulse response we define the length of the linear filter, i.e., 360 coefficients.

We now proceed with the identification of the loudspeaker and validation of the model when using a Hammerstein filter and a Legendre polynomial filter, with nonlinear orders 3 ,7, 15, 31, 47, 55, and 67. We also identify and validate the model when using a simplified 3rd-order Volterra filter with $N_d = N/32$ and

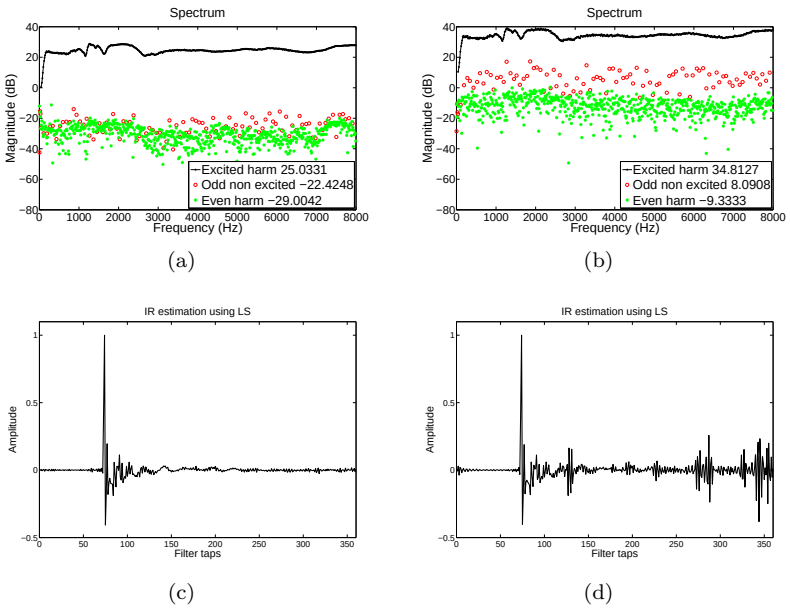


Figure 7.12: Loudspeaker measurements. (a)-(b) Output spectrum measured using two input levels with 10 dB difference. (c)-(d) Normalized impulse response estimates with least-squares using two input levels with 10 dB difference.

$N_d = N/16$ in both the 2nd and 3rd-order Volterra kernel. The reason for this choice is to keep the size of the filter reasonably small and comparable with the length of the other filters without cross terms. The value of the different averaged power spectra is shown in Table 7.2, 7.3, 7.4, where Id. stands for "in identification" and Val. for "in validation". In Figures 7.13 and 7.14, the spectrum obtained with the 3rd-order and 67th-order Hammerstein and Legendre polynomial filter is shown, and in Figure 7.15, the spectrum obtained with the simplified 3rd-order Volterra using $N_d = N/32$ and $N_d = N/16$ is shown.

In the 3rd-order case, the Hammerstein, Legendre, and simplified Volterra (both with $N_d = N/32$ and $N_d = N/16$) filter obtain similar values in identification and validation of the excited odd harmonics (around 5.5 dB Id. and 6.5 dB Val. respectively) and non-excited odd harmonics (around 4 dB Id. and 2.6 dB Val. respectively). For the even harmonics, the simplified Volterra filter achieves 2 dB performance improvement compared to the Hammerstein and Legendre polynomial filter, but this is only in the identification step. In the validation step, the Hammerstein and Legendre polynomial filter achieve a much better performance, -8.8 dB and -9.5 dB respectively, w.r.t. the simplified

Simp. Volt.			
	Size	$N_d = N/32$	$N_d = N/16$
Excited	Id.	5.6	5.5
	Val.	6.4	5.8
Odd	Id.	4	4.3
	Val.	2.6	2.7
Even	Id.	-14.9	-16
	Val.	-4.3	-2.9

Table 7.2: Loudspeaker identification using a simplified 3rd-order Volterra filter. The values correspond to the averaged power of the excited odd, non-excited odd and even harmonics independently.

Hammerstein								
	Order	3	7	15	31	47	55	67
Excited	Id.	5.3	3.8	3.8	3.8	3.8	3.8	3.8
	Val.	6.9	5.7	5.7	5.7	5.7	5.7	5.7
Odd	Id.	3.8	1.2	1.1	1.1	1.1	1.1	1.1
	Val.	2.6	-0.7	-0.8	-0.8	-0.8	-0.8	-0.8
Even	Id.	-12.7	-13.5	-13.7	-13.7	-13.7	-13.7	-13.7
	Val.	-8.8	-10.2	-10.4	-10.4	-10.4	-10.4	-10.4

Table 7.3: Loudspeaker identification using Hammerstein filter. The values correspond to the averaged power of the excited odd, non-excited odd and even harmonics independently.

Volterra filter using $N_d = N/32$ and $N_d = N/16$, which only achieves -4.3 dB and -2.9 dB respectively. Moreover, in the simplified 3rd-order Volterra filter case, the performance improvement between the $N_d = N/32$ and $N_d = N/16$ is nearly negligible.

The Legendre polynomial filter achieves a better performance than the Hammerstein filter during identification and validation. Interestingly, the Hammerstein filter performance does not improve with increasing order. Indeed, beyond 7th order there is no improvement whatsoever in identification or validation. On the other hand, the Legendre polynomial filter performance improves with increasing order in identification and, most importantly, in validation. For instance, using order 7, in the validation step the Legendre polynomial filter achieves Excited= 4.1 dB, Odd= -3.1 dB, and Even= -9.5 dB, whereas using order 67, in the validation step the Legendre polynomial filter achieves Excited= -1.9 dB, Odd= -4.3 dB, and Even= -13.2 dB. The 7th-order Legendre polynomial filter achieves a remarkable performance improvement over

Legendre								
	Order	3	7	15	31	47	55	67
Excited	Id.	5.3	2.9	2.3	1.1	0	-0.6	-1.7
	Val.	6.9	4.1	3	1.4	0.1	-0.7	-1.9
Odd	Id.	3.8	-0.2	-1	-1.4	-1.9	-2.3	-3.2
	Val.	2.6	-3.1	-3.4	-3.1	-3.3	-3.5	-4.3
Even	Id.	-12.5	-14	-14.6	-15.5	-16.2	-16.5	-16.6
	Val.	-9.5	-9.5	-9.6	-11.5	-12.3	-12.7	-13.2

Table 7.4: Loudspeaker identification using Legendre polynomial filter. The values correspond to the averaged power of the excited odd, non-excited odd and even harmonics independently.

the simplified Volterra filter, which in the validation step, achieves Excited= 5.8 dB, Odd= 2.7 dB, and Even= -2.9 dB only.

The differences between identification and validation, comparing the simplified 3rd-order Volterra filter with $N_d = N/16$ and the 67th-order Legendre polynomial filter, appear to be similar for the excited odd harmonics (0.3 dB and 0.2 dB respectively) and non-excited odd harmonics (1.6 dB and 0.9 dB respectively). However, for the even harmonics, the difference between identification and validation is remarkably high for the simplified 3rd-order Volterra filter with $N_d = N/16$ (13.1 dB) compared to the 31st-order Legendre filter case (3.4 dB). This is a clear sign of overfitting.

A general trend for the Hammerstein and Legendre filters is that the differences between identification and validation decrease with increasing order. This implies that the identified model is a better model for the actual system. However, for the simplified 3rd-order Volterra filter, the differences between identification and validation tend to increase with increasing N_d. In particular, for the even harmonics, these differences are quite high (10.6 dB and 13.1 dB respectively). The reason is that in the 3rd-order simplified Volterra filter, the order of the filter is constant, i.e., 3, but the length of the filter is increasing. The risk of overfitting in this case is clear, since keeping the order but increasing the filter length makes identification and validation differ considerably.

In [25] and [26], it is shown that a physical model based on a nonlinear version of the well known lumped parameter model of the loudspeaker can achieve a good representation of the odd harmonics when comparing simulations with measurements. This modeling approach is based on the dependence of the lumped parameters with the voice coil position. It is generally accepted that the variations of the magnetic force factor, the voice coil inductance and compliance of the suspension, are the main causes of nonlinear distortion. These are modeled as functions of cone displacement and therefore they are static

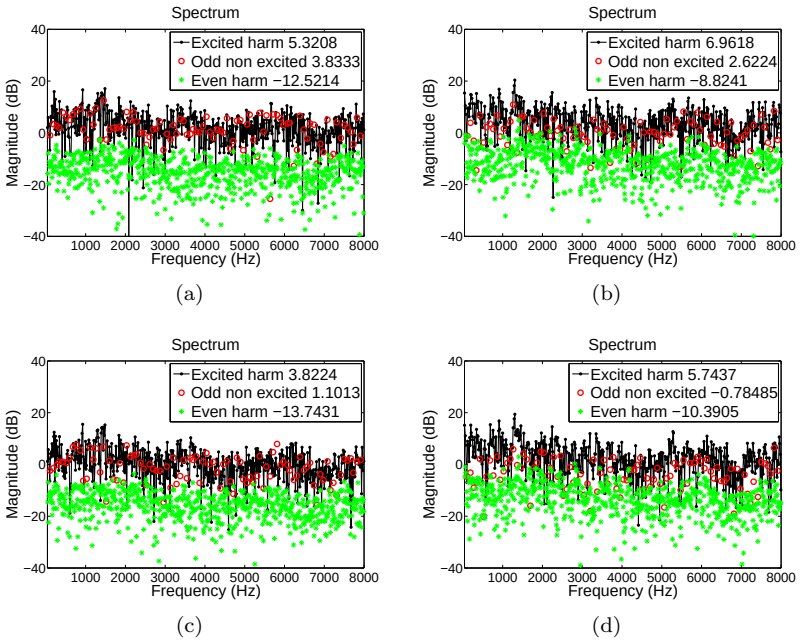


Figure 7.13: **Loudspeaker** identification and validation using a **Hammerstein filter** and NLMS. (a) Identification and (b) Validation using a 3rd-order filter. (c) Identification and (d) Validation using 67th-order filter.

elements in the model. This implies that the odd nonlinear distortions may be successfully modeled as produced by a static nonlinear system.

7.11 Conclusions

In this chapter, we have presented a method to analyze the nonlinear response, in terms of odd and even nonlinearities, of an electrodynamic loudspeaker by means of periodic random-phase multisine experiments. Moreover, we have considered the identification of a loudspeaker model by means of linear-in-the-parameters nonlinear adaptive filters. First of all, we have seen that the nonlinear odd contribution is much more predominant than the nonlinear even contribution. This mean that *at least* a 3rd-order nonlinear filter must be considered. By analyzing the residual signal after the adaptive filter has converged, we have shown, however, that a 3rd-order nonlinear filter is not sufficient to capture all the nonlinearities. This means that the odd and even nonlinear contributions are produced by higher-order nonlinearities. Second,

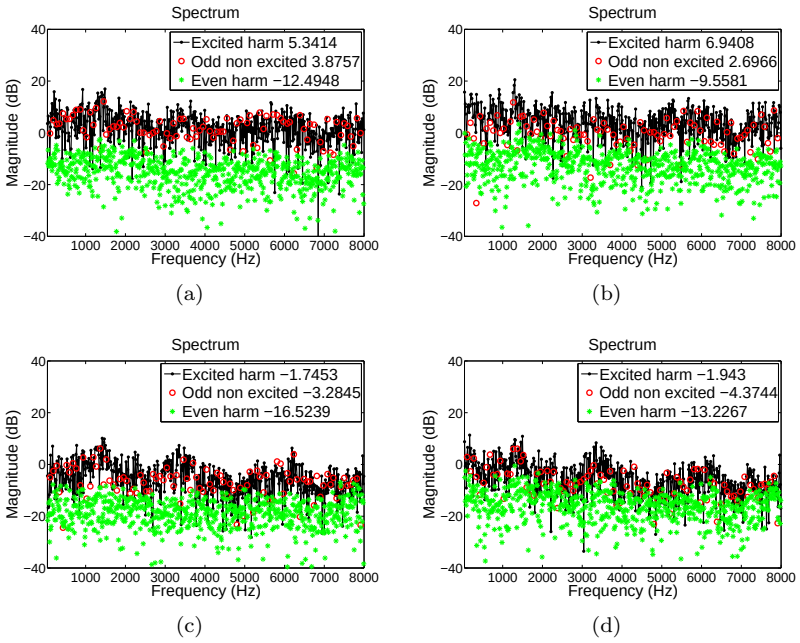


Figure 7.14: **Loudspeaker** identification and validation using a **Legendre polynomial filter** and NLMS. (a) Identification and (b) Validation using a 3rd-order filter. (c) Identification and (d) Validation using 67th-order filter.

we have compared nonlinear filters without cross-terms (i.e., Hammerstein and Legendre polynomial filters) and with cross-terms (i.e., simplified 3rd-order Volterra filters). Increasing the filter order beyond order 7 does not improve performance of Hammerstein filters. This may lead to misinterpreation of the results, concluding that this is a characteristic of the system and not of the nonlinear filter. Simplified Volterra filters, even when using low-order filters, present signs of overfitting with increasing length. Moreover, for the simplified 3rd-order Volterra filter, the performance is remarkably poorer than for the Legendre polynomial filter with the same filter length. On the other hand, Legendre polynomial filters have shown improved performance with increasing order. For the Legendre polynomial filter, the differences between identification and validation appear to be very small, which tend to decrease with increasing order. We have shown that in our set-up, a high-order nonlinear filter is much more important than the memory of a Volterra filter. Moreover, we have shown how easily a Volterra filter can give misleading results due to overfitting. This means that, although the attenuation of the error signal in AEC may be large, this does not necessarily mean that the identified model is a valid one.

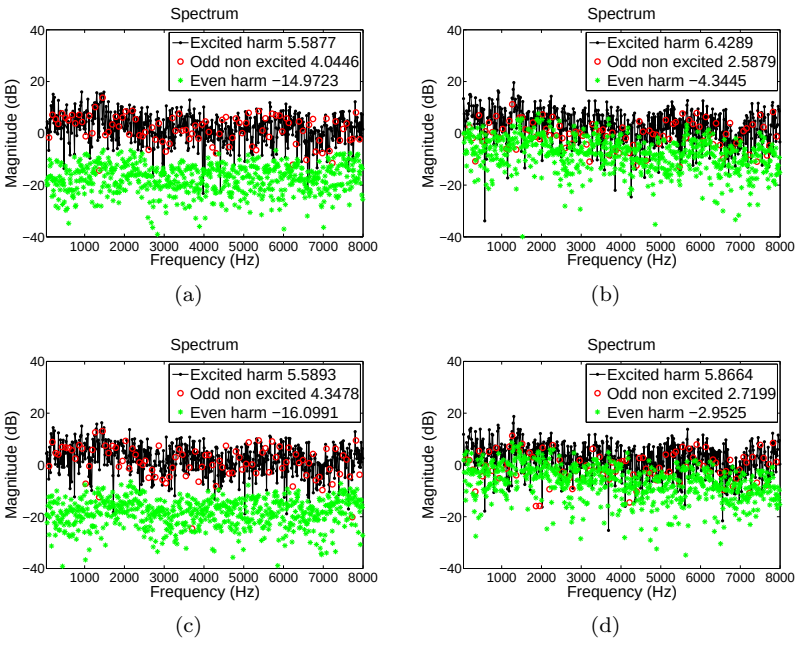


Figure 7.15: **Loudspeaker** identification and validation with a **simplified** 3**rd-order Volterra filter** and NLMS. (a) Identification and (b) Validation using $N_d = N/32$. (c) Identification and (d) Validation using $N_d = N/16$.

Bibliography

[1] J. Benesty, T. Gänsler, D. R. Morgan, M. M. Sondhi, and S. L. Gay, *Advances in Network and Acoustic Echo Cancellation*. Berlin: Springer-Verlag, 2001.

[2] P. Gil-Cacho, T. van Waterschoot, M. Moonen, and S. H. Jensen, "Multi-microphone acoustic echo cancellation using multi-channel warped linear prediction of common acoustical poles," in *Proc. 18th European Signal Process. Conf.*, (EUSIPCO'10), Aalborg, Denmark, Aug. 2010, pp. 2121–2125.

[3] ——, "Study and characterization of odd and even nonlinearities in electrodynamic loudspeakers by means of periodic random-phase multisine," in *Preprints AES 127th convention*, New York, USA, Oct. Oct. 2009, Preprint 7841.

