INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6” x 9” black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI
A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700  800/521-0600
Nonlinear Adaptive Filtering with Application to Acoustic Echo Cancellation

submitted by

A. Neil Birkett, B. Eng, M. Eng.

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

Ottawa-Carleton Institute for Electrical Engineering
Faculty of Engineering
Department of Systems and Computer Engineering

Carleton University
Ottawa, Ontario, Canada K1S 5B6
Feb. 28, 1997

© copyright
1997, Neil Birkett
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author’s permission.

L’auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L’auteur conserve la propriété du droit d’auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-26845-4
The undersigned recommend to the Faculty of Graduate Studies and Research acceptance of the thesis

Nonlinear Adaptive Filtering with Application to

Acoustic Echo Cancellation

submitted by

A. Neil Birkett, B.Eng, M. Eng.

in partial fulfillment of the requirements for

the degree of Doctor of Philosophy

[Signature]

Thesis Supervisor

[Signature]

Chair, Department of Systems and Computer Engineering

[Signature]

External Examiner

Carleton University

April 14, 1997
Abstract

This thesis deals with nonlinear adaptive filtering for identification of systems composed of weakly nonlinear systems convolved with large order linear systems. The intended use is for acoustic echo cancellers (AEC's) operating in handsfree telephones (HFT's) where a microphone and nonlinear loudspeaker share a common enclosure.

Limitations of AEC's are first determined using analytical models verified with simple computer simulations and real-world experimental data. It is shown that enclosure resonances and vibrations, loudspeaker nonlinearity, system undermodelling and audio transducer quality all affect the achievable Echo Return Loss Enhancement (ERLE) performance.

A third-order Volterra filter achieves 6.2dB ERLE improvement over a linear Finite Impulse Response (FIR) filter, however with a training complexity almost 20 times the simple Normalized Least Mean Square (NLMS) algorithm.

Simple feedforward neural based filters are shown to achieve the best performance/complexity trade-off for compensating nonlinear loudspeaker effects in AEC's, compared to Volterra and recursive Infinite Impulse Response (IIR) structures. A simple two-stage neural filter is proposed. It achieves 11 dB improvement over the NLMS-FIR structure with a 2% increase in complexity.

By using a mixed linear/sigmoidal activation function in the neural based filters, several dB's of ERLE improvement can be achieved depending on the range of the linear section and the severity of nonlinearity. An adaptive activation function is subsequently proposed, with a resulting in ERLE improvement of between 1 and 5 dB over the equivalent fixed activation function architecture.
A nonlinear fast conjugate gradient (NFCG) backpropagation algorithm is developed next for improving the convergence rate for coloured signals like speech. The algorithm has a simple gradient-based update and provides a complexity/performance trade-off determined by the size of a gradient window $n_w$. When applied to the two stage neural filter using real speech signals, a 5 dB improvement in ERLE is achieved compared to the Stabilized Fast Transversal Filter (SFTF) with the same initial convergence rate as the SFTF algorithm.

A linear version of the NFCG is also developed using a line search based on a variable step size technique.
Acknowledgments

This thesis was completed with the help of many people who contributed both directly and indirectly in many ways.

First and foremost, it is my pleasure to acknowledge my supervisor Dr. Rafik Goubran for his constant support, motivation, patience and reassurance during some trying times. I am indebted to you.

I am grateful to Jeff Lariviere, Trevor Moat, Tim Webster, and Nong Li, people who shared the DSPLAB with me for several years, and also who provided invaluable discussions, diversions, and entertainment. Thanks also to Rajeev Bector for helping design and develop the experimental setup. To Danny Lemay for always attending to the DSPLAB network when required, and keeping the PCs operational.

Thanks to Dr. Jack Kelly for allowing unlimited access to the anechoic chamber in the Loeb Building.

To the Nortel staff, special thanks go to Dr. Andre Van Shyndel for providing speech files, several loudspeakers handsfree telephone sets and vital information on audio transducers. Thanks also to Scot McLennan for providing valuable feedback on industrial concerns, and especially Dr. Heping Ding who provided constructive comments on Chapter 6 of this thesis.

To the Department of Systems and Computer Engineering at Carleton University, I thank you for financial support that was provided through scholarship and teaching employment. Thanks are also due to NSERC, OGS, TRIO and Nortel for additional financial assistance.
To my mother and father and sister I would like to express a deep thanks for the encouragement and love that have always been there.

I would like to thank Darlene, my wife, for being loving, patient and understanding during the completion of this thesis, and finally my sons Colin, Byron and Sheldon for always providing me with a daily reminder of the really important studies in life.
# Table of Contents

Abstract ......................................................................................................................... iii

Acknowledgments .......................................................................................................... v

Table of Contents .......................................................................................................... vii

List of Figures ................................................................................................................ xi

List of Tables .................................................................................................................. xviii

List of Symbols .............................................................................................................. xix

List of Abbreviations .................................................................................................... xxiv

## Chapter 1

**Introduction** .................................................................................................................. 1

1.1 Motivation for the Research ..................................................................................... 1

1.2 Contributions of the Thesis .................................................................................... 3

1.3 Thesis Outline and Scope ....................................................................................... 6

## Chapter 2

**The Electrodynamical Loudspeaker** ......................................................................... 8

2.1 Loudspeaker Model ............................................................................................... 8

2.2 Nonlinear Modelling ............................................................................................. 13

2.3 Measurement of Nonlinear Parameters ............................................................... 16

2.4 Methods for Reducing Nonlinear Distortion ....................................................... 16

2.5 Summary ................................................................................................................ 17

## Chapter 3

**Review of Nonlinear Adaptive Filtering Techniques** .............................................. 18

3.1 Applications and a Survey ..................................................................................... 18

3.2 Regressors, Mappings and Definitions .................................................................. 21

3.3 Linear Models ....................................................................................................... 24

3.4 Search Methods and Algorithms .......................................................................... 26

3.4.1 Gradient Learning Algorithms for FIR Structures ........................................ 28

3.4.2 Gradient Learning Algorithms for IIR Structures ......................................... 29

3.4.3 Recursive Least Squares Learning Algorithms ............................................. 31

3.4.4 Conjugate Gradient Learning Algorithms .................................................... 33

3.5 Nonlinear Models .................................................................................................. 38

3.6 The Volterra Filter .................................................................................................. 39

3.6.1 Performance Surfaces ..................................................................................... 42
5.4.2 Experimental Results ................................................................. 118
5.4.3 Discussion ................................................................................. 119
5.5 Variable Activation Function ....................................................... 121
  5.5.1 Development of the Learning Algorithm ................................. 122
  5.5.2 Simulation Examples .............................................................. 128
  5.5.3 Experimental Results ............................................................ 129
  5.5.4 Discussion ............................................................................. 130
5.6 MLP with FIR Synapses and Variable Activation Function .......... 131
  5.6.1 Network Architecture .............................................................. 132
  5.6.2 Derivation of the Modified Temporal BP Learning Algorithm ... 133
  5.6.3 Simulation Results ................................................................. 142
  5.6.4 Experimental Results ............................................................ 144
  5.6.5 Discussion ............................................................................. 146
5.7 Summary ...................................................................................... 146

Chapter 6 Conjugate Gradient Methods for Improved Performance .......... 149

  6.1 Fast Conjugate Gradient Backpropagation ............................... 150
    6.1.1 Fast Conjugate Gradient Algorithm for Linear Adaptive Filters 150
    6.1.2 Extension of the FCG Algorithm to Neural Networks .......... 153
    6.1.3 Computer Simulation ............................................................ 157
    6.1.4 Experimental Results ............................................................ 159
    6.1.5 Discussion ............................................................................. 163
  6.2 Conjugate Gradient Reuse Algorithm with Dynamic Line Search ... 164
    6.2.1 Inaccurate Line Search ............................................................ 165
    6.2.2 The MVSS Line Search Algorithm ......................................... 165
    6.2.3 Maximum and Minimum Step Sizes ....................................... 167
    6.2.4 Variable Step Size CG Algorithm using MVSS Line Search ... 168
    6.2.5 Gradient Reuse ................................................................. 171
    6.2.6 Complexity ........................................................................... 173
    6.2.7 Computer Simulations ............................................................ 174
    6.2.8 Application to Acoustic Echo Cancellation ......................... 180
    6.2.9 Discussion ............................................................................. 184
  6.3 Summary ...................................................................................... 185

Chapter 7 Conclusions 186

  7.1 Summary of this Research .......................................................... 186
  7.2 Summary of Contributions .......................................................... 192
  7.3 Suggestions for Future Research ............................................... 193
  7.4 Conclusion ................................................................................. 194

References ....................................................................................... 195

Appendix A Recording Venues ......................................................... 204

Appendix B Equipment Parameters .................................................. 207

ix
### Appendix C  
**Circuit Schematics**

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.1</td>
<td>Primary Conditioning Amplifier</td>
<td>208</td>
</tr>
<tr>
<td>C.2</td>
<td>Reference Conditioning Amplifier</td>
<td>209</td>
</tr>
<tr>
<td>C.3</td>
<td>Switched Capacitor Filter</td>
<td>210</td>
</tr>
<tr>
<td>C.4</td>
<td>Resistive Attenuator Circuits</td>
<td>211</td>
</tr>
<tr>
<td>C.5</td>
<td>General Purpose Amplifiers</td>
<td>212</td>
</tr>
</tbody>
</table>

### Appendix D  
**Algorithms**

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>D.1</td>
<td>The LMS Algorithm</td>
<td>213</td>
</tr>
<tr>
<td>D.2</td>
<td>The Normalized LMS Algorithm</td>
<td>213</td>
</tr>
<tr>
<td>D.3</td>
<td>The Modified Variable Step Size Algorithm</td>
<td>214</td>
</tr>
<tr>
<td>D.4</td>
<td>The Exponentially Weighted RLS Algorithm</td>
<td>215</td>
</tr>
<tr>
<td>D.5</td>
<td>The Accelerated 8N-SFTF Algorithm</td>
<td>216</td>
</tr>
<tr>
<td>D.6</td>
<td>Equation Error LMS-IIR Algorithm</td>
<td>219</td>
</tr>
<tr>
<td>D.7</td>
<td>Output Error LMS-IIR Algorithm with Simplified Gradient</td>
<td>219</td>
</tr>
<tr>
<td>D.8</td>
<td>The LMS Volterra Algorithm</td>
<td>223</td>
</tr>
</tbody>
</table>

### Appendix E  
**Statistics of a Nonlinear Function**

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>E.1</td>
<td>Mean and Variance</td>
<td>225</td>
</tr>
<tr>
<td>E.2</td>
<td>The Normal and Uniform Distribution</td>
<td>225</td>
</tr>
<tr>
<td>E.3</td>
<td>Mean and Variance of a Nonlinear Function</td>
<td>227</td>
</tr>
<tr>
<td>E.4</td>
<td>Nonlinear Examples</td>
<td>227</td>
</tr>
<tr>
<td>E.5</td>
<td>Implications for Adaptive Systems</td>
<td>230</td>
</tr>
</tbody>
</table>
List of Figures

FIGURE 2.1 Loudspeaker electro-mechanical model....................................................... 9
FIGURE 2.2 (a) Typical force vs. displacement curve of a loudspeaker. (b) Typical force vs. displacement curve of the spider................................................................. 12
FIGURE 2.3 Comparison of the input signal e(t) with the linear model and nonlinear model outputs. The nonlinear model exhibits soft-clipping during amplitude peaks. .... 15
FIGURE 2.4 Nonlinear state-space model impulse response........................................ 15
FIGURE 3.1 A weakened Volterra filter that consists of cascaded linear-nonlinear-linear subsystems........................................................................................................ 20
FIGURE 3.2 An adaptive filter..................................................................................... 22
FIGURE 3.3 EE and OE model structures in system identification.............................. 30
FIGURE 3.4 An adaptive Volterra filter structure consists of a number of homogenous systems operating on extended input vectors. The extended inputs are calculated using the tensor product nonlinear mapping strategy........................................ 41
FIGURE 3.5 (a) Linear filter with two weights. (b) Nonlinear filter with two weights........ 42
FIGURE 3.6 Error performance surface (a) Linear model. (b) Nonlinear model has elliptical performance surface................................................................. 42
FIGURE 3.7 Forward signal propagation in a neural network. The single weight values θ are the bias weights................................................................. 46
FIGURE 3.8 Backward error propagation................................................................. 48
FIGURE 4.1 Echo cancellation in the handsfree environment. The hybrid echo canceller is shown on the left and the acoustic echo canceller is shown on the right. The echo replica is subtracted from primary signal to give the error signal ................. 53
FIGURE 4.2 Block diagram of the experimental setup.................................................. 60
FIGURE 4.3  HFT#1 impulse responses. (a) as measured in the anechoic chamber (b) as measured in conference room #2 ................................................................. 61

FIGURE 4.4  Optimum steady-state ERLE performance of the experimental setup based on the converged ERLE of a 10 tap NLMS adaptive filter with a step size of 0.5 (Linear FIR curve), and an estimate of the SNR of the primary signal (Asymptote curve) ............................................................................. 64

FIGURE 4.5  Comparison of optimum steady-state ERLE of the Ariel, the DAT and Ariel, and the full setup based on a 10 tap NLMS adaptive filter with a step size of 0.5. .... 65

FIGURE 4.6  Converged ERLE for six HFT's at various sound pressure levels (measured 1m from the loudspeaker) in an anechoic chamber. 1000 tap FIR trained with NLMS, all cases. ............................................................................................................. 66

FIGURE 4.7  Approximation error bound [dB] for identifying LREM's with different reverberation times. Impulse responses are obtained from measured LREM's truncated to 256 points. (a) Slowly decaying LREM needs an IIR filter order of 200 to achieve a -30 dB error bound. (b) Quickly decaying LREM needs an IIR order of 110 to achieve the same error bound. ........................................... 69

FIGURE 4.8  Experimental results of IIR modelling showing ERLE vs. ratio of poles to zeros for IIR models fitted to experimental LREM data. (a) OE-IIR model for HFT#6 in conference room #2. (b) ARX modelling for SPK#1 and MIC#2 in an anechoic chamber. ............................................................................................................. 70

FIGURE 4.9  Complete LREM model includes enclosure reflections and vibration as well as transducer nonlinear responses. The model should also include room, quantization and circuit noise ......................................................................... 73

FIGURE 4.10  (a) Total MSE as a function of m where \(B_d=B_c=16\) bits, MMSE=1e-6, M=500 taps, reference input variance \(s_r^2=0.1\). The dashed line shows the equivalent infinite precision case. (b) ERLE for a variable primary signal power when \(p(n)\) is quantized to 16 bits (dotted line) and for \(p(n)\) unquantized(solid line). .......... 75

FIGURE 4.11  Undermodelling of the LREM. Measured results of HFT #6 at 85 dB SPL. (a) Impulse response measured in a conference room #2. (b) calculated TIP/TP and measured ERLE using NLMS algorithm. ................................................................. 77
FIGURE 4.12  Comparison of TIP/TP ratio with Hankel error bound for data measured in conference room #2 using HFT #6 at 65 dB SPL. .................................................. 78

FIGURE 4.13  Effects on achievable ERLE due to vibration and rattling. HFT #1. Converged ERLE as volume in increased from 60 dB SPL to 100 dB SPL with various handset configurations. 600 tap FIR trained with the NLMS algorithm, all cases. ........................................................................................................... 81

FIGURE 4.14  Effect of enclosure vibration and nonlinearity. (a) HFT #1. Primary signal PSD with loudspeaker and microphone inside the HFT enclosure. Out-of-band components increase in level as the volume is increased from 60 dB SPL to 100 dB SPL. (b) same as (a) but with components removed from enclosure and mounted inside a standard baffle. ...................................................................................... 82

FIGURE 4.15  Experimental Results inside the anechoic chamber. (a) Impulse response of HFT#5 measured at medium volume. (b) Impulse response of the microphone and loudspeaker removed removed from HFT #5 and placed in a standard baffle is significantly different. Parameters: NLMS algorithm, a=0.5, 32000 samples @ 16kHz, averaged over last 2000. .......................................................................................... 83

FIGURE 4.16  Nonlinear system identification model. The system to be identified consists of a fixed or variable memoryless nonlinearity followed by an exponentially decaying impulse transfer function. ................................................................. 85

FIGURE 4.17  Telephony Filter Characteristics. (a) Bandpass (b) Lowpass. ......................... 86

FIGURE 4.18  Variable hyperbolic tangent squashing function for different values of the parameter a. .................................................................................................................. 87

FIGURE 4.19  Simulation results showing ERLE performance of an adaptive FIR filter trained with the DC-compensated NLMS algorithm. (a) Distortion model #1. (b) Distortion model #2. (c) Distortion Model #3. ................................................................. 88

FIGURE 4.20  Simulation results showing the PSD of the primary, reference and error signals for distortion model #3, a=1.5. (a) Unfiltered primary signal. PSD of error signal closely approximates the primary signals at the frequencies beyond the filter cutoff resulting in a converged ERLE of 18.14 dB. (b) Filtered primary signal.
PSD of error signal is attenuated due to filter, however, in-band distortion products still limit ERLE to 18.94 dB. ........................................................................................................... 89

FIGURE 4.21 Experimental results showing the PSD of the primary, reference and error signals for HFT #1 transducer components as measured inside the anechoic chamber. The error signal closely approximates the primary signals at the frequencies beyond the filter cutoff. .......................................................................................... 90

FIGURE 4.22 Combination of high or low quality loudspeaker (HQL or LQL) and a high or low quality microphone (HQM or LQM) on the ERLE performance as a function of loudspeaker power. The loudspeaker quality is the major component affecting achievable ERLE and the microphone quality is secondary. Parameters: 1000 tap dc compensated NLMS, a=0.5 ........................................................................................................... 91

FIGURE 4.23 Achievable ERLE as a function of physical limitations ................................................................................................................................. 94

FIGURE 5.1 Simulation results. Volterra identification of a nonlinear system generated using distortion model #1 (a) converged ERLE vs. SDR, unfiltered r(n). (b) convergence curve for highest distortion level, unfiltered r(n). (c) converged ERLE vs. SDR, filtered r(n). (d) convergence curve for highest distortion level, filtered r(n). .................................................................................................................... 97

FIGURE 5.2 Simulation results. Converged ERLE using a Volterra filter for the identification of a simulated nonlinear system generated using (a) distortion method #2 (b) distortion method #3 ............................................................................................................ 98

FIGURE 5.3 HFT #1 results, anechoic chamber. Converged ERLE vs. SPL for (a) separately mounted transducer components and (b) with components mounted inside the enclosure........................................................................................................ 99

FIGURE 5.4 Volterra tap weights for HFT #1 at 100 dB SPL. (a) Linear tap weights, (b) quadratic tap weights. (c) Cubic tap weights. (d) difference between Volterra and FIR linear tap weights. ................................................................................................................... 100

FIGURE 5.5 HFT #6 results. (a) Converged ERLE vs. SPL. (b) Convergence curves for data at 90 dB SPL........................................................................................................................................... 101

FIGURE 5.6 Time delay neural network (TDNN) filter ................................................................................................................................. 103

xiv
FIGURE 5.7  Simulation #1 results showing converged ERLE vs. increasing number of hidden nodes for a (10,h1,1) TDNN. (a) Unfiltered reference (b) Filtered reference. .................................................. 104

FIGURE 5.8  Simulation results for (a) distortion method #1, (b) distortion method #2 and (c) distortion method #3. A (10,5,1) TDNN is used for (a) and (b) and a (15,5,1) TDNN is used for (c) to cover the impulse response generated by method #3. ...................................................................................... 106

FIGURE 5.9  Adaptive activation function and derivative with respect to s for p=0.0, 0.5 and 0.9 ........................................................................................................ 107

FIGURE 5.10  Effect of changing linear region in a mixed linear-sigmoidal activation function. (10,5,1) TDNN on distortion model #1 with a=1 and b=c=0.01 (low distortion) and b=c=0.5 (high distortion). ........................................................................ 108

FIGURE 5.11  HFT #3 experimental results, anechoic chamber, components only. .................. 109

FIGURE 5.12  HFT #1 results, anechoic chamber, TDNN. Converged ERLE vs. SPL for (a) separately mounted transducer components and (b) with components mounted inside the HFT enclosure. ........................................................................... 110

FIGURE 5.13  HFT #6 results for a 2 layer TDNN. (a) Converged ERLE vs. SPL for the NLMS, (1000,1,1) and (200,1,1) structure. (b) Convergence curve for data at 65 dB SPL. ................................................................................................. 110

FIGURE 5.14  HFT #3 components (i.e. no enclosure), anechoic chamber. In an undermodelled state a TDNN obtains a higher ERLE as compared with the FIR structure. 112

FIGURE 5.15  Proposed two stage nonlinear AEC structure consists of both nonlinear and linear sections ........................................................................................................ 116

FIGURE 5.16  Experimental results showing performance of the proposed structure using HFT #6 in conference room #2. (a) Converged ERLE, keys taped down. (b) Plot of PSD of signals for the taped keys case. (c) Converged ERLE, keys not taped. (d) Convergence curve, keys not taped down. ........................................................................ 120

FIGURE 5.17  Adaptive activation function and derivative with respect to p as the input s varies for values of p=0.0, 0.5 and 0.9 ........................................................................ 124