[4] M. I. Mossi, C. Yemdji, N. Evans, C. Beaugeant, and P. Degry, "An assessment of linear adaptive filter performance with nonlinear distortions," in *Proc. 2011 IEEE Int. Conf. Acoust. Speech Signal Process.*, (ICASSP'11), Dallas, TX, USA, Mar. 2011, pp. 313 – 316.

[5] T. van Waterschoot, G. Rombouts, P. Verhoeve, and M. Moonen, "Double-talk-robust prediction error identification algorithms for acoustic echo cancellation," *IEEE Trans. Signal Process.*, vol. 55, no. 3, pp. 846–858, Mar. 2007.

[6] R. Ravaud, G. Lemarquand, and T. Roussel, "Time-varying non linear modeling of electrodynamic loudspeakers," *Applied Acoust.*, vol. 70, no. 3, pp. 450 – 458, Mar. 2009.

[7] C. A. Henricksen, "Heat-transfer mechanisms in loudspeakers: Analysis, measurement, and design," *J. Audio Eng. Soc.*, vol. 35, p. 778 791, Oct. 1987.

[8] C. Zuccatti, "Thermal parameters and power ratings of loudspeakers," *J. Audio Eng. Soc.*, vol. 38, pp. 34 –39, Feb. 1990.

[9] A. Fermo, A. Carini, and G. L. Sicuranza, "Simplified Volterra filters for acoustic echo cancellation in GSM receivers," in *Proc. 10th European Signal Process. Conf.*, (EUSIPCO'00), Tampere, Finland, Sept. 2000, pp. 2413–2416.

[10] G. L. Sicuranza, A. Carini, and A. Fermo, "Nonlinear adaptive filters for acoustic echo cancellation in mobile terminals," in *Nonlinear Signal and Image Processing: Theory, Methods, and Applications*, ser. The Electrical Engineering and Applied Signal Processing Series, K. E. Barner and G. R. Arce, Eds. Florida: CRC Press LLC, 2004, ch. 7, pp. 223–255.

[11] J. M. Gil-Cacho, M. Signoretto, T. van Waterschoot, M. Moonen, and S. H. Jensen, "Nonlinear acoustic echo cancellation based on a sliding-window leaky kernel affine projection algorithm," *IEEE Trans. Audio Speech Lang. Process.*, vol. 99, p. Early Access Articles, Apr. 2013.

[12] F. Kuech and W. Kellerman, "Orthogonalized power filters for nonlinear acoustic echo cancellation," *Signal Process.*, vol. 86, pp. 1168–1181, Nov. 2006.

[13] A. Stenger and R. Rabenstein, "Adaptive Volterra filters for acoustic echo cancellation," in *Proc. 1999 IEEE-EURASIP Workshop Nonlinear Signal and Image Process.*, (NSIP'99), Antalya, Turkey, June 1999.

[14] J. P. Costa, A. Lagrange, and A. Arliaud, "Acoustic echo cancellation using nonlinear cascade filters," in *Proc. 2003 IEEE Int. Conf. Acoust. Speech Signal Process.*, (ICASSP'03), Hong Kong, China, Apr. 2003, pp. 389–392.

[15] F. Kuech and W. Kellerman, "A novel multidelay adaptive algorithm for Volterra filters in diagonal coordinates," in *Proc. 2004 IEEE Int. Conf. Acoust. Speech Signal Process.*, (ICASSP'04), Montreal, Quebec, Canada, May 2004, pp. 869–872.

[16] F. Küch, "Adaptive polynomial filters and their application to nonlinear acoustic echo cancellation," Ph.D. dissertation, Friedrich–Alexander–Universität Erlangen–Nürnberg, 2005.

[17] A. Guérin, G. Faucon, and R. L. Bouquin-Jeannés, "Nonlinear acoustic echo cancellation based on Volterra filters," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 6, pp. 672–683, Nov. 2003.

[18] L. A. Azpicueta-Ruiz, M. Zeller, J. Arenas-García, and W. Kellermann, "Novel schemes for nonlinear acoustic echo cancellation based on filter combinations," in *Proc. 2010 IEEE Int. Conf. Acoust. Speech Signal Process.*, (ICASSP'10), Taipei, Taiwan, Apr. 2010, pp. 193–196.

[19] S. Haykin, *Adaptive Filter Theory.* Upper Saddle River, NY: Prentice Hall, 2002.

[20] R. Pintelon, G. Vandersteen, L. D. Locht, Y. Rolain, and J. Schoukens, "Experimental characterization of operational amplifiers: a system identification approach - part i: theory and simulations," *IEEE Trans. Instrum. Meas.*, vol. 53, no. 3, pp. 854 – 862, 2004.

[21] R. Pintelon and J. Schoukens, *System Identification: A frequency domain approach.* IEEE Press and John Wiley, 2001.

[22] G. L. Sicuranza and A. Carini, "On a class of nonlinear filters," in *Festschrift in Honor of Jaakko Astola on the Occasion of his 60th Birthday*, K. E. I. Tabus and M.Gabbouj, Eds. TICSP series, no. 47, Tampere, 2009, pp. 115 – 144.

[23] V. J. Mathews and G. L. Sicuranza, *Polynomial Signal Processing*. New York: Wiley, 2000.

[24] G. L. Sicuranza and A. Carini, "On the accuracy of generalized hammerstein models for nonlinear active noise control," in *Proc. 2006 IEEE Inst. Meas. Tech. Conf.*, Sorrento, Italy, Apr. 2006, pp. 24–27.

[25] F. T. Agerkvist,"Modelling loudspeaker non-linearities", in *Proc. 32nd AES Conference*, Hillerød Denmark, Sep. 2007.

[26] B. R. Pedersen, *Error correction of loudspeakers*. PhD thesis. Aalborg University, Denmark, May 2001.

Chapter 8

Nonlinear Acoustic Echo Cancellation based on a Sliding-Window Leaky Kernel Affine Projection Algorithm

Nonlinear Acoustic Echo Cancellation based on a Sliding-Window Leaky Kernel Affine Projection Algorithm

Jose M. Gil-Cacho, Marco Signoretto, Toon van Waterschoot, Marc Moonen and Søren Holdt Jensen

Published in IEEE Transactions on Audio Speech and Language Processing, vol. 21, no. 9, pp. 1867-1878, Sept. 2013.

Contributions of first author

- literature study
- co-development of the SWLKAPA algorithm
- design of simulation scenarios
- software implementation and computer simulations
- co-interpretation of simulation results
- text redaction and editing

Abstract

In this chapter, we propose an algorithm based on the kernel affine projection algorithm (KAPA). KAPA has been successfully applied to many areas in signal processing but not yet to nonlinear AEC (NLAEC). The contribution of this chapter is three-fold: (1) to apply KAPA to the NLAEC problem, (2) to develop a sliding-window leaky KAPA (SWL-KAPA) that is well suited for NLAEC applications, and (3) to propose a kernel function, consisting of a weighted sum of a linear and a Gaussian kernel. In our experiment set-up, the proposed SWL-KAPA for NLAEC consistently outperforms the linear APA, resulting in up to 12 dB of improvement in echo return loss enhancement (ERLE) at a computational cost that is only 4.6 times higher. Moreover, it is shown that the SWL-KAPA outperforms, by $4-6$ dB, a Volterra-based NLAEC, which itself has a much higher computational cost than the linear APA.

8.1 Introduction

ACOUSTIC echo cancellation (AEC) is used in many speech communication systems where the existence of echoes degrades the speech intelligibility and listening comfort [1], [2]. These applications range from mobile or hands-free telephony to teleconferencing or voice over IP (VoIP) services, which are often integrated in smartphones, tablets, notebooks, laptops, etc., which are becoming standard devices in our every-day life. While mobile devices are becoming smaller, the demand for quality, performance, and special features in audio products is increasing. Moreover, the need for higher sound pressure levels, provided by smaller devices, presents a huge challenge to engineers who have to deal with (usually cheap) loudspeakers working close to saturation.

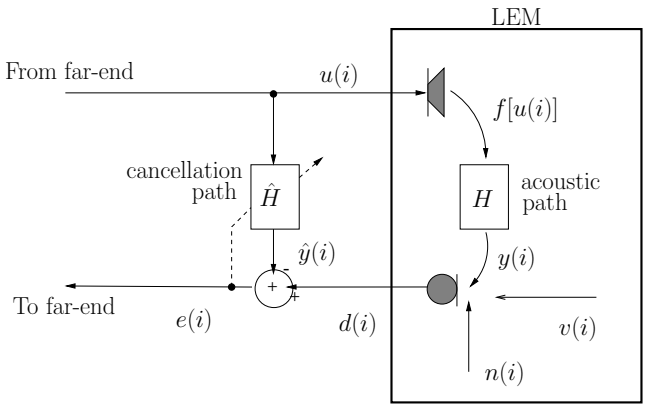


Figure 8.1: Typical set-up for AEC.

The general set-up for an AEC is depicted in Figure 8.1. A far-end speech signal $u(i)$ is played back into an enclosure (i.e., the room) through a loudspeaker. In the room, there is a microphone to record a near-end speech signal, which is to be transmitted back to the far-end side. An acoustic echo path H between the loudspeaker and the microphone exists, so that the microphone (i.e., desired) signal $d(i) = y(i) + v(i) + n(i)$ contains an undesired echo $y(i)$ plus the near-end speech $v(i)$ and the background noise $n(i)$. The echo signal $y(i)$ can be considered as the far-end signal $u(i)$ filtered by the echo path, i.e., the loudspeaker-enclosure-microphone (LEM) response. An acoustic echo canceler seeks to cancel the echo signal component $y(i)$ in the microphone signal $d(i)$, ideally leading to an *echo-free* error (or residual) signal $e(i)$, which is then transmitted to the far-end side. This is done by subtracting an estimate of the echo signal $\hat{y}(i)$ from the microphone signal, i.e., $e(i) = d(i) - \hat{y}(i) = y(i) - \hat{y}(i) + n(i)$, where a perfect double-talk detector (DTD) is assumed to be available so that periods when only the far-end speaker is active, i.e., $v(i) = 0, \forall i$, can be considered. Thus, the main objective of an echo canceler is to identify a model that represents a best fit to the echo path (i.e., similar to a system identification problem) or, in other words, a function $f(\cdot)$ that models the underlying mapping $y(i) = f[\mathbf{u}(i)]$, where $\mathbf{u}(i)$ is the input signal vector defined as $\mathbf{u}(i) = [u(i), u(i-1), ..., u(i-L+1)]^T \in \mathbb{R}^L$.

Typically, a linear finite-impulse-response (FIR) filter is assumed to provide a good model for the echo path, leading to a class of linear-adaptive-filter-based AEC systems. This assumption may be valid for some components in the LEM system. In fact, even in nonlinear AEC (NLAEC), the room (enclosure) and microphone responses may be modeled as linear systems. However, loudspeakers, as well as amplifiers, DACs, coders, etc., included in the LEM introduce nonlinear distortion and must be modeled as nonlinear systems [3]. In this chapter, we will show simulation results obtained from a simulated nonlinear system in a controlled scenario, so as to avoid other problems related to real AEC applications (e.g., insufficient-order adaptive filters, or scenarios comprising non-stationary and colored background noise). If the mapping between $\mathbf{u}(i)$ and $y(i)$ is highly nonlinear, poor performance may be expected from any linear-adaptive-filter-based AEC. Several nonlinear models, intended to overcome the limitations of linear filters, have been used for NLAEC with more or less success.

Neural networks have been used for NLAEC [4], [5], but in that case the adaptation consists of solving a non-convex optimization problem with many local minima, potentially leading to suboptimal solutions. A proper initialization is then required to avoid local minima, which is already a big challenge in itself. Structured and block-oriented nonlinear models [6], [7] have also been studied but, besides the need for proper initialization, the modeling capabilities are then limited to the specified nonlinear structure. Linear-in-the-parameters models that require nonlinear expansions of the input vector are appealing from

an adaptive filtering perspective since standard linear adaptive filtering techniques can be applied and global optimality can be guaranteed [8], [9]. The main problem however is that many more filter coefficients are needed than in the linear case, resulting in a large computational complexity together with inherently slow convergence [10]- [13]. Volterra [14]- [16] and simplified Volterra [8] filters are common solutions belonging to the linear-in-the-parameters family, although typically only very low nonlinear orders are considered due to computational complexity constraints.

On the other hand, kernel adaptive algorithms [17]- [20] and on-line learning algorithms [21], [22] have been subjects of great attention due to their good performance in nonlinear signal processing applications. Kernel methods are developed based on the theory of reproducing kernel Hilbert spaces (RKHS) [23]; a nonlinear model is obtained via a reproducing kernel by implicitly mapping data into a high-dimensional feature space. If the adaptive filtering operations can be expressed by particular vector inner products, then it is possible to apply the so-called *kernel trick*. The power of this idea is that the model parameters of the underlying nonlinear mapping $y(i) = f[\mathbf{u}(i)]$ are calculated by applying *nonlinear* methods on the input data, corresponding to *linear* operations in the feature space. This enables formulations where convergence to a global optimum may be guaranteed. In particular, the kernel affine projection algorithm (KAPA) [17] has been successfully applied to nonlinear equalization, nonlinear system identification, and nonlinear noise cancellation, as well as to the prediction of nonlinear time series. Although in [24] we have already introduced the use of KAPA for NLAEC, its formal application is lacking so far.

The contribution of this chapter is therefore three-fold: (1) to formally apply KAPA to the NLAEC problem for the first time and confirm its potential for this application, (2) to develop a sliding-window leaky KAPA (SWL-KAPA) that is best suited for NLAEC applications, and (3) to propose a kernel function, that is a weighted sum of a linear and a Gaussian kernel, which is efficient for acoustic applications. The motivation in designing this kernel is basically to separate the problem into linear and nonlinear sub-problems. The weights in the kernel also impose different forgetting mechanisms in the sliding window, which in turn translates to more flexible regularization.

The outline of the chapter is as follows. In Section 8.2, we present a review of the literature, which is instrumental for our purposes, namely, the affine projection algorithm (APA), kernel methods, and the kernel APA (KAPA). In Section 8.3, the SWL-KAPA and the proposed kernel is described with explanation and justification of the choices made for NLAEC. In Section 8.4, computer simulation results are provided to verify the performance of the proposed algorithm, an overview of existing results in the NLAEC literature is given, and a computational complexity analysis of the competing algorithms is shown, in particular, the linear APA and the 3rd-order Hammerstein and Volterra filters. Finally, Section 8.5 concludes the chapter.

8.2 Linear and Kernel APA

8.2.1 Linear APA

Standard approaches to AEC rely on the assumption that the echo path can be modeled by a linear FIR filter [1], [2], [25]- [27], in which the coefficients of the echo path are collected in the parameter vector $\mathbf{h} = [h_0, h_1, ..., h_{L-1}]^T \in \mathbb{R}^L$ with model order $L-1$. An adaptive filter is used to provide an estimate $\hat{\mathbf{h}} \in \mathbb{R}^L$ of $\mathbf{h}$ with sufficient order, such that the echo signal estimate at the microphone position is $\hat{y}(i) = \hat{\mathbf{h}}^T(i)\mathbf{u}(i)$ with $\mathbf{u}(i) = [u(i), u(i-1), \ldots, u(i-L+1)]^T$. Notice that, for now, we have also assumed that the echo path is time-invariant, $\mathbf{h}^T(i) = \mathbf{h}^T, \forall i$ over the estimation time window $[1, 2, \ldots, i]$ with $i \gg L$; we will consider time-varying echo paths further on. A least-squares regression data model may be constructed as

$$\mathbf{d}_{\text{LS}}(i) = \mathbf{U}_{\text{LS}}(i)\mathbf{h} + \mathbf{e}_{\text{LS}}(i) \tag{8.1}$$

where

$$\mathbf{d}_{\text{LS}}(i) = [d(i), d(i-1), ..., d(1)]^T \tag{8.2}$$

$$\mathbf{e}_{\text{LS}}(i) = [e(i), e(i-1), ..., e(1)]^T \tag{8.3}$$

$$\mathbf{U}_{\text{LS}}(i) = [\mathbf{u}(i), \mathbf{u}(i-1), ..., \mathbf{u}(1)]^T. \tag{8.4}$$

The error in (8.1) is minimized in a least-squares sense, giving

$$\hat{\mathbf{h}}(i) = \arg\min_{\hat{\mathbf{h}}(i)} \sum_{j=1}^{i} \left| d(j) - \hat{\mathbf{h}}^T(j)\mathbf{u}(j) \right|^2 \tag{8.5}$$

$$= \left[\mathbf{U}_{\text{LS}}^T(i)\mathbf{U}_{\text{LS}}(i)\right]^{-1} \mathbf{U}_{\text{LS}}^T(i)\mathbf{d}_{\text{LS}}(i) \tag{8.6}$$

where $\mathbf{R}_{uu}(i) = \frac{1}{i}\mathbf{U}_{\text{LS}}^T(i)\mathbf{U}_{\text{LS}}(i)$ and $\mathbf{r}_{du}(i) = \frac{1}{i}\mathbf{U}_{\text{LS}}^T(i)\mathbf{d}_{\text{LS}}(i)$ is the sample autocorrelation matrix and the sample cross-correlation vector, respectively. Nearly all linear adaptive filtering algorithms are related to the LS estimate, which is unique if the autocorrelation matrix is nonsingular. This condition is, however, not always met. Indeed a common problem in AEC applications is that the matrix $\mathbf{R}_{uu}(i)$ is ill-conditioned or even singular due to poor excitation [28]. This situation will generate large variance in the estimation of $\hat{\mathbf{h}}(i)$. Therefore, to improve data generalization (i.e., to prevent small changes in the data from producing large changes in the solution), the l_2-norm of the vector $\hat{\mathbf{h}}(i)$ is often used as a penalty term. This type of regression, commonly known as *ridge regression*, was introduced by Tikhonov and Arsenin [29] (and formulated as a linear algebra problem in [30]) as a remedy for ill-posedness. Essentially, it establishes a trade-off between fitting the data and reducing the norm of the solution, i.e., the smoothness of the solution. Thus, the modified so called

regularized LS criterion is formulated as

$$\hat{\mathbf{h}}(i) = \arg\min_{\hat{\mathbf{h}}(i)} \sum_{j=1}^{i} \left| d(j) - \hat{\mathbf{h}}^T(i)\mathbf{u}(j) \right|^2 + \lambda \left\| \hat{\mathbf{h}}(i) \right\|_2^2 \tag{8.7}$$

$$= \left[\mathbf{U}_{\text{LS}}^T(i)\mathbf{U}_{\text{LS}}(i) + \lambda \mathbf{I}_L \right]^{-1} \mathbf{U}_{\text{LS}}^T(i)\mathbf{d}_{\text{LS}}(i) \tag{8.8}$$

where $\mathbf{I}_L$ is the $L \times L$ identity matrix and λ is a regularization parameter that controls the smoothness of the solution. This regularized LS solution basically turns an ill-posed problem into a well-posed problem by adding a scaled identity matrix to $\mathbf{R}_{uu}(i)$.

Since the echo path may be time-varying due to changes in the room (e.g., people moving around, or changes in microphone or loudspeaker position) and/or the loudspeaker characteristics (e.g., temperature or acoustic load changes), the system must be equipped with a mechanism that adapts to changes in the echo path. Therefore, a recursive least-squares (RLS) algorithm may be derived from the LS estimate of the filter $\hat{\mathbf{h}}(i)$ [26]. Essentially the RLS algorithm updates the LS estimate every time a new pair of data $\{\mathbf{u}(i), d(i)\}$ is available [26]. The RLS algorithm includes an exponential weighting factor δ that "forgets" older data in order to improve its ability for tracking changes in the echo path. The *Tikhonov regularized* RLS, featuring both exponential weighting and regularization, was proposed in [31]. It was called the leaky RLS algorithm because of its similarity with the leaky least mean squares (LMS) algorithm [32], [33].

In general, RLS is computationally too expensive for AEC applications [1]; hence, computationally cheaper algorithms are called for. The most popular and extensively used algorithm is the normalized (NLMS) version of Widrow's LMS algorithm [34], which works with instantaneous estimates of the autocorrelation matrix and hence typically has slow convergence. On the other hand, the so-called affine projection algorithm (APA), originally proposed in [35], computes an estimate of $\mathbf{R}_{uu}(i)$ from the current and $P-1$ past input vectors $\mathbf{u}(j)$. APA obtains faster convergence rates compared to NLMS and better tracking capabilities compared to RLS, so it may be seen as a good compromise between NLMS and RLS in both performance and complexity. APA is, therefore, also adopted in AEC applications when faster convergence is necessary [36], [37].

In [38], APA is derived formally from the underdetermined RLS (URLS) family (whereas RLS is an overdetermined algorithm). The derivation in [38] adopts a deterministic framework, in contrast to the *classical* stochastic derivation [26], [27], [35]. In [28], optimally-regularized APA and NLMS algorithms are derived as part of the URLS family. We use this framework to be consistent with the notation and derivations in this chapter. The APA (and also NLMS)

weight-update equation can be written as [39],

$$\hat{\mathbf{h}}(i) = \hat{\mathbf{h}}(i-1) + \mu \left[\hat{\mathbf{R}}_{uu}(i) + \lambda \mathbf{I}\right]^{-1} \left[\hat{\mathbf{r}}_{du} - \hat{\mathbf{R}}_{uu}(i)\hat{\mathbf{h}}(i-1)\right] \tag{8.9}$$

where μ is a fixed step size that is included as a relaxation mechanism to protect the recursion from divergence due to the use of noisy estimates $\hat{\mathbf{R}}_{uu}(i)$ and $\hat{\mathbf{r}}_{du}(i)$, and λ is a fixed regularization parameter. In contrast to NLMS, which updates the weight vector based only on the current input data $\{\mathbf{u}(i), d(i)\}$ (i.e., $P = 1$), APA updates the weight vector based on $\{\mathbf{u}(i) \dots \mathbf{u}(i-P+1), d(i) \dots d(i-P+1)\}$ (i.e., $P > 1$). Using moving-average estimates

$$\hat{\mathbf{R}}_{uu}(i) = \frac{1}{P}\mathbf{U}^T(i)\mathbf{U}(i), \tag{8.10}$$

and

$$\hat{\mathbf{r}}_{du}(i) = \frac{1}{P}\mathbf{U}^T(i)\mathbf{d}(i) \tag{8.11}$$

where $\mathbf{U}(i) = [\mathbf{u}(i), \mathbf{u}(i-1), ..., \mathbf{u}(i-P+1)]^T \in \mathbb{R}^{P \times L}$, $\mathbf{u}(i) = [u(i), \dots, u(i-L+1)]^T \in \mathbb{R}^L$, and $\mathbf{d}(i) = [d(i), d(i-1), ..., d(i-P+1)]^T \in \mathbb{R}^P$, the weight-update (8.9) becomes

$$\hat{\mathbf{h}}(i) = \hat{\mathbf{h}}(i-1) + \mu \left[\mathbf{U}^T(i)\mathbf{U}(i) + \lambda \mathbf{I}\right]^{-1} \mathbf{U}^T(i) \left[\mathbf{d}(i) - \mathbf{U}(i)\hat{\mathbf{h}}(i-1)\right] \tag{8.12}$$

By using the matrix inversion lemma

$$\left[\mathbf{U}^T(i)\mathbf{U}(i) + \lambda \mathbf{I}_L\right]^{-1} \mathbf{U}^T(i) = \mathbf{U}^T(i) \left[\mathbf{U}(i)\mathbf{U}^T(i) + \lambda \mathbf{I}_P\right]^{-1} \tag{8.13}$$

the basic APA recursion is finally given as

$$\mathbf{e}(i) = \mathbf{d}(i) - \mathbf{U}(i)\hat{\mathbf{h}}(i-1) \tag{8.14}$$

$$\hat{\mathbf{h}}(i) = \hat{\mathbf{h}}(i-1) + \mu \mathbf{U}^T(i) \left[\mathbf{U}(i)\mathbf{U}^T(i) + \lambda \mathbf{I}\right]^{-1} \mathbf{e}(i). \tag{8.15}$$

The *Tikhonov regularized*, or leaky, APA is the underdetermined version of leaky RLS and is written as

$$\mathbf{e}(i) = \mathbf{d}(i) - \mathbf{U}(i)\hat{\mathbf{h}}(i-1) \tag{8.16}$$

$$\hat{\mathbf{h}}(i) = (1 - \lambda\mu)\hat{\mathbf{h}}(i-1) + \mu \mathbf{U}^T(i) \left[\mathbf{U}(i)\mathbf{U}^T(i) + \lambda \mathbf{I}\right]^{-1} \mathbf{e}(i). \tag{8.17}$$

The presence of the leakage term in (8.17) is known to introduce filter coefficient bias, which degrades mean-squared error performance [26], [33]. However, in some applications, its use proves to be beneficial, in particular when using non-stationary, non-white signals like speech [40]- [42]. This is due to the poor excitation properties of this type of signal and the subsequent need for regularization [28], as explained in the introduction. Here, the leaky APA will be the basis for developing a regularized kernel adaptive algorithm that moreover permits the use of a sliding-window, as will be explained in the following sections.

8.2.2 Kernel Methods

A kernel [23] is a continuous, symmetric, and positive-definite function $\kappa : \mathbb{U} \times \mathbb{U} \to \mathbb{R}$ where $\mathbb{U}$ is the input space. A *feature* mapping from the input space to the feature space, $\boldsymbol{\varphi} : \mathbb{U} \to \mathbb{F}$, can be constructed such that for $\mathbf{x}$, $\mathbf{y} \in \mathbb{U}$

$$\kappa(\mathbf{x}, \mathbf{y}) = \boldsymbol{\varphi}^T(\mathbf{x})\boldsymbol{\varphi}(\mathbf{y}), \tag{8.18}$$

where $\mathbb{F}$ is referred to as the feature space and $\boldsymbol{\varphi}(\mathbf{x})$ is referred to as the feature vector. Conveniently, based on the theory of RKHS [23] and making use of the Mercer's condition [43], several algorithms may be constructed that apply the so-called kernel trick based on (8.18), which was originally proposed in [44]. We give an example using a simple kernel for which explicit computations of the mapping $\boldsymbol{\varphi}$ can be shown.

The polynomial kernel is defined as $\kappa(\mathbf{x}, \mathbf{y}) = \left(\mathbf{x}^T\mathbf{y} + c\right)^d$ for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^L$ and $c, d \in \mathbb{R}$. For example, if $d = 2$, $\mathbf{x} = [x_1, \ldots, x_L]^T$ and $\mathbf{y} = [y_1, \ldots, y_L]^T$, then

$$\kappa(\mathbf{x}, \mathbf{y}) = \left(\sum_i x_i y_i + c\right)^2 \tag{8.19}$$

$$= \sum_i \sum_j x_i x_j y_i y_j + \sum_i \left(\sqrt{2c}x_i\right)\left(\sqrt{2c}y_i\right) + c^2. \tag{8.20}$$

From (8.19), we see that $\kappa(\mathbf{x}, \mathbf{y})$ represents the inner product in $\mathbb{F}$, i.e., $\kappa(\mathbf{x}, \mathbf{y}) = \left(\mathbf{x}^T\mathbf{y} + c\right)^2 = \boldsymbol{\varphi}^T(\mathbf{x})\boldsymbol{\varphi}(\mathbf{y})$, implicitly constructed by the mapping $\boldsymbol{\varphi}$ defined as

$$\boldsymbol{\varphi}(\mathbf{x}) = \left[x_1x_1, ..., x_1x_L, ..., x_Lx_L, \sqrt{2c}x_1, ..., \sqrt{2c}x_L, c\right]^T \tag{8.21}$$

$$\boldsymbol{\varphi}(\mathbf{y}) = \left[y_1y_1, ..., y_1y_L, ..., y_Ly_L, \sqrt{2c}y_1, ..., \sqrt{2c}y_L, c\right]^T. \tag{8.22}$$

The feature mapping, in this simple example, takes the form

$$\boldsymbol{\varphi} : \mathbb{R}^L \to \mathbb{R}^{L^2+L+1} \tag{8.23}$$

The kernel trick is applied here to work implicitly in an (L^2+L+1)-dimensional space without having to transform the input data into this space. In fact, the mapping does not necessarily have to be known, since all computations can be done by evaluating the kernel function in the input space. This also holds for any other polynomial order (i.e, $d > 2$) or for any other kernel function, e.g., the Gaussian kernel $\kappa(\mathbf{x}, \mathbf{y}) = \mathrm{e}^{-\rho\|\mathbf{x}-\mathbf{y}\|_2^2}$. By using the Taylor's expansion of $\mathrm{e}^x = \sum_{k=0}^{\infty} \frac{x^k}{k!}$, we see that the Gaussian kernel has an infinite set of polynomial terms, each corresponding to a feature i.e., $\boldsymbol{\varphi} : \mathbb{R}^L \to \mathbb{R}^\infty$. This in particular provides powerful modeling capabilities to kernel adaptive algorithms when dealing with nonlinear mappings; the only requirement for using the kernel trick is that the algorithm in the feature space can be expressed in terms of

inner products, which are then replaced by kernel operations in the input space. Furthermore, it is clear that the estimation complexity will be determined by the input dimension, independent of the order of the nonlinearity [45]. This is one of the most appealing characteristics of kernel methods, as opposed to nonlinear adaptive filters requiring explicit nonlinear expansions of the input data as, for instance, Volterra filters [46].

8.2.3 Leaky APA in the Feature Space

To apply the kernel trick, the input signal vector $\mathbf{u}(i) \in \mathbb{R}^L$ is transformed by means of the feature mapping $\boldsymbol{\varphi}$, into a high-dimensional (possibly infinite-dimensional) feature vector $\boldsymbol{\varphi}[\mathbf{u}(i)] \in \mathbb{R}^F$. The LS problem is defined in the feature space as,

$$\hat{\boldsymbol{\omega}}(i) = \arg\min_{\hat{\boldsymbol{\omega}}(i)} \sum_{j=1}^{i} \left| d(j) - \hat{\boldsymbol{\omega}}^T(i)\boldsymbol{\varphi}[\mathbf{u}(j)] \right|^2. \tag{8.24}$$

Note that $\hat{\boldsymbol{\omega}} \in \mathbb{R}^F$ is a vector of estimated filter coefficients in the feature space $\mathbb{F}$ and therefore has the same dimension as $\boldsymbol{\varphi}[\mathbf{u}(i)]$. In the sequel, the simplified notation

$$\boldsymbol{\varphi}(i) = \boldsymbol{\varphi}[\mathbf{u}(i)] \quad \text{and} \quad \kappa(i,j) = \kappa[\mathbf{u}(i), \mathbf{u}(j)] \tag{8.25}$$

is introduced for compactness. The use of a high-dimensional feature space $\mathbb{F}$ provides kernel methods with a very high degree of flexibility in solving minimization problems [18]. However, this appealing characteristic may cause the solution to perfectly fit any given data set while not generalizing well to new incoming data. This so-called overfitting problem particularly occurs if the Gaussian kernel is used and no precautions are taken. In order to prevent overfitting, the minimization problem should be regularized, which is again achieved by using the l_2-norm of the solution as a penalty term [17], [18], [21]. By introducing a regularization term, the solution is indeed found to generalize better to new data points [22], [47], [48]. The regularized LS problem on the data $\{d(1), d(2), ...\}$ and $\{\boldsymbol{\varphi}(1), \boldsymbol{\varphi}(2), ...\}$ can be formulated in the feature space as

$$\hat{\boldsymbol{\omega}}(i) = \arg\min_{\hat{\boldsymbol{\omega}}(i)} \sum_{j=1}^{i} \left| d(j) - \hat{\boldsymbol{\omega}}^T(i)\boldsymbol{\varphi}(j) \right|^2 + \lambda \left\| \hat{\boldsymbol{\omega}}(i) \right\|_2^2 \tag{8.26}$$

where λ is the regularization parameter. Following the same reasoning as in the previous section, the leaky APA (8.17) can be formulated in the feature space to compute $\hat{\boldsymbol{\omega}}$ using the *primal* model representation as

$$\mathbf{e}(i) = \mathbf{d}(i) - \boldsymbol{\Phi}(i)\hat{\boldsymbol{\omega}}(i-1) \tag{8.27}$$

$$\hat{\boldsymbol{\omega}}(i) = (1 - \lambda\mu)\hat{\boldsymbol{\omega}}(i-1) + \mu\boldsymbol{\Phi}^T(i)\mathbf{G}^{-1}(i)\mathbf{e}(i) \tag{8.28}$$

where $\mathbf{d}(i) = [d(i), d(i-1), \ldots, d(i-P+1)]^T$, $\mathbf{e}(i) = [e_1(i), e_2(i), \ldots, e_P(i)]^T$, $\boldsymbol{\Phi}(i) = [\boldsymbol{\varphi}(i), \boldsymbol{\varphi}(i-1), \ldots, \boldsymbol{\varphi}(i-P+1)]^T$, $\mathbf{G}(i) = \left[\boldsymbol{\Phi}(i)\boldsymbol{\Phi}^T(i) + \lambda\mathbf{I}\right] \in \mathbb{R}^{P\times P}$.