FIGURE 5.18  Alternate activation function and derivative with respect to s as the input s varies (b) derivative with respect to p as the input s varies. .................................................. 126
FIGURE 5.19 Simulation results for distortion method #1, (10,5,1) variable activation TDNN, filtered reference signal. (a) converged ERLE vs. SDR (b) convergence curve for the highest distortion level. ........................................................................... 129

FIGURE 5.20 Experimental results for the variable activation function TDNN. (a) Converged ERLE vs. SPL (b) convergence curve for an SPL of 95 dB. ............................................. 130

FIGURE 5.21 Using the FIR MLP to represent a cascaded linear-nonlinear-linear subsystem (i.e. nonlinear HFT). ....................................................................................... 132

FIGURE 5.22 Forward signal propagation in the FIR MLP. .................................................. 134

FIGURE 5.23 Backward filter propagation of “accumulated” gradient terms. ....................... 139

FIGURE 5.24 System identification using the proposed model. ........................................ 142

FIGURE 5.25 Comparison of convergence using the proposed algorithms. ....................... 144

FIGURE 5.26 Experimental results. HFT #6 in conference room 2 showing a comparison of FIR, VA-FIR MLP using 150/800 taps in first/second FIR sections, and simplified VA-FIR MLP with only one FIR section following adaptive sigmoid. ................................................................. 145

FIGURE 6.1 System identification model. ............................................................................. 157

FIGURE 6.2 Simulation results showing the averaged NMSE performance of the BP and NFCG algorithms with n_w=2, 5, and 10 for the system identification model of Figure 6.1. Two hundred independent trials are used in the averaging process. ........................................................................................................ 158

FIGURE 6.3 Experiment #1 results comparing converged ERLE curves of a 150 tap FIR structure trained using the NLMS algorithm with that of a TDNN trained with the BP and NFCG algorithm. ................................................................................... 160

FIGURE 6.4 Reference signal speech signal. ........................................................................ 161

FIGURE 6.5 Experiment #2 results. Converged ERLE results with speech input. Gaps show where pauses in speech are located. .............................................................. 162

FIGURE 6.6 Experiment #2 results. Close up of speech period between 6 and 8 seconds. The two stage neural filter trained with proposed algorithm achieves approximately 5 dB higher ERLE that the FIR filter trained with stabilized SFTF algorithm. .... 162
FIGURE 6.7  System identification model. An uncorrelated noise source with variance $s_N^2$ is added to the adaptive filter output $y(n)$ to produce an SNR of 50 dB.................. 174

FIGURE 6.8  Simulation #1 results. Correlated noise input with a sudden change in the unknown system transfer function at iteration 1000............................................. 177

FIGURE 6.9  Simulation #2 results. Comparison of FCG and VCGR using a limited gradient reuse rate. Correlated noise input with a sudden change in the unknown system transfer function at iteration 1000. ................................................................. 178

FIGURE 6.10 Simulation #3 results. Simplified VCG performance results (VCGR). Correlated noise input with a sudden change in the unknown system transfer function at iteration 1000................................................................. 180

FIGURE 6.11 Time series plot of the speech excitation signal...................................................... 182

FIGURE 6.12 Time series plot of the complete excitation signal...................................................... 182

FIGURE 6.13 Results for different algorithms using speech excitation. A nonstationarity occurs between 11.5 and 14 seconds................................................................. 183

FIGURE 6.14 Close up view illustrating initial convergence performance of algorithms........... 183

FIGURE 6.15 Close up view illustrating tracking performance of algorithms......................... 184

FIGURE A.1  Conference room #1 (Minto 3033)................................................................. 204

FIGURE A.2  Conference room #2 (Minto 2014)................................................................. 205

FIGURE A.3  Standard baffle used to test stand-alone loudspeakers. (a) Dimensions of plywood section (b) Detail of plexiglass submount........................................... 206

FIGURE D.1  The OE simplified gradient evaluation................................................................. 221

FIGURE E.1  Computed SDR for a cubic order system where the standard deviation of the input signal changes................................................................. 230
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE 3.1</td>
<td>Special cases of the general linear model</td>
<td>25</td>
</tr>
<tr>
<td>TABLE 4.1</td>
<td>Objective performance requirements for handsfree telephones.</td>
<td>55</td>
</tr>
<tr>
<td>TABLE 4.2</td>
<td>Parameters for experimental results presented in Figures 4.8(a) and (b)</td>
<td>71</td>
</tr>
<tr>
<td>TABLE 4.3</td>
<td>Comparison of Algorithm Complexities</td>
<td>80</td>
</tr>
<tr>
<td>TABLE 4.4</td>
<td>Electret microphone acoustic sensitivity</td>
<td>84</td>
</tr>
<tr>
<td>TABLE 4.5</td>
<td>Mechanical vibrational sensitivity</td>
<td>84</td>
</tr>
<tr>
<td>TABLE 4.6</td>
<td>Distortion Parameters</td>
<td>87</td>
</tr>
<tr>
<td>TABLE 4.7</td>
<td>Transducer Parameters</td>
<td>90</td>
</tr>
<tr>
<td>TABLE 5.1</td>
<td>Summary of simulation results shown in Figure 5.1</td>
<td>97</td>
</tr>
<tr>
<td>TABLE 5.2</td>
<td>Summary of experimental parameters and results for the Volterra filter.</td>
<td>101</td>
</tr>
<tr>
<td>TABLE 5.3</td>
<td>Simulation results using TDNNs with one and two hidden layers</td>
<td>105</td>
</tr>
<tr>
<td>TABLE 5.4</td>
<td>Summary of experimental results showing regions of greatest improvement</td>
<td>111</td>
</tr>
<tr>
<td>TABLE 5.5</td>
<td>TDNN complexity assuming a single node output</td>
<td>112</td>
</tr>
<tr>
<td>TABLE 5.6</td>
<td>Complexity of the two stage neural filter</td>
<td>117</td>
</tr>
<tr>
<td>TABLE 6.1</td>
<td>Experiment #2 parameters</td>
<td>160</td>
</tr>
<tr>
<td>TABLE 6.2</td>
<td>Comparison of algorithm complexity</td>
<td>174</td>
</tr>
<tr>
<td>TABLE 6.3</td>
<td>List of parameters used for simulations #1, #2 and #3</td>
<td>176</td>
</tr>
<tr>
<td>TABLE 6.4</td>
<td>Parameter and complexity comparison for FCG (nw=5 and 8) and VCGR algorithm (nw=5, R=2)</td>
<td>177</td>
</tr>
<tr>
<td>TABLE 6.5</td>
<td>Comparative complexity using NLMS, FCG, VCG and VCGR (nw=R=P=5).</td>
<td>179</td>
</tr>
<tr>
<td>TABLE 6.6</td>
<td>Algorithm parameters</td>
<td>181</td>
</tr>
<tr>
<td>TABLE 6.7</td>
<td>Average ERLE calculated between 11.75-13.5 seconds.</td>
<td>181</td>
</tr>
<tr>
<td>TABLE B.1</td>
<td>Parameters of HFT and transducer equipment</td>
<td>207</td>
</tr>
</tbody>
</table>
List of Symbols

Rules Regarding Boldface Symbols

A boldface uppercase letter generally refers to a matrix of numbers or variables. A boldface lowercase letter generally refers to a vector of numbers or variables. An unbolded uppercase or lowercase letter is a scalar quantity.

Arabic Symbols

\( a, b, c \) Magnitude of linear, quadratic and cubic distortion coefficients.

\( a_i \) \( i \)th feedback coefficient value for a recursive IIR filter.

\( A(q) \) Polynomial of delay operator \( q^{-1} \).

\( b_i \) \( i \)th feedforward coefficient value for an FIR filter.

\( b \) Bias potential.

\( B \) Magnetic flux density in the air gap.

\( B_c \) Number of binary quantization levels for DSP coefficient representation.

\( B_d \) Number of binary quantization levels for DSP data representation.

\( B(z) \) Linear model polynomial coefficients associated with \( x(n) \) regressors.

\( C \) Compliance of the suspension system.

\( \Delta C \) Change in electret microphone capacitance.

\( d(n) \) Sample of desired response at time \( n \).

\( d_k \) Conjugate direction vector at iteration \( k \).

\( d(n) \) Desired signal sample vector at time \( n \).

\( e(n) \) Adaptive filter prediction error at time \( n \).

\( e(t) \) Applied voice coil voltage.

\( E_b \) Voltage induced in the voice coil by the mechanical circuit.

\( E(\cdot) \) Expected value of a function.

\( f_M \) Force deflection characteristic of the loudspeaker cone suspension system.
$\mathbf{F}(\cdot)$  
Hessian of an $m \times m$ matrix.

$\mathbf{g}_k$  
Conjugate gradient estimate at iteration $k$.

$h_p$  
$p^{th}$ order Volterra kernel.

$\mathbf{h}_e$  
Extended Volterra weight vector.

$h_p$  
$p^{th}$ order Volterra coefficient vector.

$H(z)$  
Transfer function in the $z$ domain

$H_1, H_2$  
Linear dispersive system transfer functions.

$i(t)$  
Amplitude of the current in the voice coil.

$J$  
General cost function.

$J_{\text{min}}$  
Minimum mean square error

$J_{\text{tot}}$  
Total mean square error

$l$  
MLP layer index.

$l$  
Length of the voice coil inductor.

$L$  
Inductance of the voice coil.

$m$  
(i) Square matrix size (ii) Total mass of coil, cone and air load.

$M$  
Order of a "supervector".

$n_a$  
Number of pole coefficients.

$n_b$  
Number of zero coefficients.

$n_w$  
CG averaging window length.

$N_p$  
Number of inputs in the $p^{th}$ order section of a polynomial filter.

$p(n)$  
Sample of primary signal at time $n$.

$p$  
Size of linear region in variable activation function.

$p_j(n)$  
$p$ value for the $j^{th}$ node at the $n^{th}$ iteration.

$\Delta p_j(n)$  
Correction value for $p_j(n)$ at the $n^{th}$ iteration.

$p(n)$  
Vector of $p$ parameters.

$\mathbf{P}_k$  
Conjugate gradient estimate at $y_k$ at iteration $k$.

$\mathbf{P}(n)$  
Matrix derived from input $x(n)$ used to improve convergence rate.
$P$ Conjugate gradient reuse rate.

$q(n)$ Measure of the input signal power.

$Q$ General $m \times m$ matrix.

$r(n)$ Sample of reference signal at time $n$.

$r_M$ Total mechanical resistance due dissipation in the air load and suspension system.

$R$ (i) Conjugate direction reuse rate (ii) Total electrical resistance of the generator and voice coil.

$R(n)$ Autocorrelation matrix of an input vector $x(n)$.

$s$ Activation potential.

$s_{j}^{(l)}(n)$ Input activation to node $j$ in layer $l$.

$u(n)$ Input regression vector.

$\Delta V$ Change in microphone output voltage.

$v_j^{(l)}(n)$ Backward error activation for the $j^{th}$ node in layer $l$ at time $n$.

$w_{i,j}^{(l)}(n)$ Weight connecting the $i^{th}$ node in layer $l$ to the $j^{th}$ node in layer $l+1$ at time $n$.

$w_{i,j}^{(l)}(n)$ Synaptic FIR weight vector connecting the $i^{th}$ node in layer $l$ to the $j^{th}$ node in layer $l+1$ at time $n$.

$w_{i,j}^{(l)}(n)$ Accumulated synaptic FIR weight vector connecting the $i^{th}$ node in layer $l$ to the $j^{th}$ node in layer $l+1$ at time $n$, of length $n_w$.

$w_{bias}$ Bias weight.

$w(n)$ Weight vector at time $n$.

$w_{opt}$ Optimum weiner filter coefficient vector.

$x(n)$ Current input sample at time $n$.

$x(n)$ Input sample vector at time $n$.

$x_j^{(l)}(n)$ Output of node $j$ in layer $l$.

$x_j^{(l)}(n)$ Output vector of node $j$ in layer $l$.

$x_e$ Extended Volterra regression vector.

$y(n)$ Adaptive filter output sample at time $n$.

$y(n)$ Output sample vector at time $n$. 
$y_k$ Estimate of the new conjugate weight vector.

**Greek Symbols**

$\alpha, \beta, \gamma$ Nonlinear compliance modelling coefficients.

$\alpha$ Momentum constant.

$\tilde{\alpha}$ Normalized step size parameter

$\alpha_k$ Direction vector constants.

$\beta_k$ Coefficient for generating $d_{k+1}$ from $d_k$.

$\gamma$ MVSS convergence rate parameter.

$\Gamma$ MVSS filtered error averaging time constant.

$\Gamma_H$ Hankel matrix of an all zero impulse response.

$\delta_j^{(l)}(n)$ Local gradient delta for node $j$ in layer $l$ at time $n$.

$\Delta_j^{(l)}(n)$ Accumulated local gradient delta vector for node $j$ in layer $l$ at time $n$ of length $n_w$.

$\varepsilon$ Small positive constant.

$\theta_j^{(l)}$ Bias potential added to node $j$ in layer $l$.

$\Sigma$ Diagonal matrix of Hankel singular values.

$\varphi$ (i) General nonlinear operator (ii) Sigmoid operator.

$\varphi'_s(s, p)$ Derivative of a sigmoid operator with respect to the $s$ parameter.

$\lambda$ Forgetting factor (RLS, FTF, SFTF)

$\lambda_i$ $i^{th}$ eigenvalue of the input correlation matrix.

$\lambda_{max}$ Maximum eigenvalue of the input correlation matrix.

$\rho$ Acceleration factor for the accelerated SFTF algorithm.

$\mu$ Step size parameter.

$\mu_{TDNN}$ Step size parameter for the TDNN portion of the two layer neural filter.

$\mu_2, \mu_3$ Step size parameters for quadratic and cubic sections of a third order Volterra filter.

$\nabla_w$ Multidimensional gradient with respect to vector $w$.

$\sigma_i$ $i^{th}$ Hankel singular value.
$\sigma^2_c$  Coefficient quantization noise.

$\sigma^2_d$  Data quantization noise.

$\sigma^2_p$  Variance of the primary signal.

$\sigma^2_e$  Variance of the error signal.

$\sigma^2_R$  Room noise.

$\sigma^2_M$  Microphone noise.

$\omega$  Angular frequency.

$\xi_j(n)$  Local gradient for the parameter $p_j(n)$.

$\zeta$  MVSS step size averaging parameter.
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AEC</td>
<td>Acoustic Echo Canceller</td>
</tr>
<tr>
<td>AF</td>
<td>Adaptive Filter</td>
</tr>
<tr>
<td>AIR</td>
<td>Acoustic Impulse Response</td>
</tr>
<tr>
<td>AR</td>
<td>Auto Regressive</td>
</tr>
<tr>
<td>ARMA</td>
<td>Auto Regressive Moving Average</td>
</tr>
<tr>
<td>ARMAX</td>
<td>Auto Regressive Moving Average with Exogenous inputs</td>
</tr>
<tr>
<td>ARX</td>
<td>Auto Regressive with Exogenous Input</td>
</tr>
<tr>
<td>BP</td>
<td>Backpropagation</td>
</tr>
<tr>
<td>BPF</td>
<td>Bandpass Filter</td>
</tr>
<tr>
<td>CES</td>
<td>Composite Error Surface</td>
</tr>
<tr>
<td>CG</td>
<td>Conjugate Gradient</td>
</tr>
<tr>
<td>CGR</td>
<td>Conjugate Gradient Reuse</td>
</tr>
<tr>
<td>CRA</td>
<td>Composite Regressor Algorithm</td>
</tr>
<tr>
<td>DAT</td>
<td>Digital Audio Tape</td>
</tr>
<tr>
<td>DSP</td>
<td>Digital Signal Processing</td>
</tr>
<tr>
<td>DT</td>
<td>Double Talk</td>
</tr>
<tr>
<td>DVM</td>
<td>Digital Voltmeter</td>
</tr>
<tr>
<td>EBP</td>
<td>Enhanced Backpropagation</td>
</tr>
<tr>
<td>EE</td>
<td>Equation Error</td>
</tr>
<tr>
<td>EKA</td>
<td>Extended Kalman Algorithm</td>
</tr>
<tr>
<td>ELR</td>
<td>Early-to-Late Ratio</td>
</tr>
<tr>
<td>ERLE</td>
<td>Echo Return Loss Enhancement</td>
</tr>
<tr>
<td>FAP</td>
<td>Fast Affine Projection</td>
</tr>
<tr>
<td>FCG</td>
<td>Fast Conjugate Gradient</td>
</tr>
<tr>
<td>FIR</td>
<td>Finite Impulse Response</td>
</tr>
<tr>
<td>FFT</td>
<td>Fast Fourier Transform</td>
</tr>
<tr>
<td>FNTF</td>
<td>Fast Newton Transversal Filter</td>
</tr>
<tr>
<td>FTF</td>
<td>Fast Transversal Filter</td>
</tr>
<tr>
<td>GF</td>
<td>Generalized Feedforward</td>
</tr>
</tbody>
</table>

xxiv
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMDF</td>
<td>Generalized Multi-Delay Filter</td>
</tr>
<tr>
<td>HFT</td>
<td>Handsfree Telephone</td>
</tr>
<tr>
<td>HQL</td>
<td>High Quality Loudspeaker</td>
</tr>
<tr>
<td>HQM</td>
<td>High Quality Microphone</td>
</tr>
<tr>
<td>IC</td>
<td>Instantaneous Cost</td>
</tr>
<tr>
<td>ICVA</td>
<td>Instantaneous Cost Variable Activation</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute for Electric and Electronic Engineers</td>
</tr>
<tr>
<td>IIR</td>
<td>Infinite Impulse Response</td>
</tr>
<tr>
<td>LMS</td>
<td>Least Mean Squares algorithm</td>
</tr>
<tr>
<td>LPF</td>
<td>Lowpass Filter</td>
</tr>
<tr>
<td>LQL</td>
<td>Low Quality Loudspeaker</td>
</tr>
<tr>
<td>LQM</td>
<td>Low Quality Microphone</td>
</tr>
<tr>
<td>LREM</td>
<td>Loudspeaker-Room-Enclosure-Microphone</td>
</tr>
<tr>
<td>LRGF</td>
<td>Locally Recurrent Globally Feedforward</td>
</tr>
<tr>
<td>MA</td>
<td>Moving Average</td>
</tr>
<tr>
<td>MLP</td>
<td>Multilayer Perceptron</td>
</tr>
<tr>
<td>MMSE</td>
<td>Minimum Mean Squared Error</td>
</tr>
<tr>
<td>MSE</td>
<td>Mean Squared Error</td>
</tr>
<tr>
<td>MVSS</td>
<td>Modified Variable Step Size</td>
</tr>
<tr>
<td>NARMAX</td>
<td>Nonlinear ARMAX</td>
</tr>
<tr>
<td>NARX</td>
<td>Nonlinear ARX</td>
</tr>
<tr>
<td>NCG</td>
<td>Nonlinear Conjugate Gradient</td>
</tr>
<tr>
<td>NFIR</td>
<td>Nonlinear FIR</td>
</tr>
<tr>
<td>NOE</td>
<td>Nonlinear Output Error</td>
</tr>
<tr>
<td>NLMS</td>
<td>Normalized Least Mean Square</td>
</tr>
<tr>
<td>NSS</td>
<td>Nonlinear State Space</td>
</tr>
<tr>
<td>OE</td>
<td>Output Error</td>
</tr>
<tr>
<td>PSD</td>
<td>Power Spectral Density</td>
</tr>
<tr>
<td>QAM</td>
<td>Quadrature Amplitude Modulation</td>
</tr>
<tr>
<td>RLS</td>
<td>Recursive Least Squares</td>
</tr>
<tr>
<td>RMS</td>
<td>Root Mean Square</td>
</tr>
<tr>
<td>ROM</td>
<td>Read Only Memory</td>
</tr>
<tr>
<td>RPE</td>
<td>Recursive Prediction Error</td>
</tr>
</tbody>
</table>

xxv
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCG</td>
<td>Scaled Conjugate Gradient</td>
</tr>
<tr>
<td>SFTF</td>
<td>Stabilized Fast Transversal Filter</td>
</tr>
<tr>
<td>SDR</td>
<td>Signal to Distortion Ratio</td>
</tr>
<tr>
<td>SMM</td>
<td>Steiglitz McBride Method</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal to Noise Ratio</td>
</tr>
<tr>
<td>SPL</td>
<td>Sound Pressure Level</td>
</tr>
<tr>
<td>SS</td>
<td>State Space</td>
</tr>
<tr>
<td>TC</td>
<td>Total Cost</td>
</tr>
<tr>
<td>TDL</td>
<td>Tapped Delay Line</td>
</tr>
<tr>
<td>TDNN</td>
<td>Time Delay Neural Network</td>
</tr>
<tr>
<td>TIP</td>
<td>Total Impulse Response Power</td>
</tr>
<tr>
<td>TP</td>
<td>Tail Power</td>
</tr>
<tr>
<td>TWT</td>
<td>Travelling Wave Tube</td>
</tr>
<tr>
<td>USASI</td>
<td>United States of America Standards Institute</td>
</tr>
<tr>
<td>VA</td>
<td>Variable Activation</td>
</tr>
<tr>
<td>VCG</td>
<td>Variable Stepsize Conjugate Gradient</td>
</tr>
<tr>
<td>VCGR</td>
<td>VCG with Gradient Reuse</td>
</tr>
<tr>
<td>VSS</td>
<td>Variable Step Size</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Motivation for the Research

Real-time system identification, filtering and/or tracking of signals produced in nonstationary environments requires the use of an adaptive filter. Much of the literature available on adaptive filter theory deals only with linear structures which have difficulty in accurately identifying systems that include nonlinearities. In the real world, examples of such nonlinear systems include the production of human speech signals, audio transducers such as loudspeakers and channel nonlinearities in high-speed data communications channels, usually caused by amplifier circuits operating near saturation.