As discussed before, $\boldsymbol{\varphi}^T(i)\hat{\boldsymbol{\omega}}(i) \in \mathbb{R}$ is a much more powerful model than the usual $\mathbf{u}^T(i)\hat{\mathbf{h}}(i) \in \mathbb{R}$ because of the transformation from $\mathbf{u}(i)$ to $\boldsymbol{\varphi}(i)$. However, in practice, $\hat{\boldsymbol{\omega}}(i)$ will never be computed explicitly. Instead, the *dual* model representation will be derived based on the Representer Theorem [49], where the filter coefficient vector $\hat{\boldsymbol{\omega}}(i)$ is expressed as an expansion of all transformed input data up to time i, which then permits application of the kernel trick as shown in the next section.

8.2.4 Leaky KAPA

It is straightforwardly shown, by repeated application of (8.28), that

$$\hat{\boldsymbol{\omega}}(i) = \sum_{j=1}^{i} a_j(i)\boldsymbol{\varphi}(j) \tag{8.29}$$

$$= \sum_{j=1}^{i-1} a_j(i-1)\boldsymbol{\varphi}(j) + \mu \sum_{k=1}^{P} \boldsymbol{\varphi}(i-k+1)\bar{e}_k(i) \tag{8.30}$$

where the expansion coefficients $a_j(i)$ are computed recursively as

$$a_j(i) = \begin{cases} \mu\bar{e}_{i-j+1}(i), & j = i \\ (1-\lambda\mu)a_j(i-1) + \mu\bar{e}_{i-j+1}(i), & j = i-1, ..., i-P+1 \\ (1-\lambda\mu)a_j(i-1), & j > P \end{cases} \tag{8.31}$$

and where $\bar{e}_{i-j+1}(i)$ is the prediction error normalized by the $P \times P$ Gram matrix $\mathbf{G}(i)$ as,

$$\bar{e}_{i-j+1}(i) = \sum_{k=1}^{P} G^{-1}_{i-j+1,k}(i)e_k(i) \tag{8.32}$$

with $G^{-1}_{j,k}(i)$ defined as the matrix element lying on the jth row and kth column of the inverse Gram matrix at time i, and with

$$e_k(i) = d(i-k+1) - \boldsymbol{\varphi}^T(i-k+1)\boldsymbol{\omega}(i-1). \tag{8.33}$$

The leaky KAPA [17], whose derivation is extended in Appendix 8.A, will include the expansion coefficients updating (8.31), but will never construct $\hat{\boldsymbol{\omega}}(i)$ explicitly. However, although (8.29) is never explicitly computed, it allows one to rearrange (8.27) and (8.28) based on the kernel trick, leading to the leaky KAPA. First, the output vector $\hat{\mathbf{y}}(i) = \boldsymbol{\Phi}(i)\hat{\boldsymbol{\omega}}(i-1)$ in (8.27) can be computed in terms of kernel evaluations as

$$\hat{\mathbf{y}}(i) = \boldsymbol{\Phi}(i) \sum_{j=1}^{i-1} a_j(i-1)\boldsymbol{\varphi}(j) \tag{8.34}$$

$$= \sum_{j=1}^{i-1} a_j(i-1)\,[\kappa(i,j),\ \kappa(i-1,j),...,\kappa(i-P+1,j)\ ]^T\,. \tag{8.35}$$

Next, the error vector in (8.27), i.e., with elements given in (8.33), can be directly computed as

$$e_k(i) = d(i-k+1) - \sum_{j=1}^{i-1} a_j(i-1)\kappa(i-k+1,j). \tag{8.36}$$

The matrix $\mathbf{G}(i)$ can be computed based on

$$\mathbf{\Phi}(i)\mathbf{\Phi}^T(i) = [\boldsymbol{\varphi}(i),...,\boldsymbol{\varphi}(i-P+1)]^T[\boldsymbol{\varphi}(i),...,\boldsymbol{\varphi}(i-P+1)] \tag{8.37}$$

$$= \begin{bmatrix} \kappa(i,i) & \kappa(i,i-1) & \kappa(i,i-P+1) \\ \kappa(i-i,i) & \kappa(i-1,i-1) & \kappa(i-1,i-P+1) \\ \vdots & \ddots & \vdots \\ \kappa(i-P+1,i) & \dots & \kappa(i-P+1,i-P+1) \end{bmatrix}. \tag{8.38}$$

In [17] and [19], it has been shown that, since KAPA is just APA in RKHS, the important properties of APA (convergence in mean, misadjustment, etc.) also hold for KAPA.

8.3 Sliding-Window Leaky KAPA for Nonlinear AEC

8.3.1 Pruning by Use of a Sliding Window

From a practical implementation point of view, leaky KAPA has to have two memory buffers, one for the set of input vectors (usually called the *dictionary*), $\mathbf{U}_{\mathrm{D}}(i) = [\mathbf{u}(1),...,\mathbf{u}(i)]$, needed to evaluate the kernel functions in (8.36) and (8.38), and another for the corresponding vector of expansion coefficients $\mathbf{a}(i)$ in (8.31). The equations defining leaky KAPA (8.31)-(8.38) start from an empty set $\mathbf{U}_{\mathrm{D}}(0)$ and gradually add new input vectors $\mathbf{u}(i)$ (usually called the *centers*) to this set. The size of the dictionary and the corresponding expansion coefficient vector grow linearly with time i. This results in an increasing memory and computational complexity requirement as time evolves. In the literature, there exist several strategies to cope with this problem, based on essentially two criteria: a sparse dictionary criterion and a *pruning* criterion.

The first criterion is to optimally construct a sparse dictionary, i.e., whenever new input data is considered, a decision is made whether or not to add the new center $\mathbf{u}(i)$ to the dictionary. The well known novelty criterion for on-line sparsification, which was originally proposed for resource-allocation networks

(RAN) in [50], is based on the Euclidean distance between the new center and the dictionary [17]. Approximate linear dependency (ALD) [51] is another criterion, which constructs the dictionary based on the linear dependency of the new input data and the dictionary. In [52], the concept of *surprise* has been proposed, which measures the information a new center can contribute to the learning system, based on Gaussian process theory. Also, in [53], sparseness has been achieved in an off-line configuration based on the solution vector of a least-squares support vector machine (LS-SVM). Coherence-based sparsification has been proposed in [54] and employed in [55] using multikernels. Also in [55], the so-called block soft-thresholding is employed in on-line sparsification. However, these strategies are not readily applicable to our problem; either because they are restricted to off-line applications or because, as in on-line sparsification, the sparse memory buffers are in fact not bounded and therefore still grow without memory limitation. This is specially so, if using non-stationary data [55].

A number of pruning criteria have also been developed that discard either redundant or least-significant information from the memory buffers. In [56], [57] a number of different criteria are presented to prune LS-SVMs. In [58] a criterion presented in [57] has been used to prune the kernel autocorrelation matrix in Kernel RLS (KRLS) based on selecting the data that incurs the least error after it is omitted. The calculation of the error and further data selection appears to be computationally expensive. A simpler strategy has been adopted for KRLS in [59] where a sliding window considers only the last pairs of data to form the kernel autocorrelation matrix. The problem with these strategies, in relation to KAPA, is that they involve operations on the kernel autocorrelation matrix, which carries much more information than our $P \times P$ matrix $\mathbf{G}(i)$. The use of these pruning strategies would therefore result in significant errors and poor performance when applied to KAPA.

On the other hand, in the NORMA algorithm [21], which is equivalent to a kernel version of leaky LMS, the problem is solved by pruning the kernel expansion coefficients. The idea is that, since at each time instant i the expansion coefficients are scaled by the leakage term $(1 - \lambda\mu)$, which is less than 1, the oldest term can be dropped without incurring significant error. This pruning criterion essentially converts NORMA into a sliding-window kernel LMS (SW-KLMS) algorithm. Following the spirit of NORMA, the leaky KAPA is converted here into a sliding-window leaky KAPA (SWL-KAPA). Thus, SWL-KAPA exploits the benefits of the leaky KAPA both in terms of regularization and in that the pruning criterion can be used. The implementation of the leaky KAPA used here differs from that of [17] in that the error signal in NLAEC applications is indeed computed explicitly, as this is the signal sent back to the far-end side.

8.3.2 Weighted Sum of Kernels

The choice of the kernel implicitly defines a RKHS with an appropriate inner product [19], [60]. Furthermore, the choice of the kernel also defines the type of nonlinearity that underlies the model used. The choice of the kernel is therefore vital in the development of different algorithms, and the rationale behind the choice may be multiple. For instance, one of the most commonly used kernels is the Gaussian kernel since its performance is superior to that of other kernels, like polynomial kernels. However in [45], polynomial kernels are the preferred choice since the obtained solutions can be directly transformed into their corresponding Wiener or Volterra representation. In NLAEC applications, the enclosure impulse response is usually of very high order, e.g., hundreds of taps in mobile communication systems and even thousands of taps in room acoustic applications. This in particular makes the single application of most of the kernels, e.g., Gaussian kernels, inefficient for real-time applications.

Kernels verifying Mercer's condition [48], which guarantees the existence of a Hilbert space and an inner product associated with the kernel considered, are of particular interest in on-line learning. Essentially, convergence of the training algorithm to a unique optimum is ensured. This type of kernel can be used to construct other families of kernels that also meet these conditions [22], [47]. Several techniques have been applied to finding the best-suited kernel for a given application. Most of the techniques are based on convex optimization of the kernel designing problem, so they are only suitable for off-line applications, e.g., [61]. On the other hand, in [55], a multikernel approach has been proposed to choose the best kernel parameters in an adaptive fashion. However, only one type of kernel (i.e., Gaussian), with different parameters, is used.

Here, a kernel that consists of a weighted sum of linear and Gaussian kernels is proposed for the NLAEC applications. The motivation for this kernel is to separate the problem into linear and nonlinear subproblems. The proposed kernel is defined as

$$\kappa_{\Sigma}(i,j) = \alpha\kappa_{\mathrm{L}}(i,j) + \beta\kappa_{\mathrm{G}}(i,j) \tag{8.39}$$

$$= \alpha\mathbf{u}^{T}(i)\mathbf{u}(j) + \beta\exp(-\rho\left\|\mathbf{u}(i)-\mathbf{u}(j)\right\|_{2}^{2}) \tag{8.40}$$

where $0 \leq \alpha \leq 1$, $\beta = (1-\alpha)$, and ρ is the Gaussian kernel parameter controlling its bandwidth. The kernel κ_{Σ} is positive definite and thus a reproducing kernel in RKHS, which immediately follows from the property that if $\alpha, \beta \geq 0$, then $\kappa_{\Sigma} = \alpha\kappa_{\mathrm{L}} + \beta\kappa_{\mathrm{G}}$ is positive definite provided that $\kappa_{\mathrm{L}}, \kappa_{\mathrm{G}}$ are two arbitrary positive definite kernels [62]. Moreover, this kernel elegantly fits into the SWL-KAPA since the parameters α and β give yet another degree of flexibility in the regularization strategy by expanding the regularization term in (8.26)

on the last D observations as

$$\lambda \|\hat{\boldsymbol{\omega}}\|^2 = \lambda \hat{\boldsymbol{\omega}}^T \hat{\boldsymbol{\omega}} = \lambda \left(\sum_{i=1}^{D} a_i(D) \boldsymbol{\varphi}\left[\mathbf{u}(i)\right] \right)^T \left(\sum_{j=1}^{D} a_j(D) \boldsymbol{\varphi}\left[\mathbf{u}(j)\right] \right) \tag{8.41}$$

$$= \lambda \sum_{i=1}^{D} \sum_{j=1}^{D} a_i a_j \boldsymbol{\varphi}(i)^T \boldsymbol{\varphi}(j) = \lambda \sum_{i=1}^{D} \sum_{j=1}^{D} a_i a_j \kappa_\Sigma(i,j) \tag{8.42}$$

$$= \lambda \sum_{i=1}^{D} \sum_{j=1}^{D} a_i a_j \left[\alpha \kappa_{\mathrm{L}}(i,j) + \beta \kappa_{\mathrm{G}}(i,j)\right] \tag{8.43}$$

$$= \lambda_{\mathrm{L}} \sum_{i=1}^{D} \sum_{j=1}^{D} a_i a_j \kappa_{\mathrm{L}}(i,j) + \lambda_{\mathrm{G}} \sum_{i=1}^{D} \sum_{j=1}^{D} a_i a_j \kappa_{\mathrm{G}}(i,j). \tag{8.44}$$

The regularization can now be more favored in one kernel than in the other by varying the weights $\lambda_{\mathrm{L}} = \lambda\alpha$ and $\lambda_{\mathrm{G}} = \lambda\beta$.

8.3.3 Sliding-Window Leaky KAPA

A complete SWL-KAPA description is provided in Algorithm 16. The following variables are adopted: P is the APA projection order, D is the length of the memory buffers, i.e., the sliding-window length, μ is the step size, λ is the regularization parameter, $\mathbf{u}(i) = [u(i), u(i-1), ..., u(i-L+1)]^T$ is the $L \times 1$ input (far-end) signal vector, $d(i)$ is the desired (microphone) signal, $\hat{\mathbf{y}}(i) = [\hat{y}_1(i), ..., \hat{y}_P(i)]^T$ is a $P \times 1$ output vector, $\mathbf{a}(i)$ is the $D \times 1$ expansion coefficients vector, the dictionary $\mathbf{U}_{\mathrm{D}}$ is a $L \times D$ matrix, $e_{\mathrm{AEC}}(i)$ is the NLAEC error (residual) signal that is sent to the far-end side, $\mathbf{z}$ is a $P \times 1$ auxiliary vector, $\mathbf{G}(i) = [\boldsymbol{\Phi}(i)\boldsymbol{\Phi}^T(i) + \lambda\mathbf{I}]$ is the matrix defined in (8.38), and $\mathbf{e}(i) = [e_1(i), ..., e_P(i)]^T$ is the $P \times 1$ error vector. Note that although the kernel function was previously defined for only two vectors, the kernel κ_Σ in Algorithm 16, line 4, takes inputs comprising an $L \times 1$ vector $\mathbf{u}(i)$, and an $L \times D$ matrix $\mathbf{U}_{\mathrm{D}}(i-1)$. The outcome of this operation is simply a $D \times 1$ vector whose elements are the kernel evaluations of the current input vector with the D previous input vectors that constitute the dictionary.

8.4 Evaluation

8.4.1 Simulated Nonlinear System

The simulated nonlinear system is a Hammerstein-like system implemented using a 3rd-order Volterra nonlinearity (instead of a memoryless nonlinearity

Algorithm 5 Sliding-Window Leaky KAPA

initialization D, P, μ, λ, $a_1(1) = \mu e_1(1) = \mu d(1)$, $\mathbf{z} = \mathbf{0}$, $\mathbf{U}_{\mathrm{D}}(1) = \mathbf{u}(1)$
while $\{\mathbf{u}(i), d(i)\}$ available **do**
  for $i = 2 \rightarrow end$ **do**
    for $k = 1 \rightarrow P$ **do**
      $\hat{y}_k(i) = \mathbf{a}^T(i-1)\boldsymbol{\kappa}_{\Sigma}\left[\mathbf{u}(i-k+1), \mathbf{U}_{\mathrm{D}}(i-1)\right]$
    end for
    $e_{\mathrm{AEC}}(i) = e_1(i)$
    $\mathbf{z} = \begin{bmatrix} \mathbf{a}_{end-P+2:end}(i-1) \\ 0 \end{bmatrix} + \mu \mathbf{G}^{-1}(i)\mathbf{e}(i)$
    $\mathbf{a}(i) = \begin{bmatrix} (1-\lambda\mu)\left[\mathbf{a}(i-1) + \mathbf{z}_{1:P-1}\right] \\ z_P \end{bmatrix}$
    $\mathbf{U}_{\mathrm{D}}(i) = [\mathbf{U}_{\mathrm{D}}(i-1) \quad \mathbf{u}(i)]$
    if length$\mathbf{a}(i) > D$ **then**
      $a_1(i) \leftarrow \emptyset$ (`Delete first element`)
      $\mathbf{U}_{\mathrm{D},1}(i) \leftarrow \emptyset$ (`Delete first column vector`)
    end if
  end for
end while

as commonly comprises a Hammerstein system):

$$y_{\mathrm{lin}}(i) = \sum_{j=0}^{L} h_j u(i-j) \tag{8.45}$$

$$y_{\mathrm{volt}}(i) = \sum_{j=0}^{L_{\mathrm{v}}} \sum_{k=j}^{L_{\mathrm{v}}} h_{2,(j,k)} u(i-j)u(i-k) \tag{8.46}$$

$$+ \sum_{j=0}^{L_{\mathrm{v}}} \sum_{k=j}^{L_{\mathrm{v}}} \sum_{l=k}^{L_{\mathrm{v}}} h_{3,(j,k,l)} u(i-j)u(i-k)u(i-l) \tag{8.47}$$

$$y_{\mathrm{nl}}(i) = \sum_{j=0}^{L} h_j y_{\mathrm{volt}}(i-j) \tag{8.48}$$

$$d(i) = y_{\mathrm{lin}}(i) + \sigma\, y_{\mathrm{nl}}(i) + n(i) = f\left[u(i)\right] + n(i) \tag{8.49}$$

where $u(i)$ is the far-end signal, $d(i)$ is the microphone signal, $n(i)$ is the near-end noise signal, $y_{\mathrm{lin}}(i)$ is the linear echo, $y_{\mathrm{nl}}(i)$ is the nonlinear echo, $y_{\mathrm{volt}}(i)$ is the output of a 3rd-order Volterra filter, $\mathbf{h} = [h_0, \ldots, h_{L-1}]^T$ is a 400-taps measured acoustic room impulse response (RIR), and σ controls the linear-to-nonlinear ratio (LNLR), i.e., $\mathrm{LNLR} = 10\log_{10}\dfrac{\sum_{j=1}^{N} y_{\mathrm{lin}}^2(j)}{\sum_{j=1}^{N} y_{\mathrm{nl}}^2(j)}$ with N the total length of the signals. The order (i.e., memory) of the linear adaptive filter is $L-1 = 399$, while the memory for the Volterra kernels is $L_{\mathrm{v}} - 1 = 79$.

Kernel in this case refers to the nested summation of each nonlinearity. The simulated nonlinear system has a physical meaning since the nonlinearities are concentrated locally on the amplifier and loudspeaker, which are then filtered by the RIR. The coefficients in the 3rd-order Volterra kernels (i.e., h_2 and h_3) were obtained from an i.i.d. Gaussian random process. This choice makes the Volterra kernels fully populated which may generalize well for any type of loudspeaker [3], [63], [64]. Moreover, the 3rd-order nonlinearity demonstrates the validity of the SWL-KAPA using the kernel κ_Σ in dealing with high-order nonlinearities, without having to specify the order of the nonlinearity. By contrast, in adaptive Volterra filters, the order of the nonlinearity has to be explicitly set in advance and besides has a direct impact on the computational complexity.

8.4.2 Competing Filters: Hammerstein and Volterra

The two nonlinear adaptive filters that will be compared to SWL-KAPA are defined by the input-output relationship

$$y_{\mathrm{F}}(i) = \mathbf{h}_{\mathrm{F}}^T \mathbf{u}_{\mathrm{F}}(i) \tag{8.50}$$

where $\mathbf{h}_{\mathrm{F}}$ is a vector containing L_{F} filter coefficients and $\mathbf{u}_{\mathrm{F}}(i)$ is the corresponding vector whose L_{F} entries are formed with a nonlinear expansion of the input samples in $\mathbf{u}(i)$. This expression can be interpreted as an FIR filter with L_{F} coefficients in which the entries of $\mathbf{u}_{\mathrm{F}}(i)$ are used in place of the samples $\mathbf{u}(i)$. If $\mathbf{u}_{\mathrm{F}}(i)$ is chosen equal to $\mathbf{u}(i)$, this corresponds to a linear FIR filter with $L_{\mathrm{F}} = L$ coefficients. It will be convenient to subdivide $\mathbf{u}_{\mathrm{F}}(i)$ and $\mathbf{h}_{\mathrm{F}}(i)$ into appropriate sub-vectors as

$$\mathbf{u}_{\mathrm{F}}(i) = \left[\mathbf{u}^T(i),\ \mathbf{u}_2^T(i),\ \mathbf{u}_3^T(i)\right]^T, \tag{8.51}$$

$$\mathbf{h}_{\mathrm{F}}(i) = \left[\mathbf{h}^T(i),\ \mathbf{h}_2^T(i),\ \mathbf{h}_3^T(i)\right]^T, \tag{8.52}$$

so that

$$y_{\mathrm{F}}(i) = \mathbf{h}^T\mathbf{u}(i) + \mathbf{h}_2^T\mathbf{u}_2(i) + \mathbf{h}_3^T\mathbf{u}_3(i). \tag{8.53}$$

It is clear that the linear APA recursion in (8.14) and (8.15) can be applied directly using (8.50) and (8.51), when the dimensions are adjusted accordingly.

- Hammerstein filter

Due to its intrinsic simplicity, the Hammerstein filter has been studied and used for many applications including NLAEC [10]. The Hammerstein filter is formed with a memoryless nonlinearity followed by a linear filter. Since the nonlinearity is usually expressed as a polynomial function and the linear filter has a finite impulse response of L samples, it is considered as a simple member of the class of Volterra filters. The sub-vectors of $\mathbf{u}_{\mathrm{F}}(i)$ in (8.51) are defined by

$$\mathbf{u}_2(i) = [u^2(i), u^2(i-1),\ \cdots, u^2(i-L+1)]^T, \tag{8.54}$$

$$\mathbf{u}_3(i) = [u^3(i), u^3(i-1), \cdots, u^3(i-L+1)]^T. \tag{8.55}$$

The total number of filter coefficients is therefore $L_{\mathrm{H}} = 3L$.

- Volterra filter

Volterra filters are characterized by input-output relationships that result from two truncations of the discrete Volterra series [46], namely a memory truncation of the Volterra kernel orders ($L_2 - 1$ and $L_3 - 1$ in this case), and a nonlinearity order truncation (3 in this case) to limit the number of Volterra kernels. The Volterra kernels can be assumed to be symmetric and thus can be recast in the more compact triangular form

$$\mathbf{u}_2(i) = [u^2(i), u(i)u(i-1), ..., u^2(i-1), \tag{8.56}$$

$$u(i-1)u(i-2), ..., u^2(i-L_2+1)]^T \tag{8.57}$$

with dimension $L_2(L_2+1)/2 \times 1$ and

$$\mathbf{u}_3(i) = [u^3(i), u(i)u(i)u(i-1), ..., u(i)u(i-1)u(i-2), \tag{8.58}$$

$$..., u^3(i-L_3+1)]^T \tag{8.59}$$

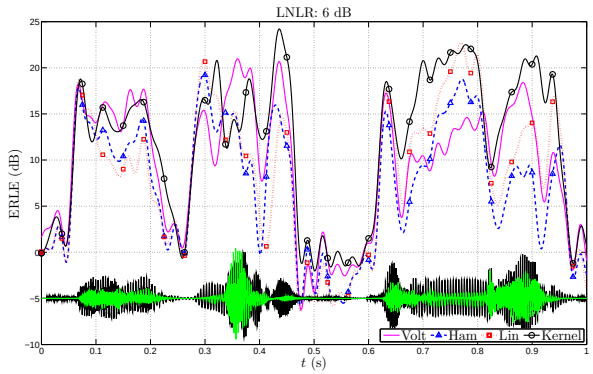
with dimension $L_3(L_3+1)(L_3+2)/6 \times 1$. Thus, the total number of filter coefficients is $L_{\mathrm{V}} = L + L_2(L_2+1)/2 + L_3(L_3+1)(L_3+2)/6$

8.4.3 Simulation Results

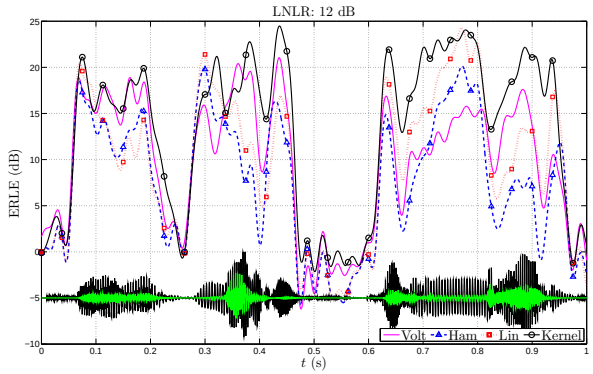
The performance measure is the *echo return loss enhancement* (ERLE), which is given as

$$\mathrm{ERLE}(m) = 10\log_{10}\frac{\sum_{j=1}^{q} d^2[(m-1)q+j]}{\sum_{j=1}^{q} e^2[(m-1)q+j]} \quad m = 1, ... \tag{8.60}$$

and which can be seen as the achieved echo attenuation averaged over time frames of length q. Simulations are performed using speech signals (female speech sampled at 8 kHz) and i.i.d. background noise $n(i)$ with a SNR of 20 dB. Results are provided for two LNLR values 6 dB and 12 dB. Some parameters in every algorithm are the same, in order to make the comparison as fair as possible: $\mu = 1$, $P = 24$, $\rho = 1$, $D = 1000$, $\lambda = 10^{-2}$. The input dimension of the Gaussian and linear kernel is $L = 400$ and the linear APA filter order is $L - 1 = 399$. The linear (dark-solid) and nonlinear (light-solid) part of the echo are shown in the bottom of each figure as a reference for the instantaneous LNLR. In each scenario, the ERLE of four algorithms is compared over time. The algorithms are the SWL-KAPA, the 3rd-order Volterra and Hammerstein filters updated using APA, and the linear APA. The 3rd-order Volterra filter is computed using lower projection and memory orders, i.e., $P = 3$, $L_2 - 1 = 349$



(a)

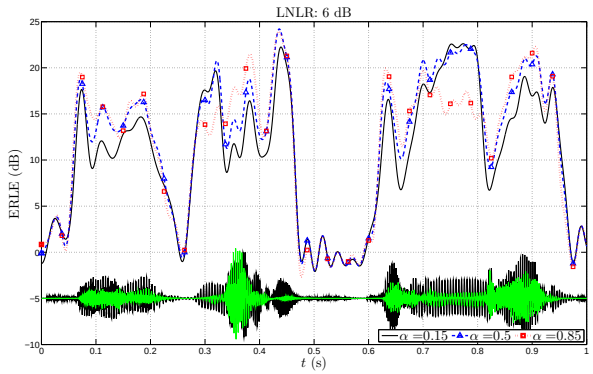


(b)

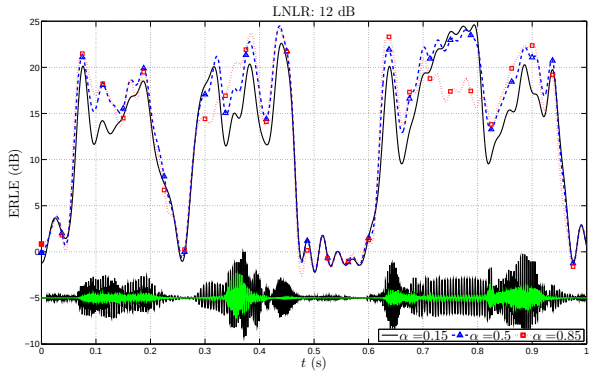
Figure 8.2: ERLE comparison of four algorithms: Volterra (solid), Hammerstein (triangle-dashed), linear APA (square-dotted) and SWL-KAPA (circle-solid). (a) 6-dB LNLR. (b) 12-dB LNLR.

and $L_3 - 1 = 349$. The reason for this adjustment in the Volterra filter is simply that it is too expensive to compute the 3rd-order Volterra using the same APA updating as in the other three algorithms; even a low APA projection order, $P = 3$, results in excessive memory usage and computation times.

Figure 8.2 shows the ERLE of the four algorithms at different LNLR. As expected, in the presence of non-negligible nonlinear echo, the linear APA provides the worst performance, followed by the Hammerstein and Volterra filters. The Volterra filter either obtains the same results as the Hammerstein filter or it performs $5 - 6$ dB better. These differences become more clear in the case with 12 dB LNLR. Interestingly, SWL-KAPA generally outperforms any other



(a)



(b)

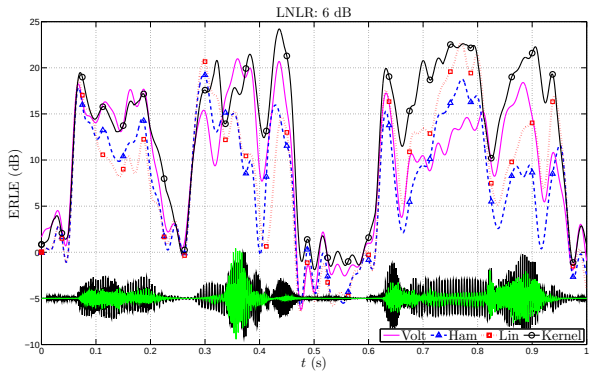
Figure 8.3: ERLE comparison of SWL-KAPA using different kernel weights: $\alpha = 0.15$ (solid), $\alpha = 0.5$ (triangle-dashed) and $\alpha = 0.85$ (square-dotted). (a) 6-dB LNLR. (b) 12-dB LNLR.

algorithm for both 6 and 12-dB LNLR, obtaining an almost constant improvement over the other three algorithms. The improvement with respect to the linear APA is at some time instants 10 to 11 dB in the 12-dB LNLR case. In this comparison, the kernel weights are equal, $\alpha = \beta = 0.5$, and thus the freedom to choose different values is not exploited. To this end, Figure 8.3 shows the ERLE obtained using SWL-KAPA for three different pairs of values for the kernel weights, $\{\alpha, \beta\} = \{0.15, 0.85\}, \{0.5, 0.5\}, \{0.85, 0.15\}$. It is clear that large ERLE improvements can be achieved by choosing different kernel weight values at different times, essentially depending on the instantaneous LNLR.

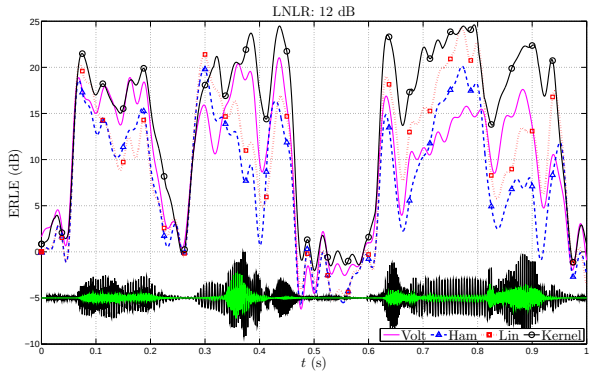
Figure 8.4 is similar to Figure 8.2, but in this case the weights are variables,

choosing the best option among the three at each time instant. It is clear that, again, SWL-KAPA generally outperforms any other algorithm, now with a general ERLE improvement with respect to Figure 8.2. Particularly, around 0.35 s, SWL-KAPA goes from being outperformed by the Volterra filter, in Figure 8.2, to reaching the same ERLE in Figure 8.4. These observations show that a dynamic choice of the kernel weights can further improve the performance of SWL-KAPA. The proposed SWL-KAPA consistently outperforms the linear APA, resulting in 12-dB maximum improvement in ERLE at a computational cost that is only 4.6 times higher. Moreover, it is shown that the SWL-KAPA outperforms, by 4 to 6 dB, the Volterra-based NLAEC, which itself is 413 times more expensive than the linear APA. Moreover, the Volterra filter performance during high instantaneous LNLR (e.g., between 0.5 s and 0.8 s in both cases) decreases considerably with respect to low LNLR, which is a known fact [16]; the linear APA performance obviously increases during instantaneous high LNLR. In contrast, SWL-KAPA maintains its superior performance constantly in both cases, during high or low instantaneous LNLR. This fact indeed constitutes another appealing characteristic of SWL-KAPA.