A handsfree telephone (HFT) is one such system that contains a nonlinear component, the loudspeaker. An acoustic echo canceller (AEC) designed for use in an HFT attempts to remove the acoustic echos by modelling the loudspeaker-room-enclosure-microphone (LREM) system and subtracting an echo replica from the microphone signal. However, in this case, the system to be
identified includes a reverberant room as well as the loudspeaker and therefore is a cascade of both linear and nonlinear sections that have memory associated with the process. The acoustic echoes associated with reverberant rooms can be modelled effectively by linear filters consisting of all-zero adaptive Finite Impulse Response (FIR) structures or in some cases Infinite Impulse Response (IIR) structures consisting of one or more poles. A loudspeaker is a nonlinear device that exhibits hysteresis, hence a structure that can process nonlinear temporal information is required to achieve significant modelling accuracy. Existing linear algorithms in the AEC domain are therefore unable to achieve high echo return loss enhancement (ERLE) when the loudspeaker is operating in the nonlinear region. This idea was originally suggested by Knappe and Goubran [1] as a steady state performance limitation in acoustic echo cancellers and was a primary motivating factor for the work presented here.

This thesis is primarily about the study of nonlinear algorithms aimed at identifying cascaded linear and nonlinear systems with specific application to the reduction of nonlinear loudspeaker distortion effects encountered in the domain of acoustic echo cancellation.

Four basic questions are answered in this thesis:

1. What sort of limitations do typical nonlinear loudspeakers present to achieving high ERLE values in typical HF's? Small inexpensive loudspeakers generate several percent nonlinear distortion at volumes typically used in the handsfree mode, and thus limit the ERLE to values less than 30 dB in most cases.

2. What kind of filters are best suited for nonlinear AEC applications and how can we arrive at that conclusion? Feedforward structures offer simplicity of design compared to recursive structures, especially in the nonlinear domain. Structures proposed also need to be robust in noisy
1.2 Contributions of the Thesis

3. How can an efficient nonlinear structure and training algorithm be designed that is not overly complicated yet provide reasonable improvements in performance? Low complexity is of utmost importance in AEC’s, hence the nonlinear training algorithm should provide a trade-off between complexity and performance. Structures and algorithms based on a combination of neural networks and linear adaptive filtering theory provide encouraging results.

4. Can the new structures/algorithms be successfully applied in real-world applications? The nonlinear AEC’s are applied to experimental data collected on several commercially available HFT’s in conference environments with positive results using both noise and speech signals.

Throughout the thesis, the handsfree telephony AEC is used to demonstrate the effectiveness of the proposed algorithms and structures, with verification using field data collected using a number of commercially available HFT’s in different anechoic and conference room environments. However it should be stressed that in general these algorithms can be applied to many fields, for example, public address (PA) systems, active noise control using remotely placed loudspeakers to cancel unwanted signals, and identification of channel nonlinearities in high speed data communications channels.

1.2 Contributions of the Thesis

The primary contribution of this thesis is the development of a set of new nonlinear adaptive filter structures and algorithms to compensate for nonlinear loudspeaker distortion effects in AEC’s intended for handsfree telephony. The procedure consists of the following steps
1.2 Contributions of the Thesis

- Determination of the limitations in AEC's due to nonlinear transducers.
- Evaluation of existing nonlinear structures for compensating nonlinearity in AEC's.
- Construction of new architectures for compensating for transducer nonlinearity.
- Development of efficient training algorithms for the proposed structures.
- Application and testing of the new nonlinear structures using speech and noise signals recorded in HFT's in audio conference environments.

The main contributions to the field of nonlinear adaptive filtering are as follows:

1. Development of a two stage neural filter consisting of an FIR filter and tapped delay line neural network (TDNN) structure, which successfully models loudspeaker nonlinearity convolved with an echo path.

2. Development and use of a mixed linear sigmoidal activation (squashing) function to replace the hyperbolic tangent function which is commonly used in neural networks.

3. Extension of the mixed linear sigmoidal activation function to the fully adaptive case.

4. Development of a new temporal adaptation mechanism to adapt the variable activation function which is sandwiched between FIR synaptic filters.

5. A fast nonlinear conjugate gradient algorithm for improved convergence speed is developed and then applied to the nonlinear structures above to improve performance, which is verified using real speech signals.

6. A linear algorithm based on the fast conjugate gradient algorithm and the concept of variable step size line search is constructed. A simplified version using gradient reuse is also developed.

A full test setup for performing experimental testing is constructed, consisting of various amplifiers, filters and interface circuits. Experimental determination of the performance limitations of the
1.2 Contributions of the Thesis

test setup using real equipment in anechoic and conference rooms conditions is made. During this phase, it was determined that enclosure vibration, resonances and rattling effects also place a limit on the achievable steady state acoustic echo cancellation. Subsequently the following statements can be made which represent a secondary contribution, specifically to the design and development of HFT's:

1. Enclosure vibration can be a more serious problem than nonlinear distortion at high loudspeaker volumes for typical desktop HFT's in a low noise conference room.

2. Effective design of the acoustic enclosure is necessary to enable the nonlinear algorithms to work properly. Otherwise, the vibration and resonances within the enclosure mask the loudspeaker nonlinearity.

3. A microphone with a low mechanical vibration sensitivity is necessary to mitigate vibration effects.

4. The steady state performance limitations of AEC's for HFT's in a typical low noise conference room environment are in order of severity for the cases studied (i) undermodelling (ii) loudspeaker nonlinearity (iii) vibration and resonances (iv) room noise (v) DSP and algorithmic noise.

---

1. No mention of the performance limitation caused by vibration and resonances has yet been found in the literature on AEC's.
2. Undermodelling is a more serious limitation than other factors when the order of the AEC is much less than the LREM.
3. Only if vibration and resonances are controlled through appropriate enclosure design.
1.3 Thesis Outline and Scope

This thesis has four central chapters: Chapter 4 discusses the acoustic echo cancellation problem with subsections on the experimental setup and performance limitations, including important new results on the effects of vibration and resonances within the HFT enclosure, and nonlinear loudspeaker distortion. Several new nonlinear adaptive structures and algorithms are developed and tested in Chapters 5, and 6. Supporting Chapters 2,3 and 7 review loudspeaker dynamics, provide the necessary background theory in nonlinear adaptive filtering, and summarize and draw conclusions.

Chapter Two presents a quick review of loudspeaker basics, including the lumped parameter equivalent model and an analysis of low frequency nonlinear distortion.

Chapter Three discusses the principles of nonlinear adaptive filtering and corresponding adaptation algorithms. It briefly introduces linear FIR and IIR filters, Volterra filters, and neural networks. This chapter furnishes the reader with the necessary background theory and lays the foundation for the following chapters.

Chapter Four discusses the handsfree telephone problem, presents a description of the experimental setup and discusses the steady state performance limitations of AEC's such as undermodeling of the acoustic transfer function, room noise and DSP/algorithmic noise. A subsection on IIR structures for AEC is presented. Subsections studying the effect of enclosure vibration and resonances within the HFT enclosure, as well as effects of nonlinear transducers are also provided. Simulation and experimental results are presented throughout to provide a measure of the relative severity of the performance limitations.
Chapter Five first presents a comparison of Volterra and neural filters for nonlinear loudspeaker compensation. Subsequently a two stage neural filter is developed to identify a nonlinear LREM and provide measurable improvements in performance. A mixed linear-sigmoidal activation function is proposed and a training algorithm is derived. A new architecture and temporal training algorithm for adaptive activation functions sandwitched between temporal FIR synapses is also derived. The behavior of the proposed models is demonstrated using both simulated and experimental HFT data.

Chapter Six outlines a new fast version of the conjugate gradient algorithm for enhancing the convergence rate of neural filters. The performance of the algorithm is subsequently tested using computer simulations and field data consisting of both noise and speech signals. A linear variation of of the fast conjugate gradient algorithm using the concept of variable step size line search and gradient reuse is also presented.

Chapter Seven summarizes the results and draws conclusions arising from the research work. Significant contributions are highlighted and finally, future research directions are suggested.
Chapter 2

The Electrodynamic Loudspeaker

As mentioned in the first chapter, one focus of this thesis is in applying nonlinear filtering techniques to compensate for loudspeaker nonlinearity in AEC's for handsfree telephony. Before proceeding, a review of loudspeaker dynamics and an analysis of loudspeaker distortion at low frequencies is necessary.

2.1 Loudspeaker Model

In an electrodynamic loudspeaker, sound waves are produced by a diaphragm which moves in response to an alternating current passing through a voice coil which is positioned in a permanent magnetic field. Figure 2.1 illustrates a typical loudspeaker transducer, and equivalent lumped parameter model [2]. The diaphragm can be plane, cone or dome-shaped. The diaphragm is suspended at the outer edge by means of a flexible surround or rim, and at the inner edge by a spider. The spider is rotationally symmetrical and centers the voice coil. It has a large stiffness for radial motion and a smaller but finite stiffness for axial motion. The simplified model of a loudspeaker
behaves like a mass-spring system. The spring part is formed by the suspension and spider and the mass is formed by the diaphragm, the voice coil, the mass of the cone/suspension system and the air load [3].

![Diagram of loudspeaker model](image)

**FIGURE 2.1** Loudspeaker electro-mechanical model.

In the model of Figure 2.1, the parameter \( e(t) \) indicates the internal voltage of the generator, \( R \) is the total electrical resistance of the generator and voice coil, \( L \) is the inductance of the voice coil, \( i(t) \) is the amplitude of the current in the voice coil, \( E_b \) is the voltage induced in the electrical circuit by the mechanical circuit, which equals \( Bl \ dx(t)/dt \). \( B \) is the magnetic flux density in the air gap, \( l \) is the length of the voice coil conductor, and \( x \) is the cone displacement. In the mechanical circuit \( m \) is the total mass of the coil, cone and air load. \( r_m \) is the total mechanical resistance due to dissipation in the air load and the suspension system. \( C \) is the compliance of the suspension and \( f_M \)
is the force generated in the voice coil and equals $Bli$. The mechanical radiation resistance $Z_{rad}$ has a real and imaginary part.

As shown in Figure 2.1 the electrical and mechanical part are connected through the magnetic field. The resulting equations of motion can be described by two coupled nonlinear differential equations [4]:

$$e(t) = i(t) R + L(x(t)) \frac{di(t)}{dt} + B(x(t)) l \frac{dx(t)}{dt}$$  \hspace{1cm} (2.1)

$$B(x(t)) li(t) = m \frac{d^2 x(t)}{dt^2} + r_M \frac{d}{dt} x(t) + \frac{x(t)}{C(x(t))}$$ \hspace{1cm} (2.2)

where the displacement dependent parameters $L(x(t))$, $B(x(t))$ and $C(x(t))$ have been modelled by a Taylor series expansion, which can be truncated after an arbitrary number of terms:

$$L(x(t)) = L_0 + L_1 x(t) + L_2 (x(t))^2$$  \hspace{1cm} (2.3)

$$B(x(t)) = B_0 + B_1 x(t) + B_2 (x(t))^2$$  \hspace{1cm} (2.4)

$$C(x(t)) = C_0 + C_1 x(t) + C_2 (x(t))^2$$  \hspace{1cm} (2.5)

In these expressions, $L_i$, $B_i$ and $C_i$ are modelling constants up to the $i$-th order and $x(t)$ is the time dependent displacement of the voice coil. If we assume for the moment that $B$ and $C$ are linear such that the higher order coefficients in (2.4) and (2.5) are equal to zero, and that in the low frequency range, the inductance is negligible so that the approximation $L(x(t)) = 0$ is valid, then a second order linear transfer function in the Laplace domain can be obtained as [5]:

$$\frac{X(s)}{E(s)} = \frac{B_0 l/R}{s^2 m + s [r_M + (B_0 l)^2 / R] + 1/C_0}$$  \hspace{1cm} (2.6)
A loudspeaker has several sources of nonlinearity which occur in the motor part (i.e. the magnet system and voice coil), in the mechanical part, and due to nonlinear sound radiation. These components have a frequency dependency, however, if we restrict our discussion to the lower frequencies, we only have to take into account those nonlinearities that depend closely on the voice coil excursion [2]. The most prominent nonlinearities corresponding to \( L(x(t)) \), \( B(x(t)) \) and \( C(x(t)) \) are [4]:

- The *electric self inductance* \( L(x(t)) \) which depends on the voice-coil excursion. This is because the voice coil protrudes from the central position, yielding a *reluctance force* \( F_x \) (or back *electromotive force*) proportional to \( i^2(t) \).

- The *force factor* \( B(x(t))l \), which depends on the voice-coil excursion. In the case of constant current drive, the force on the voice coil depends on the position of the coil, since \( \int B \, dl \) is a function of the voice coil displacement. A typical force factor vs. displacement curve is shown in Figure 2.2 a).

- Suspension system nonlinearity. The force vs. displacement curves of the spider and surround are not straight lines and show *hysteresis*. A typical curve is shown in Figure 2.2 b).

Generally, the mechanomotive force in the voice coil is a nonlinear function of the displacement \( x \).

The compliance of the suspension system can be obtained by:

\[
\dot{x}(t) = \frac{x(t)}{f_M(x(t))} = \frac{1}{\alpha + \beta x(t) + \gamma x^2(t)} = \frac{1}{k(x(t))}.
\]

(2.7)

where \( f_M(x(t)) \) represents the force deflection characteristic of the loudspeaker cone suspension system and \( k(x(t)) \) is the nonlinear suspension *stiffness* factor.

Substituting (2.7) into (2.2), we obtain:
\[ B(x(t))l(t) = m \frac{d^2}{dt^2} x(t) + r_M \frac{d}{dt} x(t) + \alpha x(t) + \beta x(t)^2 + \gamma x(t)^3 \] (2.8)

Suspension system nonlinearity manifests itself as soft clipping at the loudspeaker output and results in odd-order harmonics under large signal conditions. As well, there is a frequency dependence. Equations (2.1) and (2.2) show that at high frequencies, the derivatives are large, so that the effect of the nonlinearities is small, i.e., the system is weakly nonlinear. However, at low frequencies, the converse is true and the effect of the nonlinearities is more pronounced. It should also be noted that distortion does not only occur at large signal levels. Significant distortion also occurs at extremely low levels due to unbalanced 2-point suspension, i.e., the surround and the spider [2],[3].

In the HFI domain, it is necessary to use small (i.e., inexpensive) loudspeakers. To obtain reasonably comfortable listening levels in the lower frequencies, excessive diaphragm excursions are needed, which will generate significant distortion products which can be 5% to 10% of the signal amplitude.

![Graphs](image)

**FIGURE 2.2** (a) Typical force vs. displacement curve of a loudspeaker. (b) Typical force vs. displacement curve of the spider.
2.2 Nonlinear Modelling

An equation determining the excursion of the voice coil \( x \) may be constructed by forming a higher order differential equation by substituting (2.3), (2.4), and (2.5) into (2.1) and (2.2) and eliminating the parameter \( i \). The solution to this equation may be attempted by the use of a Volterra series expansion (See [4]), however hysteresis effects cannot be modelled by this method. As well, a loudspeaker response cannot be written as an ordinary power series as is possible for a memoryless (dispersion-free) system, like a network with resistors. For the memoryless case, the Volterra series degenerates into a power series. When dispersive effects are included, the size of the Volterra model can become large.

Alternatively, we may cast the difference equations in state space form [6]. In the equations that follow, \( L(x) = L_0 \) is assumed to be linear. Let the state variables \( x_1 = i, \ x_2 = x \) and \( x_3 = dx_2/dt \). From the equivalent electrical/mechanical circuits, one can obtain the following state-space dynamical equations:

\[
\frac{dx_1}{dt} = \frac{1}{L_0} (-Rx_1 - B_0 lx_3 + e - B_1 lx_2x_3 - B_2 lx_2^2 x_3) \tag{2.9}
\]

\[
\frac{dx_2}{dt} = x_3 \tag{2.10}
\]

\[
\frac{dx_3}{dt} = \frac{1}{m} (B_0 lx_1 - \alpha x_2 - \beta x_2^2 - \gamma x_2^3 + Ib_1 x_1 x_2 + Ib_2 x_1 x_2^2) \tag{2.11}
\]

\[
y(t) = x_2(t) \tag{2.12}
\]

The above equations can be discretized using the Euler approximation,
\[
\frac{dx}{dt}\bigg|_{t=nT} = \frac{x(n+1) - x(n)}{T} \tag{2.13}
\]

where \(T\) is the sampling period and \(x(n)\) is used to denote \(x(nT)\) for convenience. Thus, the following difference equations are formed,

\[
x(n+1) = \begin{bmatrix} a_{11} & 0 & a_{13} \\ 0 & 1 & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} x(n) + \begin{bmatrix} b_1 \\ 0 \\ 0 \end{bmatrix} u(n) + \begin{bmatrix} 0 \\ p_{11}x_2(n)x_3(n) + p_{12}x_2^2(n)x_3(n) \\ p_{31}x_2^2(n) + p_{32}x_2^3(n) + p_{33}x_1(n)x_2(n) + p_{34}x_1(n)x_2^2(n) \end{bmatrix} \tag{2.14}
\]

\[
y(n) = (0, 1, 0)^T x(n) \tag{2.15}
\]

where \(u(n)\) is the system input and \(a_{11}=-TR/L_0, a_{13}=-TB_0/L_0, a_{23}=T, a_{31}=TB_0/Lm, a_{32}=Tc/m, a_{33}=1-TR_M/m, b_1=T/L_0, p_{11}=-TB_1/L_0, p_{12}=-TB_2/L_0, p_{31}=-TB_3/m, p_{32}=-Tc/m, p_{33}=TB_3/Lm, and p_{34}=TB_3/m.\) Knowledge of the associated loudspeaker parameters (suggested in [6]) yield for the simulation:

\[
x(n+1) = \begin{bmatrix} -0.1 & 0 & -0.2 \\ 0 & 1 & 1 \\ 0.6 & -0.5 & -0.15 \end{bmatrix} x(n) + \begin{bmatrix} 0.4 \\ 0 \end{bmatrix} u(n) + \begin{bmatrix} 0 \\ -0.04x_2(n)x_3(n) + -0.05x_2^2(n)x_3(n) \\ 0 \\ -0.08x_2^3(n) + 0.01x_1(n)x_2(n) + 0.02x_1(n)x_2^2(n) \end{bmatrix} \tag{2.16}
\]

\[
y(n) = (0, 1, 0)^T x(n) \tag{2.17}
\]
The sample period $T$ is set to unity and $\beta$ is set to zero since it is small in practice. Figure 2.3 illustrates the input and output signals obtained from the linear and nonlinear models using an input test signal $x(t) = 3.5 \sin (0.6\omega t) + 5.0 \cos (1.6\omega t)$ where $\omega = 100\pi$. It is shown that the nonlinear model exhibits soft-clipping at high excursion peaks. The corresponding impulse response is illustrated in Figure 2.4.

**FIGURE 2.3** Comparison of the input signal $e(t)$ with the linear model and nonlinear model outputs. The nonlinear model exhibits soft-clipping during amplitude peaks.

**FIGURE 2.4** Nonlinear state-space model impulse response.
2.3 Measurement of Nonlinear Parameters

Accurate determination of the nonlinear parameters usually requires that the cone displacement be measured in synchronization with the applied coil voltages or currents. Laser displacement systems are typically required to achieve the accuracy. Once the displacement data is obtained, it may then be applied to a general linear (see for example Ljung [5]) or nonlinear system identification techniques [7]. The simplest method is to measure the lumped parameters as the voice coil is displaced statically and then fit the coefficients of a power series via least squares curve fitting [8]. However, to determine stiffness and hysteresis, a dynamic method is required, where the applied signal is a combination of continuous wave or swept-tone sinusoids. This method is sometimes referred to as the harmonic input excitation method. An alternate method in [9] uses several gaussian noise inputs with different root-mean-square (RMS) amplitudes. The advantage of dynamic methods like harmonic excitation is that a simple microphone can be used to obtain frequencies and relative amplitudes of the harmonic and distortion products. The coefficients of a power series may then be obtained by a set of equations (See for example [8],[9],or [10]) for estimating the nonlinear parameters. Care must be taken however in removing any echo associated with the loudspeaker-microphone measurement system, as discussed in [11].

2.4 Methods for Reducing Nonlinear Distortion

The reduction of loudspeaker distortion may be done using open-loop or closed-loop systems. An open-loop system obtains the nonlinear parameters using one-time measurement techniques described in Section 2.3 and then applies these values in a pre-distortion circuit. Examples of such systems based on the Volterra series can be found in [12][13] and [14], however the later does not compensate for loudspeaker hysteresis effects. A pre-distortion open-loop technique using the mir-
ror filter is described in [15]. An open-loop compensation system using a Matlab® model developed from measured parameters is described in [16]. In [17] an inverse loudspeaker/room model is developed using a Time Delay Neural Network to provide single point room equalization, but no experimental measurements were presented.

A closed-loop system based on the correction of the displacement dependent force factor is described in [18], however, this requires the use of a displacement transducer. Sometimes the loudspeaker itself is used as the sensor for reasons of cost and the cone velocity is detected by a bridge arrangement [19]. The method in [20] uses three operational amplifiers to detect voice coil current to provide negative feedback. Another method based on providing acceleration feedback is described in [21].

2.5 Summary

In this chapter, the nonlinear loudspeaker model is reviewed. A brief description of the electromechanical model is presented and the corresponding differential equations describing the dynamics of motion were introduced. A linear system block diagram shows that the loudspeaker is basically a mass spring system. Next, a review of the primary sources of nonlinear distortion is presented, concluding that the loudspeaker exhibits both temporal distortion (hysteresis) as well as amplitude distortion generated mainly due to nonuniform flux density and suspension system nonlinearity. Simulations based on linear/nonlinear state-space models show that at high signal levels, the loudspeaker exhibits soft-clipping distortion. Finally, a brief literature summary of some techniques available for measuring nonlinear loudspeaker parameters and reducing distortion is presented.
Chapter 3

Review of Nonlinear Adaptive Filtering Techniques

This chapter furnishes the reader with the necessary background theory, techniques, algorithms and structures related to this thesis. It starts with a brief survey of some nonlinear system identification techniques and applications followed by a definition of terms and a quick review of linear models. Subsequently, sections on the stochastic gradient, recursive least squares, and the conjugate gradient search techniques are presented. The chapter concludes with a discussion of the adaptive nonlinear Volterra filter and the multilayer perceptron neural network.

3.1 Applications and a Survey

Linear adaptive filtering techniques are unable to benefit from higher order statistics of a nonlinear process and therefore are limited in scope. For example, conventional linear adaptive filtering algorithms are usually phase blind (Li, [22]) in the sense that they do not respond to phase information contained in a signal in excess of a minimum phase characteristic. Linear models will also fail miserably when trying to correlate two signals with non-overlapping spectral components
(Mathews [23]). In order to exploit the full information context of a signal, it is necessary to invoke nonlinear signal processing techniques. The process of identifying a signal can be viewed as an "understanding" process or as a "learning" process. Understanding a process involves being able to construct a relatively accurate model based on a-priori information, and state-space models are generally of this class (Gershenfeld [24]). Learning a process on the other hand is more effective when there is little a-priori information about the underlying dynamics of a system, but the available input/output information is plentiful. Thus, we can construct an ordered system from a relatively unstructured initial model. Volterra filters and neural networks are two such "learning" structures.

The Volterra series expansion (Schetzen [25]) can model a large class of nonlinear systems and is attractive in adaptive filtering applications because the expansion is a linear combination of nonlinear functions of the input signal. However, the Volterra filter belongs to a class of polynomial filters which have a superlinear increase in the number of parameters for both increasing filter order and polynomial order, and are therefore restricted to applications where the system order is low. In the literature, Volterra filters have been applied to general nonlinear system identification [26],[27],[28],[29], nonlinear echo cancellation for data hybrids [30],[31],[32], nonlinear noise cancellation [33],[34], and estimation and compensation of loudspeaker distortion [13],[14],[35],[36]. However, rarely does the discussion go beyond a 2nd order system.

For compensation of loudspeaker nonlinearity, most distortion products are 3rd order and higher, thus filter polynomial orders greater than three are required to effectively model the speaker transfer function. However, when the loudspeaker output is convolved with a dispersive echo channel, it very quickly leads to an unmanageably huge model. As a result, structures which are "less general" are proposed in the literature. An example of a "weakened" Volterra filter is the cascaded lin-
ear-nonlinear-linear system described by Cowan and Adams [37], which consists of memoryless non-linearity sandwiched between two linear filters (See Figure 3.1). In [37], the nonlinearity takes the form of a Taylor series expansion. Most signal processing problems can be reduced to similar forms by isolating the known nonlinear component and then compensating for its nonlinearity by using a finite number of terms in the expansion. For example, one such system that can be represented by Figure 3.1 is a general satellite communications channel with a nonlinearity introduced by a TWT in the transponder (Namiki [38]).

Neural networks offer an alternative method of dealing with high order system nonlinearities without the "curse of dimensionality" associated with polynomial filters. One of the attractive features of a neural network is its ability to adaptively learn subtle relationships from the data without knowing the underlying process. As well, the network has the ability to generalize, i.e. being able to respond correctly to input patterns not contained in the original training sequence. This is useful in real-world applications where the data is often distorted and incomplete. In the literature, neural networks filters have been used for general nonlinear system identification (Chen [39],[40]) prediction of nonstationary nonlinear speech signals (Haykin [41]), equalization of high power ampli-
fiers in communications systems (Paolo et al. [42],[43]), speech enhancement and noise reduction in hearing aid systems (Knecht [44],[45]), and inverse filtering of loudspeaker-room acoustics (Chang et al. [17]). Neural filters can be constructed by buffering the input in time, as in the Time Delay Neural Network (TDNN) first proposed by Waibel et al. [46]. It is possible to construct similar architectures as shown in Figure 3.1 using a neuron as the memoryless nonlinearity. This form of neural network is called a synaptic FIR neural network and requires the use of a temporal training algorithm if the output error is used to train the network. The FIR temporal training algorithm for neural networks can be traced back to Wan [47] and is an extension original work by Werbos [48].

FIR synaptic neural networks belong to a class of generalized feedforward (GF) structures, which by definition can have either FIR or IIR filters between nonlinear nodes. GF structures employing IIR synapses are called locally recurrent globally feedforward (LRGF) structures (Tsoi [49]). Back and Tsoi [50] have shown that models based on local feedback have better convergence and stability behavior than those based on global feedback.

3.2 Regressors, Mappings and Definitions

The basic adaptive filter structure, be it linear or nonlinear, is illustrated in Figure 3.2 The adaptive filter is assumed to be discrete in nature. In Figure 3.2 the output \( y(n) \) of the adaptive filter is an estimate of the desired signal \( d(n) \) when applied with the input \( x(n) \). A feedforward structure ensures that the output \( y(n) \) is a function of the input data \( x(n) \) only. A well known example of a feedforward structure is the finite impulse response (FIR) filter. A recurrent structure on the other hand will generate an output which is dependent on the input data as well as past values of the output \( y(n) \). Infinite impulse response (IIR) structures are recurrent.
3.2 Regressors, Mappings and Definitions

FIGURE 3.2 An adaptive filter.

If we have observation inputs \( x(n) \) and outputs \( d(n) \) from a dynamical system:

\[
x(n) = [x(n), x(n-1), \ldots, x(1)]^T
\]

(3.1)

\[
d(n) = [d(n), d(n-1), \ldots, d(1)]^T
\]

(3.2)

we can state the general nonlinear relationship between past observations \([x(n-1), d(n-1)]\) and future system outputs \(d(n)\):

\[
d(n) = \phi [x(n-1), d(n-1)] + e(n)
\]

(3.3)

where \(\phi\) is a general nonlinear operator. The additive term \(e(n)\) accounts for the fact that the next output \(d(n)\) will not be an exact function of the past data. If \(e(n)\) is small, we may think of \(\phi [x(n-1), d(n-1)]\) as a good prediction of \(d(n)\), given past data.

The model structure of (3.3) has been found to be too general. It is more useful to construct the nonlinear operator \(\phi\) as a concatenation of two mappings: one that takes the increasing number of past observations \(x(n-1), d(n-1)\) and past outputs \(y(n-1)\) and maps them to a finite dimensional vector \(u(n)\), and one that takes this vector to the output space via a nonlinearity. Hence, the output estimate at time \(n\) is simply:

\[
y(n) = \phi [u(n), w(n)]
\]

(3.4)
where,

\[ u(n) = u(x(n-1), d(n-1), y(n-1)) \]  \hspace{1cm} (3.5)

is called the \textit{regression vector}, and its components are referred to as \textit{regressors}, and the vector \( w(n) \) can be thought of as a \textit{weight vector} operating on regression vector \( u(n) \) with a length equal to the choice of regression vector. The choice of nonlinear mapping in (3.4) is thus decomposed into two partial problems for dynamical systems:

1. Selection of the regression vector \( u(n) \) from a finite value of past inputs and outputs.
2. Selection of the nonlinear mapping \( \varphi \) from the regressor space to the output space.

Nonlinear mappings come in a variety of flavors including tensor products, radial basis functions, fuzzy networks, sigmoidal neural networks and wavelets. The ones that are considered in this thesis are:

- \textit{Tensor Product}: The regression vector is put into a \textit{state-expander}, which maps the inputs to a large number of outputs, depending on the size and order of nonlinearity desired. The Volterra and Taylor series expansions are well known examples.

- \textit{Sigmoid Functions}: For the neural networks, a number of possible nonlinear mappings exist, commonly referred to as \textit{activation functions}. A well known mapping is the unit step function.

An alternative to the sigmoid function is the \textit{hyperbolic tangent} activation function, which can output \textit{bipolar} values between 1 and -1:

\[ \varphi(v) = \tanh(av) \]  \hspace{1cm} (3.6)

where \( a \) is a slope parameter, usually set to one. Various combinations of linear and nonlinear functions can also be constructed to obtain tailor-made activation functions (see for example [51] and [52]).
3.3 Linear Models

Multi-input single-output linear structures used in practice can all be summarized by the general family (Ljung [7]):

\[
A(q) d(n) = \frac{B_1(q)}{F_1(q)} x(n-k_1) + \ldots + \frac{B_m(q)}{F_m(q)} x(n-k_m) + \frac{C(q)}{G(q)} e(n)
\]

where,

\[
A(q) = 1 + a_1 q^{-1} + \ldots + a_{n_a} q^{-n_a}
\]

\[
B_j(q) = b_{0,j} + b_{1,j} q^{-1} + \ldots + b_{n_{b,j}} q^{-n_{b,j}}
\]

\[
F_j(q) = 1 + f_{1,j} q^{-1} + \ldots + f_{n_{f,j}} q^{-n_{f,j}}
\]

\(m\) is the number of exogenous inputs, \(q^{-1}\) is the unit delay operator, and \(C(q), G(q)\) have similar forms to \(A(q)\). The simplest linear dynamical model is the FIR model, sometimes referred to the moving average (MA) model:

\[
d(n) = y(n) + e(n) = B(q) x(n-1) + e(n) = b_0 x(n-1) + b_1 x(n-2) + \ldots + b_{n_b} x(n-n_b - 1) + e(n)
\]

The various models associated with (3.7) are basically variants of (3.11) using different ways of picking up "poles" of the system and different ways of describing the noise characteristics. A list
of the special cases of the general family is illustrated in Table 3.1.

**TABLE 3.1 Special cases of the general linear model.**

<table>
<thead>
<tr>
<th>Name</th>
<th>Acronym</th>
<th>Polynomials</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moving Average</td>
<td>MA</td>
<td>A = C = G = F = 1</td>
</tr>
<tr>
<td>AutoRegressive</td>
<td>AR</td>
<td>F = C = G = 1, B = 0</td>
</tr>
<tr>
<td>AutoRegressive with eXogenous input.</td>
<td>ARX</td>
<td>F = C = G = 1</td>
</tr>
<tr>
<td>Autoregressive Moving Average</td>
<td>ARMA</td>
<td>G = F = 1, B = 0</td>
</tr>
<tr>
<td>Autoregressive Moving Average with eXogenous input</td>
<td>ARMAX</td>
<td>G = F = 1</td>
</tr>
<tr>
<td>AutoRegressive ARX</td>
<td>ARARX</td>
<td>C = F = 1</td>
</tr>
<tr>
<td>AutoRegressive ARMAX</td>
<td>ARARMAX</td>
<td>F = 1</td>
</tr>
<tr>
<td>Output Error</td>
<td>OE</td>
<td>A = C = G = 1</td>
</tr>
<tr>
<td>Box-Jenkins</td>
<td>BJ</td>
<td>A = 1</td>
</tr>
</tbody>
</table>

The predictor associated with linear models can be given in *pseudo-linear regression* form as:

\[ y(n) = w^T(n) u(n) \] (3.12)

The regressors typically associated with \( u(n) \) for the systems studied in this thesis are:

- \( x(n-k) \) (associated with the \( B \)-polynomial)
- \( d(n-k) \) (associated with the \( A \)-polynomial)

Linear *State-Space* (SS) models can be described as a pseudo-linear regression by constructing a transfer function from the state variables. For more details see Johns *et al.*[53].

The *multi-input ARX* model with two inputs is equivalent to the *Equation-Error* (EE) IIR formulation. The *ARMAX* is one of a family of more sophisticated EE model structures including *ARARX* and *ARARMAX* which utilize additional filters for the error signal, however, since they involve a greater number of coefficients they are not considered further.
3.4 Search Methods and Algorithms

The weights in an adaptive filter are adjusted by an algorithm that minimizes some function of the error $e(n)$ between the desired signal $d(n)$ and the filter output $y(n)$. A general form for filter parameter adaptation, for minimization of the cost function $J$ is stated below [54]:

$$w(n+1) = w(n) - \mu(n) P(n) \nabla_w(J)$$  \hspace{1cm} (3.13)

where:

- $w(n)$ and $w(n+1)$ are the weight vector parameter estimates at time $n$ and $n+1$.
- $\mu(n)$ is a bounded step size.
- $P(n)$ is a matrix obtained from input values that is used to improve the convergence rate.
- $J$ is a function of the prediction error.
- $\nabla_w(J)$ is the gradient of the cost function $J$ with respect to the parameter $w$, defined as:

$$\nabla_w(J) = \frac{\partial J}{\partial w}$$  \hspace{1cm} (3.14)

**Gradient Search Method.** In the gradient search method $J$ is the mean square error (MSE) cost function defined as:

$$J = E\{e^2(n)\}$$  \hspace{1cm} (3.15)

where $E$ is the statistical expectation operator and $e(n) = d(n) - y(n)$. It can be shown (see Widrow [55]) that the optimum weight vector $w_{opt}$ minimizes $J$ and can be obtained by solving the Wiener-Hopf equation.

---

1. Convergence rate is defined as the rate at which the MSE approaches the minimum value during the training phase of adaptation.
\[ E[\mathbf{u}(n)\mathbf{u}^T(n)]\mathbf{w}_{opt} = E[\mathbf{d}(n)\mathbf{u}(n)] \quad (3.16) \]

The corresponding minimum mean square error (MMSE) at \( \mathbf{w}_{opt} \) equals,

\[ \varepsilon_{min} = E[\mathbf{d}^2(n)] - E[\mathbf{d}(n)\mathbf{u}(n)]^T E[\mathbf{u}(n)\mathbf{u}^T(n)]^{-1} E[\mathbf{d}(n)\mathbf{u}(n)] \quad (3.17) \]

Gradient based learning algorithms for FIR and IIR structures are presented in Section 3.4.1 and Section 3.4.2.

**Least Squares Search Method.** The least squares search method minimizes the sum-squared error (SSE) cost function:

\[ f(n) = \sum_{i = i_1}^{i_2} \lambda^{n-i}e^2(n) \quad (3.18) \]

where \( i_1 \) and \( i_2 \) refer to the index limits over which the cost function is obtained and \( \lambda \) is a forgetting factor, between 0 and 1. Recursive Least Squares (RLS) learning algorithms applicable to AEC's are presented in Section 3.4.3.

**Conjugate Gradient Search Method.** The conjugate gradient (CG) algorithm updates the tap weights of a filter structure with new directions that are "non-interfering", in other words, conjugate to each other. More importantly, the CG algorithm can be applied to both linear and nonlinear systems as a method of obtaining improved convergence. The CG learning method is covered in Section 3.4.4.
3.4.1 Gradient Learning Algorithms for FIR Structures

The regression vector \( \mathbf{u}(n) \) of an FIR structure contains only \( x \) inputs, hence we define:

\[
\mathbf{u}(n) = \mathbf{x}(n) = [x(n), x(n-1), \ldots, x(n-n_p)]^T
\]  
(3.19)

\[
\mathbf{w}(n) = [b_0(n), b_1(n), \ldots, b_{n_p}(n)]^T
\]  
(3.20)

By replacing the expectation operator in the true gradient expression in (3.14) with an approximation based on the instantaneous error, the gradient now becomes:

\[
\nabla_{\mathbf{w}}(J) = \frac{\partial}{\partial \mathbf{w}} (e^2(n)) = 2e(n) \frac{\partial}{\partial \mathbf{w}} [d(n) - y(n)]
\]

\[
= -2e(n) \frac{\partial}{\partial \mathbf{w}} [\mathbf{w}^T(n) \mathbf{x}(n)] = -2e(n) \mathbf{x}(n)
\]  
(3.21)

Substitution of (3.21) into (3.13) we obtain the general form of the accelerated steepest descent algorithm [54].

\[
y(n) = \mathbf{w}^T(n) \mathbf{x}(n)
\]

\[
e(n) = d(n) - y(n)
\]

\[
\mathbf{w}(n+1) = \mathbf{w}(n) + \mu(n) e(n) \mathbf{P}(n) \mathbf{x}(n)
\]  
(3.22)

The factor of 2 in (3.21) has been absorbed into the step size value \( \mu(n) \). The matrix \( \mathbf{P}(n) \) is chosen based on some a priori knowledge to improve the convergence.

**LMS Algorithm.** If \( \mathbf{P}(n) \) is selected as the identity matrix, and the step size is fixed in the general formula given in (3.22), then we obtain the least-mean-square (LMS) algorithm. The LMS filter is stochastic in that it provides an approximation to the Weiner filter as it converges [55]. The tap weight update formula for the LMS algorithm is:
\[
    w(n+1) = w(n) + \mu e(n) x(n)
\] (3.23)

The complete algorithm is described in Appendix D.1.

*Normalized LMS (NLMS) algorithm.* In the NLMS algorithm, the step size is normalized by the Euclidean norm of the input vector \( x(n) \). This algorithm is appropriate when the input power is unknown or highly variable and is a benchmark standard for AEC applications. The complete algorithm is described in Appendix D.2.

*Variable Step Size (VSS) algorithm.* Variable step size (VSS) LMS-based algorithms use a large step size when the filter parameters are far from the optimum to achieve fast convergence, and a small step size when the weights are close to the optimum. The VSS algorithm used in this thesis is the Modified VSS algorithm (MVSS) as described by Mayyas [56]. The advantage of the MVSS algorithm over the standard VSS algorithm described in [57] is its relative insensitivity to noisy signals due to the time average autocorrelation process. The complete algorithm is listed in Appendix D.3.

### 3.4.2 Gradient Learning Algorithms for IIR Structures

There is a rich body of literature on IIR structures that includes direct form [54],[58],[59], lattice form [60], parallel [61], and cascaded [62] structures. Fundamentally there have been two approaches to adaptive IIR filtering that correspond to different formulations of the prediction error. The two forms are *equation error (EE)* and *output error (OE)* methods.

The difference between the *OE* and *EE* IIR structures in a system identification context is illustrated in Figure 3.3. Each method has its own advantages and disadvantages. The *EE* model gives rise to a *unimodal error surface* with easy stability monitoring, but when the plant signal is con-
taminated with noise $\sigma^2_n$, the resulting parameter estimates are biased away from the optimum values which give the MMSE [63]. The OE has no such bias, however the error surface is *multimodal* and stability monitoring is non-trivial. As a result, the parameter estimates may converge to a *local minimum* and not necessarily the *global minimum*. Various methods exist which combine the ben-

![Diagram](image)

**FIGURE 3.3** *EE and OE model structures in system identification.*

eficial properties of the OE and EE methods to provide composite algorithms. These methods are generally referred to *composite error surface (CES)* methods. A good example of a CES is the *Steiglitz-McBride Method (SMM)* [64]. Other examples of CES methods include the *composite regressor algorithm (CRA)* [65] and the *composite gradient algorithm* [66]. Still other methods exist for removing the bias associated with the EE method altogether [67], however, these are considered beyond the scope of this thesis.

**Equation Error Formulation**. The EE formulation is characterized by the multi-input *nonrecursive ARX* difference equation with two inputs:
\[ y(n) = \sum_{i=1}^{n_a} a_i(n) d(n-i) + \sum_{i=0}^{n_b} b_i(n) x(n-i) \quad (3.24) \]

Note that the output \( y(n) \) depends only on the two inputs \( x(n) \) and \( d(n) \), consequently the weights can be updated using well understood LMS type algorithms. The complete algorithm for the EE-IIR structure is listed in Appendix D.6.

**Output Error Formulation:** The OE formulation can be described using the recursive difference equation:

\[ y(n) = \sum_{i=1}^{n_a} a_i(n) y(n-i) + \sum_{i=0}^{n_b} b_i(n) x(n-i) \quad (3.25) \]

Notice that the output is now dependent on previous outputs \( y(n-i) \). The OE-IIR algorithm and derivation of the update equations using the simplified gradient is listed in Appendix D.7.