Although, comparing different methods appears to be cumbersome due to the fact that no simulation using a standard set of signals, echo paths, and, most importantly, nonlinear echo models, is found, we will now relate our results to those given in the NLAEC literature. In [11], a 2nd-order Volterra filter has been found to result in 4 to 5 dB improvement when using Gaussian noise as an input signal. In [12], a comparison has been presented among several nonlinear cascaded filters, namely Hammerstein, Wiener, Wiener-Hammerstein, and Bilinear. In a simulated echo path with Gaussian noise as input signal, up to 8.5 dB of improvement has been achieved. Compared to those results, SWL-KAPA achieves better ERLE performance, even with speech as the input signal. For speech input signals, up to 8 dB of ERLE improvement has been reported in [7] by using a memoryless (i.e., Hammerstein-like) preprocessor, however, this improvement has only been observed at specific time instants and in the almost noise-free case. In [10] an orthogonalized version of the polynomial Hammerstein filter (called "power filter") has been suggested to improve convergence. An improvement of 5 to 6 dB has been reported compared with a linear filter. SWL-KAPA still outperforms these latter examples even when they show a simpler model than a 3rd-order Volterra, and in the first case an almost noise-free environment. In [8], an improvement of 5 dB has been achieved using 2nd-order Volterra filters. In [13], a comparison has been presented among a linear filter, a time-domain Volterra filter, and the so-called multidelay filter (MDF), which applies partitioned block methods for fast convolution in the DFT domain. Results have shown around 5 dB of average ERLE improvement and up to 8 dB at specific times using the MDF as compared with the linear filter. Even though SWL-KAPA outperforms these computationally cheaper 2nd-order Volterra filters, by 4 dB in the latter case, it is necessary to stress the fact that SWL-KAPA can easily deal with 3rd-order



(a)



(b)

Figure 8.4: ERLE comparison of four algorithms: Volterra (solid), Hammerstein (triangle-dashed), linear APA (square-dotted) and SWL-KAPA (circle-solid). (a) Best weights $(0.15 - 0.5 - 0.85)$ with 6-dB LNLR. (b) Best weights $(0.15 - 0.5 - 0.85)$ with 12-dB LNLR.

Volterra nonlinearities. The only 3rd-order Volterra-based NLAEC is found in [15], where two models have been proposed, showing improvements of 5 and 2 dB, respectively. It is clear that SWL-KAPA outperforms these latter results and moreover at a much cheaper computational complexity. In [16], an interesting approach has been proposed with a convex combination of filters. The combination of linear and nonlinear filters achieves the best possible results, depending on the LNLR. Results have been given for a simulated 2nd-order Volterra nonlinear path with speech as input signal and in the noise-free case. By varying the ratio between the linear and nonlinear contribution in the echo path, the performance of their proposed algorithm (which is a convex combi-

nation of a linear and a 2nd-order Volterra filter) and the standard adaptive 2nd-order Volterra filter is compared. Improvements of 4 dB and 13 dB are reported for medium and high LNLR compared to the linear filter. This is the only reported algorithm that achieves better performance, however it should be stressed that in [16] the explicit calculation of the 2nd-order Volterra kernel is necessary, hence increasing its computational complexity. Moreover, a simpler model for the nonlinear echo (i.e., a smaller quadratic Volterra kernel, without filtering by the RIR) is considered in a noise-free case.

8.4.4 Computational Complexity

The computational complexity, i.e., the number of operations per sample update, of the linear APA is $(P^2 + 2P)L + P^3 + P^2$. SWL-KAPA has the same matrix inversion with $\mathbf{G}(i)$, but on the other hand does not have the explicit calculations with the filter coefficient vector. Instead SWL-KAPA calculates the error signal based on kernel evaluations. The linear kernel evaluation requires L operations and the Gaussian kernel requires roughly $2L$ operations plus the exponential calculation. These operations are repeated D times per sample update. To calculate the echo estimate, SWL-KAPA dot-multiplies the kernel output with the vector of expansion coefficients for each output sample, summing up to PD operations altogether. Overall, the SWL-KAPA computational complexity is $(L + 2L)D + PD + P^3$.

We now calculate the resulting filter vector lengths to obtain the overall computational complexity when using an APA update. The filter coefficient vector for linear APA and the input dimension for SWL-KAPA is $L = 400$. The Hammerstein filter vector has $L_{\mathrm{H}} = 3L = 1200$ coefficients and the Volterra filter vector has $L_{\mathrm{V}} = L + L_2(L_2 + 1)/2 + L_3(L_3 + 1)(L_3 + 2)/6 = 7269025$ coefficients, where the memory of the Volterra kernels is $L_2 - 1 = L_3 - 1 = 349$. The projection order is $P = 24$ except for the Volterra filter, which has $P_{\mathrm{V}} = 3$. Table 8.1 shows the overall complexity of the four algorithms. SWL-KAPA is only 4.6 times more computationally expensive than the linear APA and 1.6 times that of the 3rd-order Hammerstein filter. On the other hand, the 3rd-order Volterra with $P_{\mathrm{V}} = 3$ is 88 times more expensive than the SWL-KAPA and 413 times more than the linear APA.

Table 8.1: Computational complexity $L = 400$, $L_{\mathrm{H}} = 1200$ $P = 24$, $L_{\mathrm{V}} = 7269025$, $P_{\mathrm{V}} = 3$ and $D = 1000$.

APA update	Computation	Operations
Linear	$(P^2 + 2P)L + P^3 + P^2$	264000
Hammerstein 3rd	$(P^2 + 2P)L_{\mathrm{H}} + P^3 + P^2$	763200
Volterra 3rd	$(P_{\mathrm{V}}^2 + 2P_{\mathrm{V}})L_{\mathrm{V}} + P_{\mathrm{V}}^3 + P_{\mathrm{V}}^2$	109035411
SWL-KAPA	$(L + 2L)D + PD + P^3$	1237824

One can argue that the employed Volterra filter is merely the direct symmetric implementation [46], which indeed is not as cheap as the simplified Volterra filters of [8]. One can also argue that better convergence rates can be obtained using orthogonalized versions [10] or frequency-domain versions [14]. However, the complexity of the SWL-KAPA can also be reduced by implementing a sliding Gram matrix or error reuse [17]. So the aim here is not to make an exhaustive comparison of the different methods but to demonstrate the potential of an alternative approach to NLAEC based on kernel adaptive methods. Much more work has to be done in this context, particularly in view of complexity reduction, optimization of the sliding window with respect to the leakage term, the use of other kernels, the "kernelization" of other adaptive filters used in AEC, the dynamic selection of the kernel weights, etc.

8.5 Conclusions

The sliding-window leaky KAPA (SWL-KAPA) has been introduced and its validity and potential in tackling the NLAEC problem has been demonstrated. SWL-KAPA uses a kernel function that consists of a weighted sum of linear and Gaussian kernels. The motivation for using this kernel is to separate the problem into linear and nonlinear subproblems, where weighting the kernels also imposes different forgetting mechanisms in the sliding window, which in turn translates to more flexible regularization. The result of this is that large performance improvements can be achieved by choosing different kernel weight values at different times, depending on the instantaneous LNLR. The combination of SWL-KAPA with the proposed kernel function results in excellent ERLE performance compared to that of a linear APA, 3rd-order Hammerstein, and Volterra filters. In our experiment set-up, the proposed SWL-KAPA consistently outperforms the linear APA, resulting in 12-dB maximum ERLE improvement at a computational cost that is only 4.6 times higher. Moreover, it is shown that the SWL-KAPA outperforms, by 4 to 6 dB, the Volterra-based NLAEC, which itself is 413 times more expensive than the linear APA.

Appendix

8.A Derivation of leaky KAPA

The derivation starts from (8.28) by repeated application of the weight-update equation, but first it is worth noting that the error $\mathbf{e}(i) \in \mathbb{R}^P$ is

$$\begin{bmatrix} e_1(i) \\ \vdots \\ e_P(i) \end{bmatrix} = \begin{bmatrix} d(i) \\ \vdots \\ d(i-P+1) \end{bmatrix} - \begin{bmatrix} \boldsymbol{\varphi}^T(i) \\ \vdots \\ \boldsymbol{\varphi}^T(i-P+1) \end{bmatrix} \boldsymbol{\omega}(i-1)$$
$$[P \times 1] = [P \times 1] - [P \times F][F \times 1]$$

or in a more compact way:

$$e_k(i) = d(i-k+1) - \boldsymbol{\varphi}^T(i-k+1)\boldsymbol{\omega}(i-1), \quad k = 1, ..., P \tag{8.61}$$

Now let us derive step by step the expansion of $\boldsymbol{\omega}$ assuming $P > 2$ of (8.28). For the sake of compactness, the normalization matrix $\mathbf{G}(i)^{-1} \in \mathbb{R}^{P \times P}$ will be dropped out in the derivation by considering a normalized error

$$\bar{\mathbf{e}}(i) = \mathbf{G}(i)^{-1}\mathbf{e}(i) \in \mathbb{R}^P \tag{8.62}$$

to relax the notation. It will be included later on, when the kernel trick is applied to the whole recursion.

$$\begin{aligned}
\boldsymbol{\omega}(0) &= \mathbf{0} \\
\boldsymbol{\omega}(1) &= (1-\lambda\mu)\boldsymbol{\omega}(0) + \mu\boldsymbol{\Phi}^T(1)\mathbf{G}^{-1}(1)\mathbf{e}(1) = \mu\boldsymbol{\varphi}(1)\bar{e}_1(1) \\
\boldsymbol{\omega}(2) &= (1-\lambda\mu)\boldsymbol{\omega}(1) + \mu\boldsymbol{\Phi}^T(2)\mathbf{G}^{-1}(2)\mathbf{e}(2) \\
&= (1-\lambda\mu)\boldsymbol{\varphi}(1)\bar{e}_1(1) + \mu\boldsymbol{\varphi}(2)\bar{e}_1(2) + \mu\boldsymbol{\varphi}(1)\bar{e}_2(2) \\
&\vdots \\
\boldsymbol{\omega}(P) &= (1-\lambda\mu)\boldsymbol{\omega}(P-1) + \mu\sum_{k=1}^{P}\boldsymbol{\varphi}(P-k+1)\bar{e}_k(P) \\
&\vdots \\
\boldsymbol{\omega}(i) &= (1-\lambda\mu)\boldsymbol{\omega}(i-1) + \mu\sum_{k=1}^{P}\boldsymbol{\varphi}(i-k+1)\bar{e}_k(i)
\end{aligned} \tag{8.63}$$

Let us look at the summation term in the right-hand side of the weight-update equation (8.63). At different iterations it spans as

$$
\begin{aligned}
& i \Rightarrow \boldsymbol{\varphi}(i)\bar{e}_1(i) + \boldsymbol{\varphi}(i-1)\bar{e}_2(i) + ... + \boldsymbol{\varphi}(i-P+1)\bar{e}_P(i) \\
& i-1 \Rightarrow \boldsymbol{\varphi}(i-1)\bar{e}_1(i-1) + \boldsymbol{\varphi}(i-2)\bar{e}_2(i-1) + ... \\
& \qquad\qquad\qquad + \boldsymbol{\varphi}(i-P)\bar{e}_P(i-1) \\
& \qquad \vdots \\
& i-P+1 \Rightarrow \boldsymbol{\varphi}(i-P+1)\bar{e}_1(i-P+1) + ... \\
& \qquad + \boldsymbol{\varphi}(i-2P+1)\bar{e}_P(i-P+1) \\
& i-P \Rightarrow \boldsymbol{\varphi}(i-P)\bar{e}_1(i-P) + ... + \boldsymbol{\varphi}(i-2P)\bar{e}_P(i-P) \\
& \qquad \vdots
\end{aligned}
$$

If we look at it from the point of view of the basis defined by the transformed input vectors $\boldsymbol{\varphi}(\cdot)$, we may group together the ones with the same iteration index between brackets. This is done by grouping terms across different iteration index. This makes sens because the expression for $\boldsymbol{\omega}(i)$ also contains the previous $\boldsymbol{\omega}(i-1)$, $\boldsymbol{\omega}(i-2), ...$ scaled by a factor $(1-\lambda\mu)$, $(1-\lambda\mu)^2, ...$

$$
\begin{aligned}
& i \Rightarrow \boldsymbol{\varphi}(i)[\bar{e}_1(i)] \\
& i-1 \Rightarrow \boldsymbol{\varphi}(i-1)[\bar{e}_1(i-1) + \bar{e}_2(i)] \\
& \qquad \vdots \\
& i-P+1 \Rightarrow \boldsymbol{\varphi}(i-P+1)[\bar{e}_1(i-P+1) + \bar{e}_2(i-P) + ... + \bar{e}_P(i)] \\
& i-P \Rightarrow \boldsymbol{\varphi}(i-P)[\bar{e}_1(i-P) + \bar{e}_2(i-P-1) + ... + \bar{e}_P(i-1)] \\
& \qquad \vdots
\end{aligned}
\tag{8.64}
$$

Now it has the look of what we actually want, namely the dual model representation, in which the weight vector is given as a linear combination of transformed input vectors expanded by some coefficients. Let us now define the vector of expansion coefficients $\bar{\mathbf{a}}$ at iteration i i.e., $\bar{\mathbf{a}}(i) = \bar{a}_j(i),\ j = 1, ..., i,$ based on the elements of (8.64) as

$$\bar{\mathbf{a}}(i) = \begin{bmatrix} \bar{e}_1(i) \\ \bar{e}_1(i-1) + \bar{e}_2(i) \\ \vdots \\ \bar{e}_1(i-P+1) + \bar{e}_2(i-P+2) + ... + \bar{e}_P(i) \\ \bar{e}_1(i-P) + \bar{e}_2(i-P+1) + ... + \bar{e}_P(i-1) \\ \vdots \end{bmatrix} \tag{8.65}$$

and similarly for $\bar{\mathbf{a}}(i-1) = \bar{a}_j(i-1),\ j = 1, ..., i-1,$

$$\bar{\mathbf{a}}(i-1) = \begin{bmatrix} \bar{e}_1(i-1) \\ \bar{e}_1(i-2) + \bar{e}_2(i-1) \\ \vdots \\ \bar{e}_1(i-P) + \bar{e}_2(i-P+1) + ... + \bar{e}_P(i-1) \\ \bar{e}_1(i-P-1) + \bar{e}_2(i-P) + ... + \bar{e}_P(i-2) \\ \vdots \end{bmatrix} \tag{8.66}$$

This shows that at iteration i the expansion coefficient vector $\bar{\mathbf{a}}(i)$ is formed with the errors computed at time i. This is done by storing a new (current) error sample $\bar{e}_1(i)$ and updating $P-1$ previous coefficients by summing $\bar{e}_k(i), \quad k = 2, .., P$, i.e., $[\bar{e}_2(i), ..., \bar{e}_P(i)]$. All the other expansion coefficients remain unchanged as seen from the last line of (8.65) on. Now, (8.63) can be expressed by (8.64) and (8.65) as

$$\boldsymbol{\omega}(i) = \sum_{j=1}^{i} (1-\lambda\mu)^{i-j} \bar{a}_j(i) \boldsymbol{\varphi}(j) \tag{8.67}$$

$$= \sum_{j=1}^{i-1} (1-\lambda\mu)^{i-j} \bar{a}_j(i-1) \boldsymbol{\varphi}(j) + \mu \sum_{k=1}^{P} \boldsymbol{\varphi}(i-k+1) \bar{e}_k(i). \tag{8.68}$$

This shows that the weight vector at time $i-1$ is a linear combination of all previous transformed input vectors with the expansion coefficients in $\bar{\mathbf{a}}(i-1)$, defined in (8.66), weighted by increasing powers of the leakage term $(1-\lambda\mu)$. Denoting $\mathbf{a}(i)$ by its elements $a_j(i) = (1-\lambda\mu)^{i-j} \bar{a}_j(i) \quad j = 1, ..., i$. It follows that (8.67) and (8.68) become

$$\boldsymbol{\omega}(i) = \sum_{j=1}^{i} a_j(i) \boldsymbol{\varphi}(j) \tag{8.69}$$

$$= \sum_{j=1}^{i-1} a_j(i-1) \boldsymbol{\varphi}(j) + \mu \sum_{k=1}^{P} \boldsymbol{\varphi}(i-k+1) \bar{e}_k(i) \tag{8.70}$$