### 3.4.3 Recursive Least Squares Learning Algorithms

**Exponential Recursive Least-Square (RLS) Algorithm.** In the general parameter update equation (3.13), if \( P \) approximates \( R^{-1} \), where \( R \) is the true autocorrelation matrix of the input vector \( x(n) \), defined by:

\[ R = E[x(n)x^T(n)] \quad (3.26) \]

and the step size parameter \( \mu(n) \) is replaced by:

\[ \mu(n) = \frac{1}{1 + q(n)} \quad (3.27) \]

where
\[ q(n) = \lambda^{-1}x^T(n)P(n-1)x(n) \] (3.28)

is a measure of the signal input power, with a normalization introduced by \( P(n-1) \), then the convergence time of the algorithm can be reduced substantially even in cases of high eigenvalue spread. This is the basis for the RLS algorithm (\( \lambda = 1 \)) which is a special case of the Kalman filter. The forgetting factor \( \lambda \) reduces the effects of past data over time and therefore improves the ability of the algorithm to track variations in the statistics of the input data. However, this imposes a stability limit on the minimum value of \( \lambda \). The RLS algorithm for an FIR structure is described further in Appendix D.3. Note that the RLS update may also be applied to the OE-IIR structure as is described in Appendix D.4.

The important property of the RLS algorithm is its fast convergence. When the input is time-invariant, the MSE in the RLS algorithm converges in about \( 2M \) iterations where \( M \) is the filter order. The RLS algorithm demands a high computational load compared to the LMS algorithm, requiring \( O(M^2) \) multiply-adds for each update, and therefore "fast" versions of the RLS (i.e. computationally efficient) have been implemented, as discussed below.

**Fast Transversal Filter (FTF) algorithm.** The FTF algorithm takes advantage of the redundancies in the standard RLS algorithm to bring the computational load down to \( O(7M) \) operations per iteration update. The FTF algorithm suffers from numerical precision problems and is generally viewed as unstable. A method of stabilizing the algorithm, known as the Stabilized Fast Transversal Filter (SFTF) mitigates the problem but increases the complexity from \( O(7M) \) to \( O(8M) \) operations per iteration. A method of improving the tracking capability of the SFTF is by the use of a time varying acceleration factor \( \rho(n) \) which effectively modifies the time constants of the algorithm such that the effective forgetting factor \( \lambda_{\text{eff}} \) varies between \( \lambda \) (when \( \rho = 0 \)) and 0 (when \( \rho = 1 \)).
The accelerated SFTF algorithm used in this thesis was proposed in [68] and [69] and is described further in Appendix D.5.

### 3.4.4 Conjugate Gradient Learning Algorithms

The conjugate gradient algorithm is obtained by iteratively constructing successive direction vectors that are mutually conjugate and linearly independent as the method progresses (Hestenes [70]). Thus, the directions are determined sequentially at each step of the iteration. At step $k$, one evaluates the current negative gradient vector and adds to it a linear combination of the previous direction vectors to obtain a new conjugate direction vector along which to move.

The CG algorithm has convergence properties that will minimize a quadratic function $f(x)$ of $m$ variables (i.e., weights) in no more than $m$ iterations and provide fast convergence. The function $f(x)$ is defined as

$$ f(x) = \frac{1}{2} x^T Q x - b^T x $$

(3.29)

where $x$ is of size $m$ and $Q$ is of size $m \times m$. The approach in the conjugate direction method is to obtain a set of linearly independent direction vectors $d_0, d_1, \ldots, d_{m-1}$ which are conjugate with respect to $Q$ such that equation (3.29) is minimized. The solution is obtained when $g = Q x - b = 0$, i.e.,

$$ Q x_{opt} = b $$

(3.30)

The vectors $d_0, d_1, \ldots, d_{m-1}$ are said to be $Q$-conjugate if

$$ d_i^T Q d_j = 0, \quad i \neq j $$

(3.31)
Conjugate Direction Coefficients $\alpha_k$. The optimum solution vector $\mathbf{x}_{opt}$ minimizes (3.29) where $\mathbf{x}_{opt}$ can be expressed as:

$$
\mathbf{x}_{opt} = \alpha_0 \mathbf{d}_0 + \alpha_1 \mathbf{d}_1 + \ldots + \alpha_{m-1} \mathbf{d}_{m-1}
$$

(3.32)

and the constants are given by (Luenberger [71]):

$$
\alpha_k = \frac{\mathbf{g}_k^T \mathbf{d}_k}{\mathbf{d}_k^T \mathbf{Q} \mathbf{d}_k}
$$

(3.33)

where $\mathbf{g}_k = \mathbf{Q} \mathbf{x}_k - \mathbf{b}$. The conjugate gradient algorithm determines the appropriate orthogonal set of direction vectors and constants $\alpha_k$ iteratively and generates a new $\mathbf{x}_k$ according to:

$$
\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k
$$

(3.34)

If the direction vectors $\mathbf{d}_k$ are mutually conjugate and linearly independent, then the initial guess $\mathbf{x}$ will converge to the optimum $\mathbf{x}_{opt}$ after $m$ steps, that is $\mathbf{x}_m = \mathbf{x}_{opt}$.

Conjugate Direction Vectors $\mathbf{d}_k$. The initial direction vector is chosen as the negative gradient at the initial point: $\mathbf{d}_0 = -\mathbf{g}_0$. Successive directions are obtained from a linear combination of the current gradient and the previous direction:

$$
\mathbf{d}_{k+1} = -\mathbf{g}_{k+1} + \beta_k \mathbf{d}_k
$$

(3.35)

Given that successive directions must be $\mathbf{Q}$ conjugate, i.e.:

$$
\mathbf{d}_{k+1}^T \mathbf{Q} \mathbf{d}_k = \mathbf{[} -\mathbf{g}_{k+1} + \beta_k \mathbf{d}_k \mathbf{]}^T \mathbf{Q} \mathbf{d}_k = 0
$$

(3.36)

gives the update equation for $\beta_k$ as
\[
\beta_k = \frac{g_{k+1}^T Q d_k}{d_k^T Q d_k}
\]  

(3.37)

**Extension to Nonquadratic Problems.** The *quadratic approximation* method extends the CG algorithm to general *nonlinear* functions by interpreting (3.29) as a second order Taylor series expansion of the objective function. For details the reader is referred to papers by Johansson *et al.* [72], Charalambous [73], and Boray and Srinath [74]. Essentially, the following associations are made at \( x \):

\[
g_k \leftrightarrow \nabla f(x_k) \quad Q \leftrightarrow F(x_k)
\]  

(3.38)

\( F_k(x_k) = \nabla^2 f(w_k(n)) \) is the \( m \times m \) Hessian matrix of the function at \( x_k \), and \( g_k \) is the \( m \times 1 \) gradient of the function at \( x_k \).

Since the calculation of the hessian \( F \) of a matrix is \( O(m^3) \) complexity [75], avoiding the calculation of \( F \) would be advantageous for large order systems. It can be shown (see [74] and p. 138 [70]) that \( Qd_k = -Qg_k = p_k - g_k \) where \( p_k = \nabla f(y_k) \) is the gradient at \( y_k = x_k - g_k \).

Hence, the update equation for the direction constants can be given by

\[
\alpha_k = -\frac{d_k^T g_k}{d_k^T Q d_k} = \frac{g_k^T d_k}{d_k^T (p_k - g_k)} = \frac{g_k^T d_k}{d_k^T (g_k - p_k)}
\]  

(3.39)

In addition, \( Q \) may be eliminated from the expression for \( \beta_k \) to yield [72]:

\[
\beta_k = \frac{g_{k+1}^T [g_{k+1} - g_k]}{d_k^T [g_{k+1} - g_k]} = \frac{g_{k+1}^T [g_{k+1} - g_k]}{g_k^T g_k} = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}
\]  

(3.40)

The first expression is the Hestenes-Stiefel formula for \( \beta_k \) and the second and third expressions are
referred to as the Polak-Ribière and Fletcher-Reeves formulas respectively.

Using the above expressions, we may obtain a recursive formulae for computing successive $\alpha_k$ and $\beta_k$ that do not require the computation of $F$. However, the method is based on the Taylor approximation of a general nonlinear function and the Q-conjugacy of the direction vectors will deteriorate as the method proceeds. Hence, it is common practice to reinitialize $d_k$ to $g_k$ at every $n$th iteration.

**Partial Conjugate Gradient Method.** The partial conjugate gradient method carries out the conjugate gradient procedure for $k<m-1$ steps [71]. The special case of $k=0$ corresponds to the standard method of steepest descent, while $k=m-1$ corresponds to the full conjugate gradient method. Thus, choosing $k<m$ results in reduced complexity. If equations (3.39) and (3.40) are used to compute the optimum direction and step size, we do not need to compute the Hessian $F$. The penalty is that two gradient calculations must be performed per iteration, one at the current value of the vector $x_k$ and one at $y_k$. However, since the computation of the Hessian matrix is of order $O(m^3)$ and the calculation of a single gradient is of order $O(m^2)$, the savings are substantial if the filter order $m$ is large and if $k<m$.

We now replace the dependent variable $x_k$ given above with a set of weights $w_k(n)$ we obtain the following algorithm, which is based on the method of Partial CG, that does not require the use of a line search or the calculation of the Hessian matrix. Given an initial weight estimate $w_0$ of length $m$, the CG algorithm generates the sequence of new weights $w_1, w_2, ...$ for the network $\phi(u(n), w(n))$ using the following scheme. Note that $n$ refers to the time index and $k$ refers to the conjugate direction count in the sequel.
Conjugate Gradient Algorithm:

Initialization: \( w_0(0) = 0 \)

For each iteration \( n \), do steps 1, 2 and 3.

Step 1. a) Starting with an initial weight vector \( w_0(n) \) compute the following:

\[
g_0(n) = [\nabla f(w_0(n))]^T
\] (3.41)

\[
y_0(n) = w_0(n) - g_0(n)
\] (3.42)

\[
p_0(n) = [\nabla f(y_0(n))]^T
\] (3.43)

b) set \( d_0(n) = g_0(n) \)

Step 2. Repeat for \( k = 0, 1, \ldots, m-1 \)

a) set \( w_{k+1}(n) = w_k(n) + \alpha_k d_k(n) \) where \( \alpha_k \) is the optimum step size:

\[
\alpha_k = \frac{g_k(n)^T d_k(n)}{d_k(n)^T (g_k(n) - p_k(n))}
\] (3.44)

b) Compute the gradients at the new weight vector \( w_{k+1} \)

\[
g_{k+1}(n) = [\nabla f(w_{k+1}(n))]^T
\] (3.45)

\[
y_{k+1}(n) = w_{k+1}(n) - g_{k+1}(n)
\] (3.46)

\[
p_{k+1}(n) = [\nabla f(y_{k+1}(n))]^T
\] (3.47)

(c) Unless \( k = m-1 \), obtain the new direction vector

\[
d_{k+1}(n) = -g_{k+1}(n) + \beta_k d_k(n)
\] (3.48)
where \( \beta_k = \frac{g_{k+1}^T(n) \cdot g_{k+1}(n)}{g_k^T(n) \cdot g_k(n)} \) (3.49)

and repeat Step 2 (a).

Step 3. Replace \( w_i(n) \) by \( w_m(n) \) and go back to Step 1.

\( \beta_k \) gives a measure of the rate of change of successive gradients. If \( \beta_k > 1 \), then the magnitude of a successive gradient vector is not decreasing, meaning that the minimum has been reached. \( \beta_k > 1 \) is a termination condition for conjugate direction \( k \) in Step 2. The calculation of \( \beta_k \) is done according to the Fletcher-Reeves method rather than the Polak-Ribiere method [71] since it tends to give a smoother convergence.

The CG algorithm can be computationally expensive if real time processing is required, thus simplification techniques exist which can reduce the complexity substantially. This is covered in more detail in Chapter 6.

### 3.5 Nonlinear Models

In the nonlinear domain, \( \varphi \neq 1 \). Following the nomenclature for the linear models, it is natural to coin similar names for nonlinear models, for example [76]:

- **NFIR-models**, which use only \( x(n-k) \) as regressors. Examples include Volterra filters and feedforward neural networks. These are covered in Section 3.6 and Section 3.7.

- **NARX-models**, which use \( x(n-k) \) and \( d(n-k) \) as regressors. Narendra and Parthasarathy [77] present a NARX neural network which they refer to as a series-parallel model. A method of selecting the parameters in a neural network NARX model is given by Urbani [78].
• *NOE*-models. For example, Narendra and Parhasarathy [77] present a *NOE* neural network which they refer to as a *parallel* model.

• *NARMAX*-models. For example Chen and Billings [79] present a general paper on nonlinear identification using the *NARMAX* model. Jang and Kim [9] present a *NARMAX* specifically for identifying a nonlinear loudspeaker.

• *Nonlinear State-Space (NSS)* models, that use past components of virtual outputs, i.e. signal values at internal nodes of the network that do not correspond to the output variable. For examples see Gao, [80] and [81].

The adaptive *bilinear* structure [23] [82] can be considered as an NOE or NARX model depending on whether the inputs include \( y(n) \) or \( d(n) \). Despite its simplicity, the bilinear model is an important nonlinear model since it can be shown (Mohler [83]) that a large class of nonlinear systems can be approximated with arbitrary precision using truncated bilinear system models. However, like the EE method, it produces a biased MMSE if \( d(n) \) is noisy.

Next, a brief review of the Volterra filter and neural network filter structures is presented.

### 3.6 The Volterra Filter

The Volterra filter is a general nonlinear feedforward structure that has been successfully applied to identifying low order nonlinear systems. The Volterra filter is a *polynomial* structure with an output that results from a summation of homogenous systems each with consecutive degrees. A homogenous system of degree \( k \) yields an output \( a^k y_k(n) \) for an input \( ax(n) \) where \( y_k(n) \) is the response to \( x(n) \). The output \( y(n) \) of a nonlinear process can be approximated by a truncated Volterra series:
3.6 The Volterra Filter

\[ y(n) = h_0 + \sum_{m_1=0}^{N_1} h_1(m_1) x(n-m_1) + \sum_{m_1=0}^{N_1} \sum_{m_2=0}^{N_2} h_2(m_1, m_2) x(n-m_1) x(n-m_2) + \ldots \]

\[ \sum_{m_1=0}^{N_p} \sum_{m_2=0}^{N_p} \ldots \sum_{m_p=0}^{N_p} h_p(m_1, m_2, \ldots, m_p) x(n-m_1) x(n-m_2) \ldots x(n-m_p) \]

(3.50)

where \( h_p(m_1, m_2, \ldots, m_p) \) denote the so called \( p \)-th order Volterra kernels of the system and \( N_1, \ldots, N_p \) represent the orders of the nonlinear sections. Note that in equation (3.50) \((m_1, m_2, \ldots, m_p)\) are limited to \( N_p \) for the \( p \)-th order section where it is assumed that \( N_1 > N_2 > \ldots N_p \), whereas for conventional truncated Volterra filter, \( m_i \) ranges from 0 to \( N_i \). For the purposes of this discussion, we have dropped the time dependence on \( n \). We will also assume that the kernels \( h_2(m_1, m_2), h_3(m_1, m_2, m_3), \ldots, h_k(m_1, m_2, \ldots, m_k) \) are symmetric, i.e. that the indexes are exchangeable. Define the \( p \)-th order Volterra kernel vectors and \( p \)-th order input space regression vectors at time \( n \):

\[ h_p(n) = [h_1(m_1, m_1, \ldots, m_1), h_2(m_1, m_1, \ldots, m_2), \ldots, h_p(m_p, m_p, \ldots, m_p)]^T \]

(3.51)

\[ x_p(n) = x_1(n) \otimes x_{p-1}(n) \]

(3.52)

where \( \otimes \) is the Kronecker product of vectors and it is assumed that the duplicate terms have been removed. We may rewrite (3.50) as a vector product of an extended weight vector and regressor:

\[ y(n) = h_0 + h_1^T(n) x_1(n) + \ldots + h_p^T(n) x_p(n) = h_e^T(n) x_e(n) \]

(3.53)

where

\[ h_e^T(n) = [h_0^T(n), h_1^T(n), \ldots, h_p^T(n)] \]

(3.54)
\[ x_e^T(n) = [1, x_1^T(n), \ldots, x_p^T] \] (3.55)

We can see from (3.53) that the first term is the DC component of the output, and the subsequent terms are the linear, quadratic, cubic, up to the \( p \)-th order polynomial components of the output respectively, and that for \( p = 1 \), we obtain the linear FIR filter. The structure is illustrated in Figure 3.4.

**FIGURE 3.4** An adaptive Volterra filter structure consists of a number of homogenous systems operating on extended input vectors. The extended inputs are calculated using the tensor product nonlinear mapping strategy.
3.6.1 Performance Surfaces

There is a super-linear increase in the eigenvalue spread of the performance surface for nonlinear models, which limits the convergence when gradient search techniques are used to update the tap weights. RLS versions of the Volterra algorithm can be used to mitigate this problem (See for example Mathews [23]), however, for large filter orders the complexity is prohibitive. Figure 3.5 (a) and (b) show the structure of a linear and nonlinear filter, each consisting of two taps. The performance surfaces for the two models are plotted in Figure 3.6.

![Diagram](image1)

(a) Linear filter with two weights. (b) Nonlinear filter with two weights.

![Diagram](image2)

(a) Linear model. (b) Nonlinear model has elliptical performance surface.
The unknown system to be identified has the same structure, with optimum weights chosen as \( w_1=1, \ w_2=2 \). It can be seen from the results that the performance surface for the nonlinear model has an elongated bowl shape, which will result in an increased eigenvalue spread for this example.

### 3.6.2 Adaptive Learning Algorithm

Analogous to linear filter theory, the optimum coefficients solve the *extended* Wiener-Hopf equations:

\[
E[\mathbf{x}_e(n)\mathbf{x}_e^T(n)]\mathbf{h}_e(n) = E[\mathbf{d}(n)\mathbf{x}_e(n)]
\]  
\[
(3.56)
\]

where the subscript \( e \) refers to the extended vector. The corresponding LMS adaptive updating of the \( p \)-th order coefficients from the \( p \)-th state expander section is performed according to:

\[
\mathbf{h}_p(n+1) = \mathbf{h}_p(n) + \mu_p \mathbf{e}(n)\mathbf{x}_p(n)
\]  
\[
(3.57)
\]

where \( \mu_p \) is the step size for the \( p \)-th power term. The corresponding Volterra minimum MSE can be computed from:

\[
\varepsilon_{min} = E[\mathbf{d}^2(n)] - E[\mathbf{d}(n)\mathbf{x}_e(n)]^T E[\mathbf{x}_e(n)\mathbf{x}_e^T(n)]^{-1} E[\mathbf{d}(n)\mathbf{x}_e(n)]
\]  
\[
(3.58)
\]

which includes the linear MMSE in the case \( p=1 \). A summary of the LMS-Volterra algorithm is given in Appendix D.8.

The advantages of using the Volterra filter is that linear adaptive filter theory can be applied to the extended vectors for on-line adaptation. As well, the MSE surface space does not contain any local minima, because the filter output depends linearly on the extended coefficients. However, the disadvantages are that the dimension of the extended input vector \( \mathbf{x}_e(n) \) becomes very large for larger filter orders, and thus slows the convergence time. The dimension of the extended vector \( \mathbf{x}_e(n) \) can
be calculated from the following formula:

$$dim(\mathbf{x}_e) = dim(\mathbf{h}_e) = \sum_{l=0}^{p} \binom{N_l + l - 1}{l}$$ (3.59)

For example, a 3rd order Volterra system with $N_1 = N_2 = N_3 = 50$ will have 23,425 elements in $\mathbf{h}_e(n)$, and this assumes that combinations such as $h_2(2,3)$ and $h_2(3,2)$ are counted as one value! With this in mind, it is important to weed out as many non-essential terms as possible, in order to simplify the structure.

**IIR and higher order systems.** Several new methods have recently been proposed to model general higher order systems using lower order subsystems. For example, a parallel cascaded truncated Volterra system has been proposed in [84] as a method to combine lower order systems to approximate a much higher order Volterra system. Also, a recent paper [85] presents a second order Volterra infinite impulse response (IIR) structure for modelling sinusoidal harmonics, however, when applied to real-world data consisting of engine and vibration signals, the net gains were marginal compared to the second order Volterra finite impulse response (FIR) structure.

**Stability and Convergence.** It can be shown in linear adaptive theory that the values of the coefficients converge if the step sizes $\mu_p$ are chosen such that $0 < \mu_p < 2/\lambda_{max}$ where $\lambda_{max}$ is the maximum eigenvalue of the autocorrelation matrix of the extended input vector $\mathbf{x}_e(n)$. The problem for nonlinear filtering is that the eigenvalues spreads are in general very large. *Even when the input signal is white, the presence of nonlinearities in the input vector will cause the eigenvalue spread to be more than one.* Consequently, algorithms and structures that have convergence behaviors that are independent of (or less dependent on) the statistics of the input signal are often used. It is possible to apply the RLS update operator to the extended weights (Mathews [23]), however there is a
corresponding increase in complexity. An alternate method for improving the convergence rate using the LMS update is the variable step size algorithm for Volterra filters, proposed by Sicuranza [86].

3.7 Multilayer Perceptron Neural Networks

A multilayer perceptron (MLP) neural network can solve certain complex problems more accurately than linear techniques if the underlying physical mechanism responsible for the process output is inherently nonlinear. Neural networks may also be more applicable than polynomial filters to real-life systems since rarely in nature does the output of a nonlinear system increase without bound for increasing inputs. Most physical systems will exhibit some form of clipping or limiting before this happens.

The neuron constructs a nonlinear mapping from the regression space to the output space via a nonlinear activation function. There are an infinite number of possible candidate activation functions, however, the most widely used are the linear model, the McCulloch-Pitts model characterized by a threshold function, the piece-wise linear model, characterized by a linear function which is clipped beyond a defined linear range, and the sigmoid function.

Consider a multilayer feedforward neural network filter as shown in Figure 3.7. The activation level at the input to the sigmoidal nonlinearity of neuron $j$ in layer $l+1$ is:

$$s_{j}^{(l+1)}(n) = \sum_{i=0}^{N_l} w_{ij}^{(l)}(n) x_{i}^{(l)}(n) \quad (3.60)$$

where $x_{i}^{(l)}(n)$ represents the output from the previous layer and is the input to weight elements $w_{ij}^{(l)}(n)$ at time $n$ and $N_l$ is the number of nodes in layer $l$. The output of node $j$ in layer $l$ is:
\[ x_j^{(l)}(n) = \varphi(s_j^{(l)}(n)) \]  

(3.61)

where \( \varphi \) represents the nonlinear activation function. The MLP output for node \( j \) in the output layer is

\[ y_j(n) = x_j^{(L)}(n) \]  

(3.62)

### 3.7.1 Backpropagation Learning Algorithm

The basic mechanism behind supervised learning rules is to update the network weights and bias terms until the MSE between the network output \( y \) and desired target signal \( d \) is minimized to below a predetermined level. The backpropagation (BP) algorithm [87] is a supervised learning algorithm based on propagating errors through to hidden nodes using an instantaneous gradient estimate. For the on-line training mode, the instantaneous cost function \( J_{\text{inst}} \) at time \( n \) is defined as:
3.7 Multilayer Perceptron Neural Networks

\[ J(n) = J_{\text{inst}}(n) = \frac{1}{2} e^2(n) = \frac{1}{2} \sum_{i=1}^{N_L} e_i^2(n) \]  \hspace{1cm} (3.63)

where:

\[ e_i(n) = d_i(n) - y_i(n) \]  \hspace{1cm} (3.64)

is the error signal at time \( n \) and \( d_i(n) \) and \( y_i(n) \) represent the output and desired signals respectively for the output of a neuron \( i \). \( N_L = 1 \) for a network with a single output. The backpropagation algorithm attempts to minimize the \( J \) by the delta rule [88] for the vector \( \mathbf{w} \) by incrementing at each step toward the optimum vector using the negative gradient at that point:

\[ \mathbf{w}(n + 1) = \mathbf{w}(n) - \mu \frac{\partial J}{\partial \mathbf{w}} \]  \hspace{1cm} (3.65)

where \( \mu \) is a fixed step size and \( \mathbf{w} \) is a vector consisting of all the weights in the network

\[ \mathbf{w}(n) = \left[ w_{ij}^{(1)}(n), w_{ij}^{(2)}(n), ..., w_{m_i n_j}^{(L)}(n), w_{m_i n_j}^{(L+1)}(n) \right] \]  \hspace{1cm} (3.66)

Details of the derivation of the BP algorithm can be found in Pao [89] or Haykin [88]. The complete update algorithm can be expressed as follows:

\[ w_{ij}^{(l)}(n + 1) = w_{ij}^{(l)}(n) - \mu \delta_j^{(l+1)} \cdot x_i^{(l)}(n) \]  \hspace{1cm} (3.67)

\[ \delta_j^{(l)}(n) = \begin{cases} -2e(n) \varphi'(s_j^{(L)}(n)) & \text{for } l = L \\ \varphi'(s_j^{(l)}(n)) \cdot \sum_{k=1}^{N_l} \delta_k^{(l+1)}(n) \cdot w_{jk}^{(l)}(n) & \text{for } 1 \leq l \leq L - 1 \end{cases} \]  \hspace{1cm} (3.68)

where:
3.7 Multilayer Perceptron Neural Networks

\( x_i^{(l)}(n) \) represents the output from node \( i \) in layer \( l \).

\( w_{jk}^{(l)}(n) \) represents the weight connecting node \( j \) in layer \( l \) to node \( k \) in layer \( l+1 \).

\( s_j^{(l)}(n) \) represents the input activation to node \( j \) in layer \( l \).

\( \varphi'() \) represents the derivative of the sigmoid function.

\( \delta_j^{(l)}(n) \) represents the local gradient "delta" term of node \( k \) in layer \( l \).

\( L \) is the total number of layers in the network.

\( n \) is the time index.

\( \mu \) is the step size parameter.

The backpropagation terms are illustrated in Figure 3.8.

![Backpropagation Diagram](image)

**FIGURE 3.8** Backward error propagation.

3.7.2 Enhanced Backpropagation Methods

One of the simplest and most popular methods for enhancing the speed of convergence of the BP
method is to apply momentum [87] to the weight updates according to,

$$w_{ij}^{(l)}(n+1) = w_{ij}^{(l)}(n) + \alpha \left[ w_{ij}^{(l)}(n) - w_{ij}^{(l)}(n-1) \right] - \mu \delta_{j}^{(l+1)}(n) \cdot x_{i}^{(l)}(n) \quad (3.69)$$

where $0 < \alpha < 1$ is called the momentum constant. Essentially, a small amount of the previous weight value is added to the current weight update. This has the effect of stabilizing oscillating weight updates and accelerating weight updates which have the same sign [90].

The Kalman algorithm may be applied to neural networks to improve convergence rates in the same way that the RLS can speed up convergence rates for FIR based structures. The Enhanced Backpropagation (EBP) method [91][92] divides the MLP into a number of linear subsystems for each layer and invokes the RLS (or linear Kalman filter algorithm) to update the weights so as to minimize the error in each layer with respect to the activation potential into each neuron. All neurons in the same layer have the same input vector $x(n)$ and only one $P$ matrix per layer is required.

The Extended Kalman Algorithm (EKA) on the other hand divides the MLP into a number of non-linear subsystems and minimizes the error at the neuron output [92][93][94]. The activation function output is linearized by using a Taylor series expansion about the current weight estimate. Each neuron perceives its own "linearized" input and therefore must also maintain its own copy of $P$ even if the input $x(n)$ is shared with other neurons. This leads to higher computational and storage cost [95] compared to the EBP algorithm, but also provides superior performance [92].

Enhanced BP methods are attractive for enhancing convergence in certain classification problems and for inputs that have a large eigenvalue spread. In the linear domain, RLS algorithms have poorer tracking ability than algorithms based on instantaneous gradient estimates (for example LMS), when operating in low SNR conditions [96]. The fact that the RLS algorithm is used to
train the weights may cause poorer tracking ability than the conventional BP algorithm, however, the author could find no results presented in the literature to confirm this hypothesis. It may also be possible to enhance EKA based algorithms with an acceleration factor similar to the one used in the accelerated SFTF algorithm used on linear FIR structures.

3.8 Summary

In this chapter, the principles and adaptation techniques of linear and nonlinear adaptive filters were reviewed. A brief survey of the research work in this area and some applications were presented. Following a discussion of the adaptive FIR and IIR filter, the gradient based search technique was presented including the LMS, NLMS, MVSS algorithms, as well as the EE-IIR and OE-IIR LMS variations for IIR filters. The least squares search techniques presented included the exponential RLS and SFTF algorithms, including the RLS update as applied to the OE-IIR filter. The third search technique presented was the conjugate gradient algorithm. The theory of Volterra filters and the LMS based training technique were next covered, followed by neural network filters and the celebrated gradient backpropagation algorithm.
Chapter 4

Acoustic Echo Cancellation

This chapter contains six major subsections. Section 4.1 introduces the acoustic echo control problem in HFT’s and presents performance requirements. Section 4.2 presents the physical test system used to obtain the field data, including precautions taken to prevent impaired measurements. Section 4.3 investigates the use of IIR structures for AEC’s. Section 4.4 investigates the performance limitations of AEC’s including both physical and algorithmic factors, with supporting analysis, simulations and experimental test results. Section 4.5 presents results on the effects of enclosure vibrations and rattling with a typical HFT. Section 4.6 determines how transducer nonlinearities affect the performance of linear algorithms, and finally a summary is presented in Section 4.7.

4.1 The Handsfree Telephone Problem

The basic objective of a handsfree AEC is to provide ease of communications for conversational purposes by allowing the user to move about freely in his or her environment without any loss in speech quality. However, acoustical feedback and echoes are major sources of annoyance in con-
4.1 The Handsfree Telephone Problem

versations using HFT's, hence a method of controlling echo is needed. Figure 4.1 shows a hands-
free telephone set intended for full-duplex operation. The microphone or primary signal, is a
summation of the received echo, room noise and near end speech signal. The HFT contains two
echo control devices. The first is a hybrid echo canceller which removes echoes that leak through
the imperfect 2-to-4 wire hybrid coupler, as well as reflections from the line. The second is an
acoustic echo canceller which removes part or all of the acoustic signal coming from the loud-
speaker, including echoes from the environment. Hybrid echo characteristics can change from call
to call, and acoustic echo characteristics are affected by any movements within the immediate sur-
roundings, hence it is necessary to implement an adaptive filter to cancel the echoes. However, the
task of acoustic echo cancellation is a far more difficult task than hybrid echo cancellation for the
following reasons:

- The hybrid echo characteristics are typically stationary on a per-call basis, whereas the acoustic
echo path is affected by any movements within its surroundings. This means that dynamic
tracking is extremely important in AEC's.

- The acoustic echo duration is far greater than the hybrid echo duration, typically by an order of
magnitude, possibly up to a few hundred milliseconds in large rooms. This means that filter
structures with a long memory (i.e. filter order) are required in AEC applications.

- The hybrid is well modelled by a linear system whereas the acoustic path has at least one non-
linear component (the loudspeaker), as well as other components that are difficult to model, for
example air turbulence in the vicinity of the microphone and vibration and resonances within
the HFT enclosure.

- The magnitude of the coupling between the loudspeaker and microphone is significant and
results from a direct path through the air and the enclosure itself. The desired speaker signal is
usually much smaller in magnitude than the signal to be cancelled. In hybrid echo cancellation,
this large acoustical coupling is absent.
The magnitude of the background noise in a room can be significant, whereas in hybrid systems, the noise level is usually much less severe.

The effects of the above differences limit the performance of acoustic echo cancellers compared to hybrid echo cancellers. Typically, 70 dB of echo cancellation can be achieved in practice for hybrids \cite{97} whereas 25 dB seems to be the current practical limit to the achievable acoustic echo cancellation \cite{1}\cite{96}\cite{98}\cite{99}. To be commercially attractive and audibly nonintrusive, HFT's should achieve echo cancellation levels of at least 30 dB in times of less than 100 ms \footnote{Internal correspondence with Nortel.} with voice signals using reasonably priced digital signal processors (DSPs). The state-of-the-art in AEC's for handsfree terminals cannot yet reach this goal and as a result it has been a hotbed of on-going research for many years. A complete background survey in this area is beyond the scope of this chapter, however, references \cite{100}, \cite{101} and \cite{102} provide an exhaustive summary of over 100 technical papers.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{echo_cancellation_diagram}
\caption{Echo cancellation in the handsfree environment. The hybrid echo canceller is shown on the left and the acoustic echo canceller is shown on the right. The echo replica $\hat{y}(n)$ is subtracted from primary signal $p(n)$ to give the error signal $e(n)$.}
\end{figure}
4.1.1 Performance Requirements

Objective Requirements. A standard AEC performance metric is the steady-state Echo Return Loss Enhancement (ERLE) during single talk mode, i.e. when the near end speech is absent, and is defined by [103]:

\[
ERLE (dB) = \left[ 10 \log \frac{E[p^2(n)]}{E[e^2(n)]} \right] = 10 \log \left[ \frac{\sigma_p^2}{\sigma_e^2} \right] = 10 \log \left( \frac{\sum_{r=0}^{n_w} [p(n-r)]^2}{\sum_{r=0}^{n_w} [e(n-r)]^2} \right)
\]

(4.1)

where \(\sigma_p^2\) and \(\sigma_e^2\) refer to the variances of the primary and error signals respectively and \(E\) is the statistical expectation operator. For on-line measurements, the later expression in (4.1) is used to compute the ERLE at time \(n\) where \(n_w\) is the size of the window average, nominally set to 500.

The ERLE value provides a measure of how much the echo is attenuated in the absence of measurement noise, however, the value obtained from (4.1) includes the effects of both the acoustical isolation between the loudspeaker and microphone, and the electrical modelling accuracy. For example, an infinite ERLE can be obtained with either a perfect electrical model in the AEC portion or 100% acoustic isolation between the loudspeaker and microphone. Generally, the physical ERLE value recorded is a combination of the two mechanisms. In addition, a large uncorrelated noise component in the primary signal will also affect the ERLE value so some caution should be used when using ERLE as the primary objective measure of modelling performance. Objective performance standards may be found in [104] through [109] which are summarized as follows:
TABLE 4.1  Objective performance requirements for handsfree telephones.

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIC</td>
<td>Initial convergence time</td>
<td>1 sec, 20 dB</td>
</tr>
<tr>
<td>TRDT</td>
<td>Recovery time after double talk</td>
<td>1 sec, 20 dB</td>
</tr>
<tr>
<td>TCLWPV</td>
<td>Echo loss during echo path variation</td>
<td>&gt;10 dB</td>
</tr>
<tr>
<td>TRPV</td>
<td>Recovery time after echo path variation</td>
<td>1 sec, 20 dB</td>
</tr>
<tr>
<td>TCLWST</td>
<td>Echo loss in single talk</td>
<td>&gt;45 dB</td>
</tr>
<tr>
<td>TCLWDT</td>
<td>Echo loss in double talk mode</td>
<td>&gt;25 dB</td>
</tr>
<tr>
<td>ARDT</td>
<td>Received speech attenuation in double talk mode</td>
<td>&gt;6 dB</td>
</tr>
<tr>
<td>ARST</td>
<td>Transmitted speech attenuation in double talk</td>
<td>&gt;6 dB</td>
</tr>
<tr>
<td>DRST</td>
<td>Received speech distortion in double talk mode</td>
<td>currently under study</td>
</tr>
<tr>
<td>DRDT</td>
<td>Transmitted speech distortion in double talk mode</td>
<td>currently under study</td>
</tr>
<tr>
<td>TONST</td>
<td>Break-in time in single talk mode</td>
<td>20 ms, 3 dB</td>
</tr>
<tr>
<td>TOND T</td>
<td>Break-in time in double talk mode</td>
<td>20 ms, 6 dB</td>
</tr>
</tbody>
</table>

The attainable early-to-late ratio (ELR) is another objective performance measure which is defined as the ratio of energy received before 40 ms to that received after. A high ELR is subjectively more pleasing than a low ELR.

**Subjective Requirements.** Very little work has been done to correlate objective criteria and subjective test results with regard to acoustic echo control [110]. Quantities such as naturalness of transmitted speech and quality of conversation with regard to ease of speaking and interruption are not well defined in the literature, although these are most important to the user. The echo cancellation level subjectively required for HFT’s depends highly on the environment. For example, in [111] an assessment is performed in audio teleconferencing environments, where for an overall round trip delay time of 100 ms and a reverberation time of 400 ms, 40 dB of acoustic echo cancellation is considered necessary. Experiments on the minimum detectable delay of speech (assuming a single echo) confirm that 23 ms seems to be the detectable limit [112] and that human sensitivity to the echo signal increases as the delay between the original and echo signal increases. Mean opinion
scores on speech signals also degrade with increased echo delay (i.e. lower ELR). Experimental results in [110] show that the annoyance due to acoustic echo level is strongly dependent on the background noise level, and that the annoyance of the background noise subjectively masks the echo. Additional results presented in [113] for automobile interiors show that echoes are attenuated approximately 1 dB for every 1 ms of delay which means that by the time the echo is perceivable, it has already become attenuated by at least 20 dB. It would appear that given this information, the AEC problem (in automobiles) is of secondary importance with respect to noise cancellation. In the context of improving the subjective quality of HFT’s, both speech enhancement (i.e. noise and distortion reduction) and echo cancellation should therefore be taken into account for obtaining an overall quality enhancement. However, where noise reduction can be achieved with linear systems, distortion reduction usually requires a nonlinear architecture.

**Implications for HFT Design.** High figures of ERLE up to 45 dB are often proposed in the case of large transmission delays (See Table 4.1) however, since current technology/algorithms are generally unable to provide such high attenuation, additional variable losses in the receive and/or transmit path are frequently used. More importantly, there seems to be no mention of how physical limitations such as loudspeaker nonlinearity will affect the practical achievement of such high ERLE values without the inclusion of additional losses. In terms of achieving both a subjectively pleasing speech quality and the specifications listed in Table 4.1 it would appear necessary to accommodate the nonlinearity of the loudspeaker model into the AEC design.
4.1 The Handsfree Telephone Problem

4.1.2 Acoustic Reverberation in Rooms

In this section we investigate the characteristics of acoustic reverberation in rooms, and how it impacts on the architecture of AEC's.

_Reverberation Time_. The reverberation time $T_{60}$ is defined as the length of time necessary for all reflections in a room to decay by 60 dB [114]:

$$T_{60} = \frac{6.91}{\delta} = \frac{-13.8}{c\left(\frac{1}{L_x} + \frac{1}{L_y} + \frac{1}{L_z}\right)\ln \beta}$$

(4.1)

where $\delta$ is the average damping constant of all surfaces in the room, $c = 340$ m/s is the speed of sound in air and $L_x$, $L_y$, $L_z$ are the $x$, $y$, $z$ room dimensions. $\beta$ is the reflection coefficient varying between 0-1. More importantly, $\beta$ has a frequency dependence: generally low frequencies have a higher reflection coefficient than higher frequencies for most reflecting surfaces. Values of $T_{60}$ can range from 0.3s (living rooms and furnished conference rooms) up to 10s (large churches). For an AEC in a typical conference room operating with a sample rate of 8kHz, this means that (in the absence of other limitations) up to several thousand taps are required to obtain 40-60 dB echo cancellation.

_Vibrational Modes_. In a rectangular room, the number of vibrational modes $N$ in the frequency range from 0 to $f$ is given by [114]:

$$N = \frac{4\pi}{3}V\left(\frac{f}{c}\right)^3 + \frac{\pi}{4}S\left(\frac{f}{c}\right)^2 + \frac{L}{8}\left(\frac{f}{c}\right)$$

(4.2)

where $V$ is the volume of the room, $S$ is the area of all walls, $L$ the sum of all edge lengths of the walls of the rectangular room. For example: for a room with dimensions $L_x=3m$, $L_y=4m$ and

---

1. Equation (4.1) is intended for regularly shaped rooms free of furniture and people.
L_x=5\text{m}, the number of eigenfrequencies\(^1\) in the range \([0, 3.4 \text{ kHz}]\) is 247787. Since this number represents half the number of poles necessary to completely model the physical phenomenon exactly, it is obvious that exact cancellation would require an extremely complicated architecture.

*Acoustic reverberation is so complicated that it can only be investigated under statistical considerations.* The number of acoustic modes in the audiofrequency range is much greater than the coefficients available to obtain a 40-60 dB ERLE typically demanded of an AEC. Fortunately, the eigenfrequencies are highly overlapped and therefore they can be reduced to statistical averages to provide a much more parsimonious number of modes \([115]\). It has been observed that the frequency responses of typical rooms are composed of a sequence of maxima and minima spaced roughly 5 Hz apart. The average frequency spacing of adjacent maxima can be calculated using Rice's formula \((\Delta f_{\text{max}}) = \delta / (\sqrt{3})\) \([116]\) where \(\delta\) is defined in equation (4.1). If we model each maximum/minimum pair by a 2nd order IIR filter section consisting of 5 free parameters per pair, the total number of parameters to model the range \([0-3.4 \text{ kHz}]\) assuming a 5 Hz spacing is approximately 3400. This is far less than described by (4.2) however, it is still quite large.

**Temporal Distribution of Reflections.** The temporal nature of a reverberant signal in a room can be obtained by using the method of room images \([114]\). The resulting temporal density of reflections \(N_R\) arriving at time \(t\) is:

\[
\frac{dN_R}{dt} = 4\pi \frac{c^2}{\nu} \frac{t^2}{V} \tag{4.3}
\]

\(T_{60}\) and \(N_R\) can be used to generate realistic computer-simulated room impulse responses. This method and measured room impulses are used for simulations presented in this thesis.