Finally, by (8.65) and (8.70) the updating of the weight vector reduces to the updating on the expansion coefficients $\mathbf{a}(i)$ as

$$\mathbf{a}(i) = \begin{cases} \mu\bar{e}_k(i) & \text{if } k = 1 \\ (1-\lambda\mu)\mathbf{a}(i-1) + \mu\bar{e}_k(i) & \text{if } k = 2, ..., P \\ (1-\lambda\mu)\mathbf{a}(i-1), & \text{if } k > P \end{cases} \tag{8.71}$$

where $\bar{e}_k(i)$, as defined in (8.62), is the prediction error normalized by the $P \times P$ Gram matrix $\mathbf{G}(i)$. It is now when the kernel trick can be exploited: Given $\boldsymbol{\omega}(i-1)$ and the transformed input matrix $\boldsymbol{\Phi}(i)$ in the primal representation, the output vector $\hat{\mathbf{y}}(i) = \boldsymbol{\Phi}(i)\hat{\boldsymbol{\omega}}(i-1)$ in (8.27) can be computed, using the dual model representation, in terms of kernel evaluations as

$$\begin{aligned} \hat{\mathbf{y}}(i) = \boldsymbol{\Phi}(i)\boldsymbol{\omega}(i-1) &= \boldsymbol{\Phi}(i)\sum_{j=1}^{i-1} a_j(i-1)\boldsymbol{\varphi}(j) \\ &= \sum_{j=1}^{i-1} a_j(i-1)\left[\boldsymbol{\Phi}(i)\boldsymbol{\varphi}(j)\right] \\ &= \sum_{j=1}^{i-1} a_j(i-1)\left[\kappa(i,j),\ \kappa(i-1,j), ..., \kappa(i-P+1,j)\ \right]^T \end{aligned} \tag{8.72}$$

Consequently, the error signal in (8.61) can be computed by (8.72) with $k = 1, ..., P$, now in terms of kernel evaluations only as

$$\begin{aligned} \mathbf{e}(i) &= \mathbf{d}(i) - \hat{\mathbf{y}}(i) \\ e_k(i) &= d(i-k+1) - \boldsymbol{\varphi}^T(i-k+1)\boldsymbol{\omega}(i-1) \\ &= d(i-k+1) - \hat{y}_k(i) \\ &= d(i-k+1) - \sum_{j=1}^{i-1} a_j(i-1)\kappa(i-k+1,j) \end{aligned} \tag{8.73}$$

$$\tag{8.74}$$

Bibliography

[1] J. Benesty, T. Gänsler, D. R. Morgan, M. M. Sondhi, and S. L. Gay, *Advances in Network and Acoustic Echo Cancellation*. Berlin: Springer-Verlag, 2001.

[2] P. Gil-Cacho, T. van Waterschoot, M. Moonen, and S. H. Jensen, "Multi-microphone acoustic echo cancellation using multi-channel warped linear prediction of common acoustical poles," in *Proc. 18th European Signal Process. Conf.*, (EUSIPCO'10), Aalborg, Denmark, Aug. 2010, pp. 2121–2125.

[3] ——, "Study and characterization of odd and even nonlinearities in electrodynamic loudspeakers by means of periodic random-phase multisine," in *Preprints AES 127th convention*, Oct. 2009, Preprint 7841.

[4] A. N. Birkett and R. A. Goubran, "Nonlinear echo cancellation for hands-free telephony using neural networks," in *Proc. 1994 IEEE Workshop Neural Netw. Signal Process.*, (NNSP'94), Ermioni , Greece, Sept. 1994, vol. 1, pp. 249 – 258.

[5] A. B. Rabaa and R. Tourki, "Nonlinear echo cancellation for hands-free telephony," in *Proc. 10th European Signal Process. Conf.*, (EUSIPCO'00), Tampere, Finland, Sept. 2000, pp. 1875–1878.

[6] B. S. Nollett and D. L. Jones, "Nonlinear echo cancellation for hands-free speakerphones," in *Proc. 1997 IEEE Workshop Nonlinear Signal and Image Process.*, (NSIP'97), London, UK, May 1997, vol. 1, Paper number 521.

[7] A. Stenger and W. Kellermann, "Adaptation of a memoryless preprocessor for nonlinear acoustic echo cancelling," *IEEE Trans. Signal Process.*, vol. 80, no. 9, pp. 1747–1760, Sept. 2000.

[8] A. Fermo, A. Carini, and G. L. Sicuranza, "Simplified Volterra filters for acoustic echo cancellation in GSM receivers," in *Proc. 10th European Signal Process. Conf.*, (EUSIPCO'00), Tampere, Finland, Sept. 2000, pp. 2413–2416.

[9] G. L. Sicuranza, A. Carini, and A. Fermo, "Nonlinear adaptive filters for acoustic echo cancellation in mobile terminals," in *Nonlinear Signal and Image Processing: Theory, Methods, and Applications*, ser. The Electrical Engineering and Applied Signal Processing Series, K. E. Barner and G. R. Arce, Eds. Florida: CRC Press LLC, 2004, ch. 7, pp. 223–255.

[10] F. Kuech and W. Kellerman, "Orthogonalized power filters for nonlinear acoustic echo cancellation," *Signal Process.*, vol. 86, pp. 1168–1181, Nov. 2006.

[11] A. Stenger and R. Rabenstein, "Adaptive Volterra filters for acoustic echo cancellation," in *Proc. 1999 IEEE-EURASIP Workshop Nonlinear Signal and Image Process.*, (NSIP'99), Antalya, Turkey, June 1999.

[12] J. P. Costa, A. Lagrange, and A. Arliaud, "Acoustic echo cancellation using nonlinear cascade filters," in *Proc. 2003 IEEE Int. Conf. Acoust. Speech Signal Process.*, (ICASSP'03), Hong Kong, China, Apr. 2003, pp. 389–392.

[13] F. Kuech and W. Kellerman, "A novel multidelay adaptive algorithm for Volterra filters in diagonal coordinates," in *Proc. 2004 IEEE Int. Conf. Acoust. Speech Signal Process.*, (ICASSP'04), Montreal, Quebec, Canada, May 2004, pp. 869–872.

[14] F. Küch, "Adaptive polynomial filters and their application to nonlinear acoustic echo cancellation," Ph.D. dissertation, Friedrich–Alexander–Universität Erlangen–Nürnberg, 2005.

[15] A. Guérin, G. Faucon, and R. L. Bouquin-Jeannés, "Nonlinear acoustic echo cancellation based on Volterra filters," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 6, pp. 672–683, Nov. 2003.

[16] L. A. Azpicueta-Ruiz, M. Zeller, J. Arenas-García, and W. Kellermann, "Novel schemes for nonlinear acoustic echo cancellation based on filter combinations," in *Proc. 2010 IEEE Int. Conf. Acoust. Speech Signal Process.*, (ICASSP'10), Taipei, Taiwan, Apr. 2010, pp. 193–196.

[17] W. Liu, J. C. Príncipe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction*. Hoboken, NY: Wiley, 2010.

[18] S. V. Vaerenbergh, "Kernel methods for nonlinear identification, equalization and separation of signals," Ph.D. dissertation, Universidad de Cantabria, 2010.

[19] P. Bouboulis and S. Theodoridis, "The complex Gaussian kernel LMS algortihm," in *Proc. 20th Int. Conf. Artificial Neural Netw.*, (ICANN'10), Thessaloniki, Greece, Aug. 2010, pp. 11–20.

[20] S. Theodoridis, K. Slavakis, and I. Yamada, "Adaptive learning in a world of projections: a unifying framework for linear and nonlinear classification and regression tasks," *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 97–123, Jan. 2011.

[21] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 165–176, Aug. 2004.

[22] B. Schölkopf and A. J. Smola, *Learning with kernels*. Cambridge, MA: MIT Press, 2002.

[23] N. Aronszajn, "Theory of reproducing kernels," *Trans. Amer. Math. Soc.*, vol. 68, pp. 337–404, Jan. 1950.

[24] J. M. Gil-Cacho, T. van Waterschoot, M. Moonen, and S. H. Jensen, "Nonlinear acoustic echo cancellation based on a parallel-cascade kernel affine projection algorithm," in *Proc. 2012 IEEE Int. Conf. Acoust. Speech Signal Process.*, (ICASSP'12), Kyoto, Japan, Mar. 2012, pp. 33–36.

[25] T. van Waterschoot, G. Rombouts, P. Verhoeve, and M. Moonen, "Double-talk-robust prediction error identification algorithms for acoustic echo cancellation," *IEEE Trans. Signal Process.*, vol. 55, no. 3, pp. 846–858, Mar. 2007.

[26] S. Haykin, *Adaptive Filter Theory*. Upper Saddle River, NY: Prentice Hall, 2002.

[27] P. S. R. Diniz, *Adaptive filtering: Algorithms and Practical Implementations*. Boston, MA: Springer, 2008.

[28] T. van Waterschoot, G. Rombouts, and M. Moonen, "Optimally regularized adaptive filtering algorithms for room acoustic signal enhancement," *Signal Process.*, vol. 88, no. 3, pp. 594–611, Mar. 2008.

[29] A. Tikhonov and V. Arsenin, *Solutions of Ill-Posed Problems*. New York: Wiley, 1977.

[30] A. Neumaier, "Solving ill-conditioned and singular linear systems: A tutorial on regularization," *SIAM Rev.*, vol. 40, no. 3, pp. 636–666, 1998.

[31] E. Horita, K. Sumiya, H. Urakami, and S. Mitsuishi, "A leaky RLS algorithm: its optimality and implementation," *IEEE Trans. Signal Process.*, vol. 52, no. 10, pp. 2924–2936, Oct. 2004.

[32] R. D. Gitlin, H. C. Meadors, and S. B. Weinstein, "The tap-leakage algorithm: An algorithm for the stable operation of a digitally-implemented fractionally-spaced equalizer," *Bell Sys. Tech. J.*, vol. 61, no. 10, pp. 1817–9840, Oct. 1982.

[33] K. Mayyas and T. Aboulnasr, "Leaky LMS algorithm: MSE analysis for Gaussian data," *IEEE Trans. Signal Process.*, vol. 45, no. 4, pp. 927–934, Apr. 1997.

[34] B. Widrow and M. Hoff, "Adaptive switching circuits," in *Proc. WESCON Conv. Rec.*, vol. 4, 1960, pp. 96–140.

[35] K. Ozeki and T. Umeda, "An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties," *Electronics and Communication in Japan*, vol. 67, no. 5, pp. 19-27, Aug. 1984.

[36] S. S. Oh, M. Ali, and D. Linebarger, "Adaptive filtering method and apparatus employing modified fast affine projection algorithm," US Patent No: 6 137 881, Oct. 24, 2000.

[37] G. Q. Jin, "Modified affine projection algorithm for non-stationary signal," US Patent No: 7 280 654, Oct. 09, 2007.

[38] B. Baykal and A. Constantinides, "Underdetermined-order recursive least-squares adaptive filtering: the concept and algorithms," *IEEE Trans. Signal Process.*, vol. 45, no. 2, pp. 346-362, Feb. 2008.

[39] D. R. Morgan and S. G. Kratzer, "On a class of computationally efficient, rapidly converging, generalized NLMS algorithms," *IEEE Signal Process. Lett.*, vol. 3, pp. 245 – 247, Aug. 1996.

[40] P. Gil-Cacho, T. van Waterschoot, M. Moonen, and S. H. Jensen, "Regularized adaptive notch filters for acoustic howling suppression," in *Proc. 17th European Signal Process. Conf.*, (EUSIPCO'09), Glasgow, Scotland, UK, Aug. 2009, pp. 2574–2578.

[41] B. D. Rigling, "Subspace leaky LMS," *IEEE Signal Process. Lett.*, vol. 11, no. 2, pp. 136 – 139, Feb. 2004.

[42] G. B. Chandra and A. Mitra, "A fast adaptive echo canceler with leaky proportionate NLMS algorithm," in *Proc. IET-UK Int. Conf. Inf. Commun. Technol. Elect. Sc.*, (ICTES'07), Tamil Nadu, India, Dec. 2007, pp. 611–15.

[43] J. Mercer, "Functions of positive and negative type and their connection with the theory of integral equations," *Philos. T. Roy. Soc. A*, vol. 209, pp. 415-446, 1909.

[44] M. Aizerman, E. Braverman, and L. Rozonoer, "Theoretical foundations of the potential function method in pattern recognition learning," *Automat. Rem. Cont.*, vol. 25, pp. 821-837, 1964.

[45] M. O. Franz and B. Schölkopf, "A unifying view of Wiener and Volterra theory and polynomial kernel regression," *Neural Computation*, vol. 18, no. 12, pp. 3097–3118, 2006.

[46] V. J. Mathews and G. L. Sicuranza, *Polynomial Signal Processing.* New York: Wiley, 2000.

[47] C. E. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning.* Cambridge, MA: MIT Press, 2006.

[48] V. N. Vapnik, *The Nature of Statistical Learning Theory.* New York: Springer-Verlag, 1995.

[49] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *Proc. 14th Ann. Conf. Comput. Learning Theory*, (COLT'01), Amsterdam, The Netherlands, 2001, pp. 416–426.

[50] J. Platt, "A resource-allocating network for function interpolation," *Neural Computation*, vol. 3, no. 2, pp. 213–225, June 1991.

[51] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 275–285, Aug. 2004.

[52] W. Liu, I. Park, and J. C. Príncipe, "An information theoretic approach of designing sparse kernel adaptive filters," *IEEE Trans. Neural Netw.*, vol. 20, no. 12, pp. 950–961, Dec. 2004.

[53] J. A. K. Suykens, L. Lukas, and J. Vandewalle, "Sparse approximation using least squares support vector machines," in *Proc. 2000 IEEE Int. Symp. Circuits Syst.*, (ISCAS'00), Geneva, Switzerland, May 2000, vol. 2, pp. 757–760.

[54] C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Trans. Signal Process.*, vol. 57, no. 3, pp. 1058 – 1067, Mar. 2009.

[55] M. Yukawa, "Multikernel adaptive filtering," *IEEE Trans. Signal Process.*, vol. 60, no. 9, pp. 4672 – 4682, Sept. 2012.

[56] L. Hoegaerts, J. A. K. Suykens, J. Vandewalle, and B. D. Moor, "A comparison of pruning algorithms for sparse least squares support vector machines," *Lect. Notes Comp. Sc.*, vol. 1, pp. 247–253, 2004.

[57] B. J. de Kruif and T. J. A. de Vries, "Pruning error minimization in least squares support vector machines," *IEEE Trans. Neural Netw.*, vol. 14, no. 3, pp. 696–702, May 2003.

[58] S. V. Vaerenbergh, I. Santamaría, W. Liu, and J. C. Príncipe, "Fixed-budget kernel recursive least-squares," in *Proc. 2010 IEEE Int. Conf. Acoust. Speech Signal Process.*, (ICASSP'10), Dallas, TX, USA, Mar. 2010, pp. 882–885.

[59] S. V. Vaerenbergh, J. Vía, and I. Santamaría, "Nonlinear system identification using a new sliding-window kernel RLS algorithm," *J. Commun.*, vol. 2 no. 3, pp. 1–8, May 2007.

[60] G. Wahba, *Spline models for observational data*. Philadelphia, PA: SIAM, 1990.

[61] G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semi-definite programming," *J. Mach. Learn. Res.*, vol. 5, pp. 27-72, Sept. 2004.

[62] C. Berg, J. Peter, R. Christensen, and P. Ressel, *Harmonic Analysis on Semigroups*. New York: Springer-Verlag, 1984.

[63] A. J. M. Kaizer, "Modeling of the nonlinear response of an electrodynamic loudspeaker by a Volterra series expansion," *J. Audio Eng. Soc.*, vol. 35, no. 6, pp. 421-432, June 1987.

[64] M. J. Reed and M. O. J. Hawksford, "Identification of discrete Volterra series using maximum length sequences," *IEE Proc.-Circuits Devices Syst.*, vol. 143, no. 5, pp. 241 – 248, Oct. 1996.

Part V

Conclusions

Chapter 9

Conclusions and Suggestions for Future Research

9.1 Conclusions

In this thesis, we have proposed a number of novel room acoustic signal enhancement algorithms. These algorithms have been designed and evaluated for acoustic echo cancellation and acoustic feedback control applications. These two applications share many common challenges, namely, complexity reduction, robustness to near-end signals, robustness to nonlinearities in the audio chain and algorithmic efficiency. These issues have been tackled, first by understanding the systems under evaluation in the view of room acoustics modeling and nonlinear systems, second by improving existing techniques with significant computational reduction.