\(^1\) Eigenfrequencies are the solutions to the room wave equation and can be thought of as vibrational modes (or frequencies) for which a standing wave will exist in a rectangular room with rigid boundaries.
4.2 Experimental Set-up

The measurement setup used in this thesis is shown in Figure 4.2. A modified speakerphone\(^1\) is positioned in either an anechoic chamber or a conference room and excited with either bandlimited white noise or speech depending on the conditions desired. Noise was generated using a General Radio Company 1390-B Random Noise Generator. Artificial speech recordings were provided by Bell Northern Research and reproduced using a Sony TC-D7 Digital Audio Tape (DAT) recorder. The source signal selected was then filtered to the telephony bandwidth (200Hz to 3400Hz) with a cascade of two National Semiconductor TP3040 switched capacitor filters to provide greater than 60dB of stopband attenuation. The loudspeaker of the modified phone was driven with an amplified version of this filtered excitation signal. The amplification was accomplished with a 75W dual-channel Samson Servo-150 studio-quality power amplifier. The output level of the amplifier can be varied to provide a sound pressure level (SPL) anywhere from 60 dB to 100 dB as measured 0.5m directly above the loudspeaker, depending on the HFT used. Conference room dimensions and layouts may be found in Appendix A. A listing of the equipment used and relevant parameters may be found in Appendix B.

A set of pre-conditioning circuits were used to attenuate the reference signal, and amplify the primary microphone signal to equate their levels at the DAT's inputs. Both circuits are based on the Analog Devices AD524 instrumentation amplifier. A separate high quality microphone was sometimes used to bypass the electret microphone in order to remove enclosure vibration and coupling effects. The primary and reference signals were recorded to DAT using a Teac DA-P20 DAT recorder at a level of approximately -6 dB with noise, and a peak level of -6dB with speech. The data was later downloaded to a personal computer workstation (PC) for off-line processing by re-

\(^1\) Each phone is modified to allow access to the internal loudspeaker and microphone terminals.
amplifying the data at 16kHz with an Ariel DSP-96 card. The data vectors were then issued to a variety of adaptive filtering programs written specifically for this thesis in Matlab and C. Schematics of various circuits used in the experimental setup can be found in Appendix C.

![Block diagram of the experimental setup](image)

**FIGURE 4.2** Block diagram of the experimental setup

Several commercially available HFT’s are used in the experiments (refer to Appendix B for a list of HFT and transducer parameters). The HFT under test is placed either on top of a conference table (conference room recording) or on a 1m square board on the floor of an anechoic chamber (anechoic recording). Measurements were made with a number of modified handsfree terminals and separate microphone and loudspeaker components. By separating the loudspeaker and microphone, different quality loudspeakers can be tested for linearity, and in addition, the coupling between the loudspeaker and microphone due to the enclosure effects can be removed.

Loudspeakers tested outside of a handsfree terminal enclosure in this way were mounted in a standard baffle made of 3/4 inch plywood with the loudspeakers placed as indicated in Appendix A,
4.2 Experimental Set-up

Figure A.3. For quick recordings in the lab, a noise shielding box was constructed consisting of two boxes, one inside the other to provide noise immunity from the external environment. It was constructed with 0.25 inch thick cardboard with corrugated foam glued to both the inside and outside surfaces. The box provided 21.9 dB of sound attenuation, however, some reverberation was still observed at 100 ms.

Impulse response measurements are obtained experimentally by averaging the weight values over the last few thousand samples of a 4 second data file. The reference excitation in this case is filtered noise, with amplitudes varying between 55 dB and 100 dB sound pressure level (depending on HFT used and measurement desired) as measured 0.5 m from the loudspeaker. Results obtained using HFT #1 in anechoic and conference room environments are shown in Figure 4.3.

![Figure 4.3](image)

**FIGURE 4.3** HFT#1 impulse responses. (a) as measured in the anechoic chamber (b) as measured in conference room #2.

4.2.1 Considerations for Practical Problems

There were many practical problems associated with obtaining good measurements with the experimental setup. These problems included obtaining correct dynamic range for the DAT both during
recording, and during playback to the ARIEL board, common mode voltages caused by 60 Hz AC power lines and fluorescent lighting, DC bias effects and linearity settings on the ARIEL card itself.

- Particular attention to circuit layout and ground loops was necessary. Initial problems were caused by high gain amplifiers and balanced-to-single ended conversion circuits, and lack of a single point ground.

- In initial amplifier designs, harmonics from the 60 Hz AC power lines and 33 KHz electric fields generated by fluorescent lights were picked up by the microphone amplifier. A number of circuit revisions were necessary to obtain decent rejection of these interfering signals.

- Physical separation of the signal conditioning amplifiers was necessary to prevent electrical crosstalk from correlating the reference and primary signals.

- Solid grounding and installation of small decoupling capacitors on the inputs and outputs of the switched capacitor filter circuits were necessary to reduce noise generated by a 2.048 MHz TTL level master clock inside the BPF box.

- Any unshielded cables tended to pick up noise in the audio band, primarily 60Hz harmonics, due to the close proximity of the test equipment to the circuits. It was necessary to use a shielded\(^1\) twin-axial cable to obtain small (microphone) signals, as well as prevent crosstalk between the high level reference and low level primary cable inputs.

- Linearity and gain settings to avoid clipping in the ARIEL board had to be adjusted to optimal settings. Early measurements revealed that the linearity of the ARIEL card was not adjusted correctly.

**DC bias compensation.** In any physical circuit, DC bias voltages will be present. Temperature and loading fluctuations will vary the DC bias at the amplifier outputs up to 10mV or more. Offsets in the Ariel card were found to be approximately -15 mV and -30 mV for the left and right channels respectively. Algorithms that do not compensate for DC bias will not converge to the optimum

---

\(^1\) The shield is grounded to chassis ground at the AD524 end, an is ungrounded at the microphone end.
weights. Rather than modifying the data files, each of the algorithms was modified to include an adaptive DC bias term. The compensated NLMS algorithm (for example) is the same as previously described except an additional tap weight is added to insert a DC bias value:

\[ y(n) = w^T(n)x(n) + w_{bias}(n)b \]  \hspace{1cm} (4.4)

\[ w(n + 1) = w(n) + \mu e(n)x(n) \]  \hspace{1cm} (4.5)

\[ w_{bias}(n + 1) = w_{bias}(n) + \mu e(n)b \]  \hspace{1cm} (4.6)

where \( b \) is a fixed bias weight, in most cases set to 0.01 and \( \mu \) is the step size which must be chosen so that:

\[ \mu = \frac{\alpha}{\epsilon + \|x(n)\|^2 + b^2} \]  \hspace{1cm} (4.7)

The effect of this additional parameter is to slow convergence slightly and reduce the achievable ERLE from approximately 127 dB to approximately 70 dB, however, this value is still far below where most physical ERLE values were located (20-40 dB).

### 4.2.2 Maximum Achievable ERLE of the Experimental Set-up

The maximum achievable ERLE of the experimental setup was found by replacing the Loudspeaker-Room-Enclosure-Microphone (LREM) system (see Figure 4.2) with a passive attenuator network. The modified setup was excited with bandlimited white noise. The achievable ERLE was calculated using a 10 tap NLMS adaptive filter with a step size of 0.5. The resulting curve is shown in Figure 4.4. Also plotted is an asymptotic curve which represents an estimate of the maximum achievable ERLE of the experimental setup based on an estimate of the primary signal's
SNR. The maximum achievable ERLE of the measurement setup was found to be approximately 56dB.

![Graph](image)

**FIGURE 4.4** Optimum steady-state ERLE performance of the experimental setup based on the converged ERLE of a 10 tap NLMS adaptive filter with a step size of 0.5 (Linear FIR curve), and an estimate of the SNR of the primary signal (Asymptote curve).

Further experiments were conducted to evaluate the experimental setup more completely and provide an explanation for the limitation in performance. The results are displayed in Figure 4.5. The line labelled *Ariel* corresponds to converged ERLE values for a variety of equivalent DAT record levels\(^1\) obtained by injecting bandlimited white noise directly into the analog input ports of the Ariel board. An achievable ERLE in excess of 70dB was obtained for record levels greater than -13dB. The additional quantization noise added by first sampling the data to DAT degraded the achievable ERLE by approximately 8dB as shown in the curve labelled *DAT + Ariel*. Finally, the performance of the complete setup is illustrated by the line labelled *Full Setup*. This curve is similar to that shown in Figure 4.4 except that the reference signal voltage is held constant at 1 Vrms

---

\(^1\) For the *Ariel* curve the input voltage is converted to an equivalent DAT record level with knowledge of the DAT's input/output transfer characteristics to support comparison between the curves.
while the record level at the DAT is adjusted. These results suggest that the current limitation of 56dB is caused by noise added by the primary (microphone) amplifier. Improving the noise performance of this amplifier is unnecessary, however, because as demonstrated in the remainder of this chapter a variety of physical limitations act to limit ERLE to a level below 56dB.

![Graph showing limitations to achievable ERLE due to measurement setup](image)

**FIGURE 4.5** Comparison of optimum steady-state ERLE of the Ariel, the DAT and Ariel, and the full setup based on a 10 tap NLMS adaptive filter with a step size of 0.5.

### 4.2.3 Speakerphone Test Results

The six HFT's listed in Appendix B were tested in an anechoic chamber to determine the magnitude of the limitations to ERLE caused by design variation. The purpose here is to provide some measure of the spread in ERLE due to HFT designs, and not to isolate each of the factors which contribute to this error. The steady-state ERLE performance for each HFT, at each SPL level was obtained by averaging the instantaneous ERLE over the last 4000 samples of 4 seconds of adaptation (sampled at 16 kHz) using the NLMS algorithm with a step size $\alpha=0.5$. The results are plotted in Figure 4.6. The steady-state ERLE varies over a very large range, with HFT#1 and #6 having the
best performance characteristics and HFT#5 having the worst. At high volumes, all HFT's are limited by strong nonlinear distortion and vibration effects\(^1\) introduced by low-quality loudspeakers and poor acoustic design. Based on these results, HFT #1 and #6 are used for subsequent experimental baseline measurements.

![Graph showing ERLE for six HFT's at various SPL levels.](image)

**FIGURE 4.6** Converged ERLE for six HFT's at various sound pressure levels (measured 1m from the loudspeaker) in an anechoic chamber. 1000 tap FIR trained with NLMS, all cases.

### 4.3 Evaluation of Linear IIR Structures for Acoustic Echo Control

In the literature on this subject, Tahermezhadi and Liu [117] propose a real-time implementation of an IIR AEC using the LATIN (*Lattice* and *Inverse Lattice*) structure originally proposed by Chao and Tsujii [118]-[119]. Experimental ERLE values in [117] of 25 dB are reported using only 30 zeros and 30 poles on speech signals, however no simulations were provided to verify these findings and the characteristics of the LREM are not presented. Chao and Tsujii [118] use a simplified LREM consisting of one pole and 9 zeros. It has been found (see Section 4.2) that small amounts

---

1. To be shown in Section 4.6 and Section 4.5.
of crosstalk in physical circuits can correlate the primary and reference signals and provide overly
optimistic results, which might explain how 25 dB ERLE is obtained with so few variables assum-
ing that a real LREM was used. In [120] Fan and Jenkins show that an IIR echo canceller for a data
hybrid performs better than an FIR filter after convergence, and that convergence is quite slow if
the order of the filter is greater than two. However, it is not clear if the same conclusions can be
extended to the AEC case.

4.3.1 Hankel Error Bound for Approximating an FIR Filter with an IIR Filter

In [115] the fundamental question of whether poor performance stems from a sub-optimality of the
updating procedure or from an irrelevancy of the pole-zero structure itself is investigated. Is it pos-
sible to approximate with a small error bound an all-zero transfer function by a pole-zero transfer
function with fewer parameters, and does it make sense to fit a pole-zero model to the phenome-
on underlying the LREM? The Hankel-norm approximation of a known all-zero transfer function
\( H(z) \) of order \( M \) can be used to obtain an error bound on the use of a reduced order IIR filter
[121][122]. If \( H(z) \) is defined as the useful part of an all zero transfer function to be identified:

\[
H(z) = h_1 + h_2 z^{-1} + \ldots + h_M z^{-(M-1)}
\]  

(4.8)

Then the Hankel matrix \( \Gamma_H \) is defined as a symmetric matrix formed out of the coefficients
\( h_1...h_M \) of the room impulse function \( H(z) \):

---

1. Although an actual room impulse response theoretically consists of poles only, because all frequencies
are reflected to some degree, it is convenient to model the impulse response with an all-zero structure
\[
\Gamma_H = \begin{bmatrix}
h_1 & h_2 & \ldots & h_M \\
h_2 & \ldots & h_M & 0 \\
\vdots & \vdots & \ddots & \vdots \\
h_M & 0 & \ldots & 0
\end{bmatrix}
\]  \hspace{1cm} (4.9)

The maximum error bound on approximating the \(M\)\(^{th}\) order FIR response with an \(N\)\(^{th}\) order IIR response can be obtained by the following equation:

\[
\text{error bound} \leq 2 \left[ \sigma_{K+1} + \sigma_{K+2} + \ldots + \sigma_M \right] \quad \text{where} \quad \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_M > 0
\]  \hspace{1cm} (4.10)

where \(\sigma_1, \sigma_2, \ldots, \sigma_M\) are the Hankel singular values of \(\Gamma_H\), \(K=M-N\), where \(M\) is the number of coefficients in \(H(z)\) and \(N\) is the order of the approximating IIR filter. The diagonal matrix \(\Sigma\) may be obtained by finding a nonsingular transformation matrix \(T\) such that

\[
T^T W T = \Sigma = \text{diag} [\sigma_1, \sigma_2, \ldots, \sigma_M]
\]  \hspace{1cm} (4.11)

where \(W = \Gamma_H^T \Gamma_H\) \hspace{1cm} (4.12)

Simulations results in [115] showed that a radio-mobile echo path consisting of 512 all-zero coefficients could efficiently be modelled with a -30 dB error using a pole-zero structure with 128 parameters, however, a -28.8 dB error could be obtained with an FIR structure with the same number of coefficients!

**Hankel Bound Experimental Results.** Two room impulse responses (slowly decaying and quickly decaying) were generated using the techniques described in Section 4.1.2. The results showing the approximation error bounds are shown in Figure 4.7. The low error bound near \(O(\text{IIR})=256\) is artifactual and comes about as perfect modelling is achieved when \(O(\text{FIR})=O(\text{IIR})\). However, the
results do provide a rough estimate of the achievable error bound for lower IIR filter orders up to approximately 200.

\begin{figure}[h]
\centering
\begin{subfigure}{0.45\textwidth}
\includegraphics[width=\textwidth]{figure_a}
\caption{(a)}
\end{subfigure}
\begin{subfigure}{0.45\textwidth}
\includegraphics[width=\textwidth]{figure_b}
\caption{(b)}
\end{subfigure}
\caption{Approximation error bound [dB] for identifying LREM's with different reverberation times. Impulse responses are obtained from measured LREM's truncated to 256 points. (a) Slowly decaying LREM needs an IIR filter order of 200 to achieve a -30 dB error bound. (b) Quickly decaying LREM needs an IIR order of 110 to achieve the same error bound.}
\end{figure}

For a slowly decaying LREM impulse response, the number of IIR parameters required to model the impulse is almost the same as the FIR case, however, for faster decaying impulses, the number of IIR parameters is correspondingly less. This analysis suggests that for environments that possess small reverberation times, like automobiles, IIR structures might be suitable filter candidates, however, for environments that have long reverberation times, the advantages of IIR compared to FIR structures are marginal.

\subsection{4.3.2 Comparison of MA, ARX and OE-IIR Modelling}

In the literature, Gundvangen and Flockton [123] [124] made a comparison between pole-zero and all-zero modelling of acoustic transfer functions. Three different identification structures were used, one FIR and two IIR structures based on output error (OE) and equation error\(^1\) (EE) formulations. In the IIR models there is another parameter in addition to the total number of coefficients,
4.3 Evaluation of Linear IIR Structures for Acoustic Echo Control

namely the ratio of zeros to poles. The ratio that gave the best ERLE for the OE model was approximately 5:2, although the exact ratio was not at all critical according to their results. They found that the OE consistently gave marginally better performance than the FIR and that the EE model performed a few dB worse than the FIR due to the bias of this particular model. They also reported that to achieve a particular ERLE, the number of coefficients for both the FIR and IIR models was found to increase substantially with increased reverberation time. This conclusion is also supported by the Hankel error bound analysis presented in Section 4.3.1.

Experimental Results using OE and EE-IIR Modelling. In order to investigate the pole/zero ratio question as applied to transducer components only (which was not covered in [124]) an OE-IIR model was fitted to measured data obtained in an anechoic chamber with results shown in Figure 4.8 (a). For comparison, results using the EE-IIR algorithm on data collected using HFI#6 are shown in Figure 4.8 (b).

![Graph 4.8(a)](image1)

![Graph 4.8(b)](image2)

**FIGURE 4.8** Experimental results of IIR modelling showing ERLE vs. ratio of poles to zeros for IIR models fitted to experimental LREM data. (a) OE-IIR model for HFT#6 in conference room #2. (b) ARX modelling for SPK#1 and MIC#2 in an anechoic chamber.

1. The EE formulation can be obtained through a two input ARX fitting.
For both experiments, the total number of parameters was fixed at 600, with the number of poles varying between 0 (i.e. all-zeros) to 600 (all-pole). The parameters used are shown in Table 4.2.

<table>
<thead>
<tr>
<th>Figure</th>
<th>Algorithm</th>
<th>Parameters</th>
<th>Location</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>EE LMS IIR, Appendix D.6</td>
<td>$\mu_a=\mu_b=0.001$</td>
<td>Anechoic</td>
<td>Transducers</td>
</tr>
<tr>
<td>(b)</td>
<td>OE LMS IIR, Appendix D.7</td>
<td>$\mu_a=\mu_b=0.001$</td>
<td>Conf. Rm. #2</td>
<td>HFT#6</td>
</tr>
</tbody>
</table>

The EE-IIR algorithm obtains a slight improvement in performance with an increasing number of poles, for low volumes. Almost no change in ERLE is observed at high volumes by changing the pole/zero ratio. The results presented here verify conclusions made in [124] about the ratio of pole/zero ratio (being not critical) and also suggest, as in [124] that in the HFT context, the advantages of IIR modelling are minimal. Results using the OE-IIR algorithm shown in Figure 4.8 (b) indicate again that there seems to be no overall gain in ERLE by swapping poles for zeros.

**Conclusions.** It does not appear possible to obtain the same ERLE with *significantly less* parameters using IIR filters (compared to FIR filters) to model a reverberant echo path due to the underlying physical *complexity* of the LREM system. This seems to be confirmed by the Hankel-norm error bound results of Figure 4.7. Recursive IIR elements may be added to an existing FIR structure to improve performance by a few dB (see Figure 4.8), however this only occurs at low SPL when the loudspeaker operates in the "linear region". IIR structures are better suited to modelling simpler physical systems with oscillatory behaviors and fewer parameters than are typically required to completely model a real-world LREM. *Based on the above analysis, literature survey, and DSP processing limitations, it would appear that all-zero FIR structures have an advantage over IIR methods.* Considering that a typical LREM impulse may be thousands of milliseconds duration, it is best to model the physical phenomenon with an undermodelled FIR system and try to obtain the best fit possible according to the reverberation characteristics.
4.4 AEC Performance Limitations

The limitations of AEC's include the following:

1. **Noise, Finite Precision and Quantization**: Acoustic noise from fans and air conditioning in the room as well as thermal and impulsive circuit noise from amplifiers, and DSP related noise such as truncation, quantization and finite word lengths place a limit on the achievable steady-state ERLE [125] [98].

2. **Undermodelling of the Acoustic Transfer Function**: When the number of taps or variables in the AEC adaptive filter is less than the acoustic impulse response of the room, the unc cancelling tail portion of the LREM manifests itself as a finite error at the output of the AEC [1][126].

3. **Enclosure Vibration Effects**: An important new finding\(^1\) is that vibration and resonances within the enclosure can significantly limit achievable ERLE if proper *acoustic* design of the enclosure is not realized. In addition, the early reflections in the impulse response are due to LREM coupling effects which are generally much larger in amplitude than the delayed echoes [127].

4. **Transducer Nonlinearities**:\(^2\) Generated mainly in the loudspeaker, nonlinear distortion effectively puts a limit on the achievable ERLE of algorithms based on linear mechanics [128].

5. **Algorithmic Performance**: Initial convergence rate, dynamic tracking ability in nonstationary conditions, behavior with correlated (speech) inputs [96] and the ability to detect and handle double talk situations [129].

All of the above mentioned limitations will serve to reduce the achievable ERLE and will be discussed in greater detail subsequently. The model used in the analysis includes the above limitations is shown in Figure 4.9.

4.4.1 Noise, Finite Precision and Quantization

Noise components in the primary signal include room noise, microphone circuit noise and quanti-
4.4 AEC Performance Limitations

![Diagram](image)

**FIGURE 4.9** Complete LREM model includes enclosure reflections and vibration as well as transducer nonlinear responses. The model should also include room, quantization and circuit noise.

Quantization noise. These can be modelled as white noise sources with variances $\sigma_R^2, \sigma_M^2$ and $\sigma_d^2$. The effect of these noise components is to reduce the achievable steady-state ERLE to a level approximately equal to equation (4.1) where $\sigma_e^2$ is replaced by $\sigma_T^2$ and $\sigma_T^2 = \sigma_R^2 + \sigma_M^2 + \sigma_d^2$.

Uncorrelated room noise is usually the largest contributor to the overall noise introduced into the primary path. In the absence of other effects, room noise contribution becomes the asymptote for the achievable converged ERLE [1].