The first part of the thesis has focused on room acoustics modeling for room transfer function (RTF) equalization. The equalization performance then depends on the model from which an inverse filter is derived. The utilization of different norms (i.e., 2-norm and 1-norm) and models (i.e., all-pole and pole-zero) for RTF modeling and equalization has been studied. The most common model for the RTF is an *all-zero* model but the equalization filter is then a single-point inverse filter. One of the alternatives has been to use a *pole-zero* model for the RTF. Poles can represent long impulse responses caused by resonances, while zeros represent time delays and anti-resonances. Yet another alternative model has been the *all-pole* model, which represents only the acous-

tic resonances in an effort to model the RTF spectral envelope. The concept of common acoustical poles has been applied to these two alternative models. Equalization has then been achieved by canceling the poles associated with the main resonances common to multiple RTFs in a room. Therefore, the poles have been estimated by means of three different methods. Poles estimated using 2-norm minimization and an all-pole model have offered a residual RTF having the flattest response. Poles estimated using 2-norm minimization and a pole-zero model have offered a large reduction in the main low-frequency acoustic resonances. However the residual RTF have exhibited a highly colored response, while the residual RIR was shorter in time. Finally, poles estimated using 1-norm minimization and an all-pole model have offered a flat residual RTF with its main resonances canceled and a sparse residual RIR. This means that the residual RIR represents sparsely distributed discrete reflections.

We have applied this type of equalization in a prefiltering stage for multi-microphone acoustic echo cancellation (AEC), employing the idea of common-acoustical-pole and zero RTF modeling. Moreover, we have proposed the warped linear prediction of the impulse responses to calculate the poles. By frequency-warping the measured impulse responses, we have been able to focus the computational resources to the low frequency region where the predominant spectral peaks are located. The use of frequency warping has led to a higher echo reduction for the same number of filter coefficients. Moreover, these predominant spectral peaks are common to every RTF which has allowed us to have a fixed common denominator polynomial for every channel. Hence only the numerator polynomial has had to be estimated recursively to adapt to changes in the room. In a first configuration, the prefiltering has been done on the adaptive filter signal, hence achieving a pole-zero model of the RTF in the AEC. In a second configuration, the prefiltering has been done on the loudspeaker signal, hence achieving a dereverberation effect, in addition to AEC, on the microphone signals. Compared with all-zero modeling, this approach has allowed the number of numerator coefficients to be decreased by up to one order of magnitude for each microphone, still obtaining satisfactory results (about 22 dB of echo attenuation). We have obtained better results in the case of prefiltering in the adaptive filter signal path. This is due to the IIR nature of this configuration which resulted in a better modeling capability. i The second part of the thesis has mainly focused on linear methods for signal enhancement. The algorithms treated in this part have been designed and evaluated to tackle the issue of robustness to double-talk (DT) in AEC and robustness to model mismatch in acoustic feedback cancellation (AFC). Moreover, from an acoustic feedback control point of view, an algorithm for acoustic howling suppression based on adaptive notch filters with regularization has been proposed.

We have proposed a new framework to tackle the problem of DT in AEC which is based on a frequency-domain adaptive filtering (FDAF) implementation of the so-called PEM-AFROW algorithm (FDAF-PEM-AFROW). It

has been shown that the FDAF-PEM-AFROW minimizes the BLUE criterion which can be considered an optimal acoustic echo path estimate during DT. We have departed from the FDAF-PEM-AFROW algorithm to show an improvement in performance with respect to two other adaptive algorithms minimizing a BLUE criterion, namely, the PEM-AFROW and the FDAF-NLMS with near-end signal normalization. Although the three algorithm are constructed from the BLUE framework and should therefore obtain the minimum variance echo path estimate during DT, we have shown that the differences between them are clear. The FDAF-NLMS with near-end signal normalization have performed better than PEM-AFROW as the conditions to achieve the BLUE seem to be less restricting. FDAF-NLMS with near-end signal normalization has been clearly outperformed by FDAF-PEM-AFROW which has the benefits of being implemented in the frequency-domain, featuring a decorrelation prefilter and including the non-frequency-dependent near-end signal variance estimate. We, moreover, have proposed the instantaneous pseudo-correlation (IPC) measure between the near-end speech signal and the loudspeaker signal. The IPC measure has been shown to be a clear indication of the effect of a DT situation occurring during adaptation. We have shown that the IPC measure is significantly reduced using the combination of FDAF and PEM, as in FDAF-PEM-AFROW. We, moreover, have used the FDAF-PEM-AFROW framework to improve several state-of-the-art variable step-size (VSS) and variable regularization (VR) algorithms. In particular, the affine projection algorithm (APA)-based projection-correlation VSS (PCVSS-APA), the practical VSS-APA (PVSS-APA) and the APA-based VR (VR-APA). It has been showed that the FDAF-PEM-AFROW versions significantly outperform the original versions in every simulation by at least 6 dB during DT. In terms of computational complexity the FDAF-PEM-AFROW versions are themselves 10 times cheaper than the original versions.

Based on the FDAF-PEM-AFROW framework, we have derived a practical yet highly robust algorithm for DT-robust AEC and AFC. The proposed algorithm contains two modifications to the FDAF-PEM-AFROW algorithm, namely (a) the WIener variable Step sizE (WISE) and (b) the GRAdient Spectral variance Smoothing (GRASS), leading to the WISE-GRASS-FDAF-PEM-AFROW algorithm. The WISE has been implemented as a single-channel noise reduction Wiener filter applied to the microphone signal after the decorrelation prefilter, where the Wiener filter gain is used as a VSS in the adaptive filter. On the other hand, the GRASS aims at reducing the variance in the noisy gradient estimates and is based on time-recursive averaging of gradient estimates. WISE-GRASS-FDAF-PEM-AFROW has been shown to obtain improved robustness and smooth adaptation in highly adverse scenarios such as in bursting DT at high levels, and with a change of the acoustic path during continuous DT. Simulations have shown that WISE-GRASS-FDAF-PEM-AFROW outperforms other competing algorithms in adverse scenarios both in AEC and AFC applications when the near-end signals (i.e., speech and/or back-

ground noise) strongly affect the microphone signal. WISE-GRASS-FDAF-PEM-AFROW combines the best characteristics of robust adaptation both in continuous DT and bursting DT without the need for a DTD. WISE-GRASS-FDAF-PEM-AFROW consequently gathers all the characteristic we have been seeking for namely, decorrelation properties (PEM, FDAF), minimum variance (GRASS, FDAF, PEM), variable step size (WISE), and computational efficiency (FDAF).

From an acoustic feedback control point of view, a new method for notch-filter-based howling suppression (NHS) has been proposed. NHS methods typically perform a frequency analysis, howling detection and howling suppression. We have chosen a method based on adaptive notch filters (ANF) over those based on fast Fourier transform (FFT). Adaptive notch filters (ANFs) perform parametric frequency estimation, and allow for a simultaneous howling estimation/detection and suppression. We have shown the advantage of ANF-based methods over FFT-based methods: 1) the required processing delay is minimal since the ANF-based method works on a sample-by-sample basis, 2) ANFs are able to track changes in the howling frequency on a sample-by-sample basis, 3) avoiding FFT operations strongly reduces computational complexity, and 4) the achievable frequency estimation accuracy is generally high for a limited amount of data. Howling detection is, however, difficult in ANF-based methods since no power spectrum information is available. We have therefore proposed a novel and accurate howling detection method to overcome the limitations of ANF-based NHS. The proposed method has been based on adaptive notch filters (ANF) that include a regularization term (RANF). Simulations have shown that the method is able to suppress and track howling frequencies in situations where the howling frequency is confined to mid-range frequencies.

Finally, the third part of the thesis has dealt with the problem of nonlinearities in the acoustic path. We, firstly, have performed a thorough analysis of three common types of active loudspeakers found in practical applications where AEC and AFC are required. Moreover we have proposed a novel sliding-window leaky kernel affine projection algorithm for nonlinear acoustic echo cancellation (NLAEC).

We have observed that the level of nonlinearities in all measured (active) loudspeakers is significant. The level of the nonlinearities depends on the quality of the (active) loudspeaker, although eventually any (active) loudspeaker suffers from nonlinear distortion. Therefore, we have considered the characterization of the nonlinear response, in terms of odd and even nonlinearities, in active loudspeakers by means of periodic random-phase multisine signals. First of all, we have seen that the nonlinear odd contributions are much more predominant than the even ones. This means that *at least* a 3rd-order nonlinear filter has to be considered. We have shown, however, that a 3rd-order nonlinear filter is not enough to capture all the nonlinearities, meaning that odd and even nonlinear contributions are produced by higher-order nonlinearities. More-

over, we have considered the identification of a loudspeaker model by means of linear-in-the-parameters nonlinear adaptive filters. We have compared nonlinear filters without memory (Hammerstein and Legendre polynomial filters) and with memory (simplified 3rd-order Volterra filter). The results have shown that Legendre polynomial filters improve performance monotonically with increasing order. In the simplified 3rd-order Volterra filter case, the performance is remarkably poorer than in the Legendre polynomial filter case for the same filter length. The differences between identification and validation appear to be very small in the Legendre polynomial filter case. Moreover, these differences tend to decrease with increasing order. In the simplified 3rd-order Volterra filter case, the differences between identification and validation in the even harmonics case are much higher than in the Legendre polynomial filter case. Moreover, these differences tend to increase with increasing order which is a sign of overfitting. In Hammerstein filter identification after order 7, increasing the filter order does not improve performance. Therefore, misleading results about considering nonlinearities with or without cross terms may occur if only a Hammerstein filter or a 3rd-order Volterra filter is considered for identification. We have shown that in our set-up the use of a high-order filter is much more important than the memory of a Volterra filter. If the nonlinear system appears to have some memory it is clear that the use of high-order adaptive Volterra filters becomes prohibitive in AEC applications.

Our main contribution to NLAEC is to propose a sliding-window leaky kernel affine projection algorithm (SWL-KAPA). The SWL-KAPA has been introduced and its validity and potential in tackling the NLAEC problem has been demonstrated. SWL-KAPA uses a kernel function that consists of a weighted sum of linear and Gaussian kernels. The motivation for using this kernel is to separate the problem into linear and nonlinear subproblems, where weighting the kernels also imposes different forgetting mechanisms in the sliding window, which in turn translates to more flexible regularization. The result of this is that large performance improvements can be achieved by choosing different kernel weight values at different time instants, depending on the instantaneous linear-to-nonlinear ratio. The combination of SWL-KAPA with the proposed kernel function results in excellent echo return loss enhancement performance compared to that of a linear APA, 3rd-order Hammerstein, and Volterra filters. In our experiment set-up, the proposed SWL-KAPA consistently outperforms the linear APA, resulting in 12-dB maximum ERLE improvement at a computational cost that is only 4.6 times higher. Moreover, it is shown that the SWL-KAPA outperforms, by 4 to 6 dB, the Volterra-based NLAEC, which itself is 413 times more expensive than the linear APA.

To conclude, throughout this thesis several novel algorithms have been proposed for acoustic echo cancellation and acoustic feedback control. We have successfully approached some of the challenges typically found in practical applications, namely, complexity reduction and efficiency, robustness to near-end

signals and nonlinearities. The main contribution of this thesis is therefore a step towards bridging the gap between fundamental research and practical applications. The next section gives suggestions for future research following the spirit of this thesis.

9.2 Future research

In the first part of the thesis, a recurrent estimation of poles in a batch fashion may be beneficial for AEC/AFC applications. One can use measured impulse responses calculated at system start-up or in subsequent time windows. The study of different adaptive filters for the different types of equalized RTF model has not yet been performed. Presumably the pole-zero model with the shorter residual RIR can be very beneficial in terms of computation reduction. The sparse residual RIR obtained from 1-norm minimization can be exploited in sparse adaptive filtering algorithms.

In the context of acoustic feedback control, it would be worth investigating the use of multiple RANFs for cancelling multiple howling components, and studying its behavior in terms of howling detection. Moreover, including the RANF into an AFC should be investigated as a mechanism to avoid sudden howling frequencies. Finally, the performance of the RANF would be increased by featuring time-variable parameters, such as the pole of the notch filter, the regularization parameters and the step size.

In the WISE, the underlying cost function to derive the step size is a quadratic function of the error. We can look at different cost functions such as the Huber function, the median cost function or the least-absolute value function, which have all been related to robust adaptive filtering. In the GRASS, we can choose different pole filters for each frequency bin or a variable pole depending on the variance in the frequency bins. This in fact relates to the variable momentum LMS, in which the momentum term is variable.

Finally, the SWL-KAPA has been studied only for a simulated nonlinear system and controlled scenario, thus avoiding other problems related to real AEC applications such as insufficient order condition or double-talk situation. Therefore, the application of SWL-KAPA to real AEC applications would be worth investigating. Further research in this context would be beneficial, particularly in view of complexity reduction, optimization of the sliding window with respect to the leakage term, the use of other kernels, the "kernelization" of other adaptive filters used in AEC and the dynamic selection of the kernel weights.

Publication List[1]

International Journal Papers

1. Gil-Cacho J. M., Signoretto M., van Waterschoot T., Moonen M. and Holdt Jensen S., "Nonlinear Acoustic Echo Cancellation based on a Sliding-Window Leaky Kernel Affine Projection Algorithm", in IEEE Transactions on Audio Speech and Language Processing, vol. 21, no. 9, pp. 1867-1878, Sept. 2013.
2. Gil-Cacho J. M., van Waterschoot T., Moonen M. and Holdt Jensen S., "Wiener Variable Step Size and Gradient Spectral Variance Smoothing for Double-Talk-Robust Acoustic Echo cancellation and Acoustic Feedback Cancellation", submitted to Elsevier Signal Processing, May. 2013.
3. Gil-Cacho J. M., van Waterschoot T., Moonen M. and Holdt Jensen S., "A frequency-domain adaptive filter (FDAF) prediction error method (PEM) framework for double-talk-robust acoustic echo cancellation", submitted to IEEE Transactions on Audio Speech and Language Processing.
4. Gil-Cacho J. M., van Waterschoot T., Moonen M. and Holdt Jensen S., "Linear-in-the-parameter nonlinear adaptive filters for acoustic echo cancellation.", accepted with major revision in Journal Audio Engineering Society (JAES), June 2013.

International Conference Papers

1. Gil-Cacho J. M., van Waterschoot T., Moonen M. and Holdt Jensen S., "Transform domain prediction error method for improved acoustic echo and feedback cancellation", in Proc. 20th European Signal Processing Conference 2012 (EUSIPCO'12),Bucharest, Romania, Aug. 2012.

[1] An up-to-date publication list can be accessed online via `http://homes.esat.kuleuven.be/~dspuser`, where the full-text of most publications, as well as additional material (such as presentation slides, and MATLAB software code) is available for download.

2. Gil-Cacho J. M., van Waterschoot T., Moonen M. and Holdt Jensen S., "Nonlinear acoustic echo cancellation based on a parallel-cascade kernel affine projection algorithm",in Proc. of the 37th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'12), Kyoto, Japan, Mar. 2012.
3. Gil-Cacho J. M., van Waterschoot T., Moonen M. and Holdt Jensen S., "Estimation of acoustic resonances for room transfer function equalization", in Proc. of the International Workshop in Acoustic Echo and Noise Control (IWAENC'10), Tel Aviv, Israel, Aug. 2010.
4. Gil-Cacho J. M., van Waterschoot T., Moonen M. and Holdt Jensen S., "Multi-Microphone Acoustic Echo Cancellation using Warped Multi-Channel Linear Prediction of Common Acoustical Poles", in Proc. 18th European Signal Processing Conference 2010 (EUSIPCO'10), Aalborg, Denmark, Aug. 2010.
5. Gil-Cacho J. M., van Waterschoot T., Moonen M. and Holdt Jensen S., "Study and characterization of odd and even nonlinearities in electrodynamic loudspeakers by means of periodic random-phase multisine", in Proc. of the 127th Audio Engineering Society (AES) convention, NY, USA, Oct. 2009.
6. Gil-Cacho J. M., van Waterschoot T., Moonen M. and Holdt Jensen S., "Regularized Adaptive Notch Filters for Acoustic Howling Suppression", in Proc. 17th European Signal Processing Conference 2009 (EUSIPCO'09), Glasgow, UK, Aug. 2009