**Microphone/Circuit Noise.** This is generated mainly in the sensing electronics for the microphone element. A typical electret microphone will be biased through a dropping resistor of a few kΩ to provide a bias voltage for the microphone element. The output voltage change $\Delta V$ from such a microphone is defined by $\Delta V = \alpha V_b \Delta C$ where $\alpha$ is a constant, $V_b$ is the bias voltage across the electret capacitive element and $\Delta C$ is the change in capacitance due to the impinging sound wave.
Thermal noise from the bias resistor will be added to the microphone which itself has a noise level in the vicinity of -100 dBV. The microphone output signal is amplified to line levels of approximately 100 mVrms which also amplifies the noise. However, except for anechoic conditions at very low volume levels, this noise has been found to be generally far below typical room noise. This conclusion is also verified by the results shown in Figure 4.5.

**Quantization and Fixed Point Internal Arithmetic in DSPs.** A full analysis of fixed point arithmetic for the LMS algorithm can be found in [126] which states that the total output mean square error for the LMS algorithm can be expressed as:

\[
J = J_{\text{min}} + \frac{1}{2} \mu J_{\text{min}} \text{tr}(\mathbf{R}) + \frac{N \sigma^2_c}{2a^2\mu} + \frac{1}{a^2} (|w_o| + c) \sigma^2_d \tag{4.13}
\]

where \( J \) is the MSE at the output, \( J_{\text{min}} \) is the minimum MSE of the optimal (Weiner) filter \( w_o \), \( a \) is a scaling factor used to bring the maximum levels to +/- 1.0, \( N \) is the number of taps in the FIR structure, \( \mu \) is the step size parameter, and \( c \) is a constant. The coefficient quantization noise \( \sigma^2_c = 2^{-2B_c}/12 \) is the variance introduced by the coefficient quantization where \( B_c \) is the number of binary quantization levels (i.e. bits) in the coefficient representation. Similarly, \( \sigma^2_d \) is the variance introduced by the data (sampling) quantization introduced by \( B_d \) quantization levels.

Although one may be tempted to reduce the step size \( \mu \) to reduce the excess mean square error (i.e. the second term), it may result in a large quantization error generated in the third term due to the **stalling phenomenon** in fixed point processors. There exists an optimum value of \( \mu \) which minimizes the total output MSE, however, it is too small to allow the algorithm to converge completely. Figure 4.10(a) illustrates the MSE as a function of the adaptation step size \( \mu \) where the number of bits \( B_d \) in the data representation and \( B_c \) in the coefficient representation are the same.
4.4 AEC Performance Limitations

![Graphs](image)

**FIGURE 4.10** (a) Total MSE as a function of $\mu$ where $B_d = B_r = 16$ bits, MMSE = 1e-6, $M = 500$ taps, reference input variance $\sigma_r^2 = 0.1$. The dashed line shows the equivalent infinite precision case. (b) ERLE for a variable primary signal power when $p(n)$ is quantized to 16 bits (dotted line) and for $p(n)$ unquantized (solid line).

In the LREM model of Figure 4.9, quantization noise is uncorrelated with the primary signal so appears at the output essentially unchanged. Reference signal quantization noise on the other hand is modified by the adaptive filter transfer function as it adapts. The effect of A/D quantization noise on ERLE is illustrated in Figure 4.10(b). The converged ERLE is independent of the signal level of the primary or reference signal levels when floating point is used. However, when the primary signal is quantized, the maximum level of converged ERLE will be determined by the ratio of the primary signal power to the quantization noise at the location of A/D converter.

In practice an ERLE of 25 to 35 dB seems to be the physical limit to the achievable ERLE in real systems. The results presented in this section have shown that the limitations due to algorithmic noise and truncation effects are far below this limit, and therefore should not have a significant impact on the final ERLE value.
4.4.2 Undermodelling of the LREM

In this section we investigate the effect of using an FIR structure to model a transfer function where the number of parameters in the candidate system will be less than that required to exactly identify the system. This gives the undermodelled system: \( \text{deg}(\hat{H}) < \text{deg}(H) \). An analysis of the finite length AEC by Kuo and Chen [130] shows that for an undermodelled system, an FIR filter will have a better ERLE performance using speech than white noise. Poltmann [131] showed that the achievable ERLE is a function of both the step size and magnitude of the modelled and undermodelled LREM coefficients. For a system modelled by an FIR transfer function the achievable steady state ERLE assuming a white noise input can be calculated from:

\[
ERLE = 10 \log \left( \frac{2 - \mu}{2} \left( \frac{\| h \|^2}{\| \Delta h \|^2 + 1} \right) \right) = TIP/TP
\]  

(4.14)

where \( \| h \|^2 \) represents the power in the modelled coefficients up to order \( M-1 \) and \( \| \Delta h \|^2 \) represents the power in the tail portion of the LREM from \( M \) to infinity. If \( \mu \) is set to 0, (4.14) is equal to the Total Impulse response Power to the uncancelled Tail Power (TIP/TP) ratio originally proposed by Knappe and Goubren [1], who demonstrated that the TIP/TP ratio defines the achievable ERLE up to approximately 20 dB. Beyond this point, other effects dominate. Experimental measurements in [1] show that even at ratios of \((S+N)/N\) of greater than 40 dB, the ERLE did not go beyond 25 dB, with the most likely suggested cause of this ERLE limitation being loudspeaker non-linearities.

The TIP/TP ratio is invaluable for determining the optimum number of AEC filter taps to use given a certain loudspeaker, microphone and enclosure. For example, the impulse response of HFT #6 measured inside conference room #2 at 85 dB SPL is shown in Figure 4.11 (a). Figure 4.11 (b) shows the calculated TIP/TP vs. ERLE ratio compared to the measured ERLE by using (4.14) and
4.4 AEC Performance Limitations

FIGURE 4.11 Undermodelling of the LREM. Measured results of HFT #6 at 85 dB SPL. (a) Impulse response measured in a conference room #2. (b) calculated TIP/TP and measured ERLE using NLMS algorithm.

the NLMS algorithm. The ERLE will follow the TIP/TP ratio very closely up to a certain number of taps according to (4.14), however, in experimental recordings, nonlinearities and vibration effects will limit the achievable ERLE.

Comparison with Hankel Norm Error Bound. A plot showing the comparison of the logarithmic "inverse" of the TIP/TP ratio and Hankel norm error bound for HFT#6 is shown in Figure 4.12. The results show that the Hankel error bound closely follows (but is lower than) the TIP/TP ratio and verifies earlier conclusions that the performance gains of IIR structures over FIR structures are not substantial, and that given the added complexity, FIR based structures are preferable.

4.4.3 Algorithmic Limitations

Dynamic Tracking in Nonstationary Conditions. As objects move and the input signal characteristics become nonstationary, the tracking ability of the algorithm becomes important. For example, although RLS based algorithms have fast convergence and have been shown theoretically to have
tracking capability equivalent or better than the LMS algorithm in low noise [132], it has been found in [96] that algorithms based on instantaneous gradient estimates like the LMS family outperform RLS algorithms in conference room conditions using real speech where the SNR of the primary signal is often quite low.

Colour Insensitive Algorithms. LMS based algorithms suffer from poor convergence when trained by quasi-periodic signals with highly coloured spectra, like speech. The most mature scheme for mitigating this problem is subband filtering [133] [134]. However, RLS based algorithms [135] and block frequency domain methods [136] are also popular. Often, a combination of architectures and algorithms is necessary to obtain satisfactory performance. A brief summary is presented in [113]. A comparative analysis of eight different algorithms is presented in [137] showing measured performance metrics (See Table 4.1) for the single talk mode only using both
USASI noise signals and speech signals. Of eight algorithms/structures tested, the generalized multi-delay filter (GMDF) [138], which is based on [139] obtains the best performance metrics. The unconstrained fast LMS [140] and wavelet decomposition technique [141] also produce good results. An algorithm presented in [142] uses a fast Newton training scheme to obtain performance enhancement with speech signals. However, measurements were obtained using a short impulse response (for use in automobile environments). No results were presented for an HFT in a highly reverberant venue. Recently, the Fast Affine Projection (FAP) algorithm [144] [145] has been proposed as an alternative to RLS type algorithms. The FAP can be considered a generalization of the NLMS algorithm with weight updates based on an affine projection in multiple dimensions.

**Double Talk.** Double talk (DT) occurs during periods when the far end speaker and near end speaker are simultaneously talking. The effect of DT is to increase the noise in the primary signal (similar to additive room noise described in Section 4.4.1) causing a temporary decrease in the ERLE and a slowing of the convergence and tracking ability. In a full-duplex system, it is often necessary to freeze the adaptive filter coefficients such that divergence of the tap weights does not occur. The most drastic form of DT control is push-to-talk (half-duplex or single-talk mode) which was the “de facto” standard until the advent of adaptive filters for removing echo. The literature is full of techniques for performing DT, for example [138] describes a method of detecting local speaker activity by comparing the spectral shapes of the primary and reference signals, using an appropriate distance. A large Euclidean distance is an indicator of the presence of a local talker. The method described in [146] proposes a short term cross correlation between the error output \( e(n) \) and the canceller output \( y(n) \) for controlling the step size. The correlation is used to obtain fast convergence during single-talk periods and low sensitivity during double-talk periods. Other methods are outlined in [129] and [131].
4.4.4 Complexity Issues

The choice of algorithm has an impact on the type of hardware resources needed to implement a real-time system. For a review of the impact of specific algorithm architectures on available DSP platforms see [143]. In terms of complexity, Table 4.3 summarizes the comparison between various algorithms previously introduced.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Type</th>
<th>Convergence Speed</th>
<th>Multiplications per sample</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>LMS</td>
<td>LMS</td>
<td>slow</td>
<td>$2M$</td>
<td>simple</td>
</tr>
<tr>
<td>NLMS</td>
<td>LMS</td>
<td>slow</td>
<td>$2M+1$</td>
<td></td>
</tr>
<tr>
<td>VSS</td>
<td>LMS</td>
<td>slow</td>
<td>$2M$</td>
<td></td>
</tr>
<tr>
<td>MVSS</td>
<td>LMS</td>
<td>slow</td>
<td>$2M$</td>
<td></td>
</tr>
<tr>
<td>RLS</td>
<td>RLS</td>
<td>fast</td>
<td>$2M^2+6M$</td>
<td>Complex</td>
</tr>
<tr>
<td>FTF</td>
<td>RLS</td>
<td>fast</td>
<td>$7M$</td>
<td>unstable</td>
</tr>
<tr>
<td>SFTF</td>
<td>RLS</td>
<td>fast</td>
<td>$8M$</td>
<td></td>
</tr>
<tr>
<td>LMS-IIR</td>
<td>OE</td>
<td>slow</td>
<td>$4(n_a+n_b)$</td>
<td>local minima</td>
</tr>
<tr>
<td>LMS-IIR</td>
<td>EE</td>
<td>slow</td>
<td>$2(n_a+n_b)$</td>
<td>biased minimum</td>
</tr>
<tr>
<td>FCG</td>
<td>CG</td>
<td>selectable</td>
<td>LMS to RLS</td>
<td>complexity/performance tradeoff</td>
</tr>
<tr>
<td>FNTF</td>
<td>RLS</td>
<td>selectable</td>
<td>$2M+12P$</td>
<td>complexity/performance tradeoff</td>
</tr>
<tr>
<td>FAP</td>
<td>RLS</td>
<td>selectable</td>
<td>$2M+30N$</td>
<td>complexity/performance tradeoff</td>
</tr>
</tbody>
</table>

Note: $M$ refers to the order of the filter, and $N$ refers to the projection window size and $P$ refers to the size of the predictor variables. $n_a$ and $n_b$ refer to the number of poles and zeros in an IIR structure.

4.5 Effect of Enclosure Vibration and Resonance

An important new discovery is that vibration and resonances within the enclosure can significantly limit achievable ERLE if proper acoustic design of the enclosure is not realized. In addition, rattling of the handset and keys has been observed. Recent measurements have shown that in HFT's with plastic enclosures, rattling and vibration cause a significant increase in the uncorrelated noise
signal introduced into the primary path, depending on the volume of the loudspeaker signal.

4.5.1 Experimental Results

*Experimental Results: Converged ERLE.* Figure 4.13 shows the effects that rattling and vibration have on the achievable ERLE of HFT #1. The basic loudspeaker and microphone configuration obtains the highest ERLE. The performance drops when the components are added into the enclosure. When the keys are allowed to rattle, the ERLE drops even further and finally, when the handset is placed on the set, a 10 dB reduction in ERLE is observed at 75 dB SPL as compared to the case with microphone and loudspeaker only. At the low volume levels near 50 dB SPL, the ERLE is limited primarily by the SNR of the primary signal and unbalanced two point suspension system nonlinearities within the loudspeaker.

![Graph showing experimental results in anechoic chamber](image)

*FIGURE 4.13* Effects on achievable ERLE due to vibration and rattling, HFT #1. Converged ERLE as volume is increased from 60 dB SPL to 100 dB SPL with various handset configurations. 600 tap FIR trained with the NLMS algorithm, all cases.
4.5 Effect of Enclosure Vibration and Resonance

**Primary Power Spectral Density.** The primary *power spectral density* (PSD) of HFT #1 is illustrated in Figure 4.14(a). It is clear that when the components are mounted inside the enclosure, the level of the out-of-band (distortion) signal increases substantially with an increase in the reference signal level. Figure 4.14(b) shows the PSD of the loudspeaker and microphone components when they are removed from the plastic enclosure (this removes the effect of vibration, noise and echo). *Notice that the distortion in the frequency range 4-8 kHz is significantly reduced.* However, there is still an increase in the out-of-band signal level which is essentially the nonlinear components of the original bandlimited (reference) signal. However, the level of distortion is much less than that due to vibration (shown in 4.14 (b)).

*This is an important result which tells us that vibration can be a more serious problem than loudspeaker distortion in the HFT domain.*

![Figure 4.14](image)

**FIGURE 4.14** Effect of enclosure vibration and nonlinearity. (a) HFT #1. Primary signal PSD with loudspeaker and microphone inside the HFT enclosure. Out-of-band components increase in level as the volume is increased from 60 dB SPL to 100 dB SPL. (b) same as (a) but with components removed from enclosure and mounted inside a standard baffle.
**Enclosure Effects on Impulse Response.** The early reflections in the impulse response are in part due to LREM coupling effects which are generally much larger in amplitude than the delayed echoes. Some of this coupling comes from the air path between the loudspeaker and microphone, however, *a substantial part is due to vibrational coupling in the HFT enclosure itself.* Results shown in Figure 4.15(a) illustrate the large early reflections when the transducers are mounted inside the HFT enclosure. The measurements were performed using the setup of Figure 4.2 in the anechoic chamber. Also shown in Figure 4.15 (b) is the recovered impulse response when the transducers have been removed from the HFT enclosure. The loudspeaker is placed in a standard baffle with the microphone placed 8" directly in front. The large tap values at the beginning of the LREM impulse response, which correspond to the enclosure vibration coupling are totally absent.

Direct coupling can be modelled with fixed parameters (if the parameters are known) or using the adaptive filter with a small step size for the early part of the reflection. This technique is called *Beta-grading* and is described further in [113] and [147].

**FIGURE 4.15** Experimental Results inside the anechoic chamber. (a) Impulse response of HFT#5 measured at medium volume. (b) Impulse response of the microphone and loudspeaker removed from HFT #5 and placed in a standard baffle is significantly different. Parameters: NLMS algorithm, $\alpha=0.5$, 32000 samples @ 16kHz, averaged over last 2000.
4.5.2 Microphone vibrational sensitivity

Given that enclosure vibration is a problem, a microphone element with low mechanical vibration sensitivity will reduce the vibration effect mentioned previously and minimize the magnitude of the early reflections. The mechanical sensitivity of a microphone will depend on the orientation of the microphone element: with respect to the axis of vibration and also displays a frequency dependence with the lower frequencies being more sensitive than higher frequencies. A typical electret microphone will exhibit a peak vibration response in the low frequency ranges in the vicinity of 300 Hz. Table 4.4 lists some typical measured audio sensitivities of electret microphones and Table 4.5 lists the corresponding mechanical vibrational sensitivities in dBV/G.

<table>
<thead>
<tr>
<th>Orientation</th>
<th>Microphone Sensitivity</th>
<th>Nominal Acoustic Level</th>
<th>Output Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parallel</td>
<td>-30 (to -40) dBV/Pa</td>
<td>-30 dBPa</td>
<td>-64 dBV</td>
</tr>
</tbody>
</table>

**TABLE 4.4 Electret microphone acoustic sensitivity**

<table>
<thead>
<tr>
<th>Orientation</th>
<th>Frequency</th>
<th>Sensitivity</th>
<th>Acceleration</th>
<th>Output Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parallel</td>
<td>300 Hz</td>
<td>-32 dBV/G</td>
<td>1 G</td>
<td>-32dBV</td>
</tr>
<tr>
<td>Parallel</td>
<td>1 KHz</td>
<td>-36 dBV/G</td>
<td>1 G</td>
<td>-36dBV</td>
</tr>
<tr>
<td>Perpendicular</td>
<td>1 KHz</td>
<td>-65 dBV/G</td>
<td>1 G</td>
<td>-65 dBV</td>
</tr>
</tbody>
</table>

If we model the vibrational acceleration using a one dimensional harmonic motion $x = r \cos \omega t$, the acceleration $a$ acting on the microphone element is $a = -\omega^2 r \cos \omega t$. Thus, a microphone element must travel a radius of 2.8 μm to generate 1G acceleration (9.8m/s²) at 300 Hz. It is reasonable to assume¹ that the microphone output due to vibrational coupling is not negligible when compared to the acoustical coupling. The measurements presented here (See Figure 4.14) seem to confirm this hypothesis.

¹. Based on correspondence with [148].
4.6 Effect of Transducer Nonlinearities

In this section transducer nonlinearities are investigated. Generated mainly in the loudspeaker, nonlinear distortion effectively puts a limit on the achievable ERLE when using algorithms based on linear mechanics.

4.6.1 Computer Simulation Setup

A computer simulation setup to determine the effects of distortion products on an FIR filter trained with the NLMS is described. The setup for the identification of a nonlinear system is illustrated in Figure 4.16.

![Diagram](image)

**FIGURE 4.16** Nonlinear system identification model. The system to be identified consists of a fixed or variable memoryless nonlinearity followed by an exponentially decaying impulse transfer function.

The reference training signal is white gaussian noise with a unit variance which is subsequently bandlimited by a 10th order elliptical bandpass or lowpass filter with a frequency response that closely corresponds to the characteristics of the switched capacitor transmit filters used in the experimental setup. The frequency responses are shown in Figure 4.17.

The implied sampling rate is 16 kHz, even though the filter cutoff is 3.4 kHz. The justification for
this oversampling is that significant distortion products are generated “outside” the bandwidth of the filter allowing one to obtain a rough estimate of the severity of the distortion by observing the increase in out-of-band distortion products. In the model of Figure 4.16, the distortion is generated by one of three methods given below.

**Distortion Method #1.** The first distortion model generates an output which is defined by:

\[
d(n) = \frac{a x(n) + b x^2(n) + c x^3(n)}{|a| + |b| + |c|}
\]  

(5.1)

where \(a, b\) and \(c\) refer to the magnitudes of the linear, quadratic and cubic products respectively.

**Distortion Method #2.** The second distortion model generates an output defined by:

\[
d(n) = \frac{1}{a} \tanh (ax(n)), \quad 0 < a < \infty
\]  

(5.2)

where \(a\) is a parameter which regulates the amount of squashing. For high values of \(a\), the squashing distortion becomes more severe as shown in Figure 4.18.
4.6 Effect of Transducer Nonlinearities

![Graph showing a variable hyperbolic tangent squashing function for different values of the parameter $a$.](image)

**FIGURE 4.18** Variable hyperbolic tangent squashing function for different values of the parameter $a$.

**Distortion Method #3.** The third method of generating distortion is by using the nonlinear state-space equations (2.9) through (2.12) as listed in Section 2.2. In this case, the amplitude of the reference signal is amplified before being applied to the distortion model.

### 4.6.2 Computer Simulation Results

Table 4.6 lists the distortion parameters used for the three distortion methods. The artificial room impulse is obtained from the methods outlined in Section 4.1.2 and is truncated to a length of 10 for simulation purposes. A 15th order DC-compensated NLMS algorithm with a normalized step

<table>
<thead>
<tr>
<th>Distortion Method</th>
<th>Figure</th>
<th>Distortion Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Figure 4.19(a)</td>
<td>$a=1$, $b$ varied $1e-3, 3e-3, 1, e2, 3e-2, 1e-1, 3e-1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$c=0$</td>
</tr>
<tr>
<td>1</td>
<td>Figure 4.19(a)</td>
<td>$a=1$, $b=0$, $c$ varied: $1e-3, 3e-3, 1, e2, 3e-2, 1e-1, 3e-1$</td>
</tr>
<tr>
<td>1</td>
<td>Figure 4.19(a)</td>
<td>$a=1$, $b=c=1e-3, 3e-3, 1, e2, 3e-2, 1e-1, 3e-1$</td>
</tr>
<tr>
<td>2</td>
<td>Figure 4.19(b)</td>
<td>Squashing function $a=0.1, 0.3, 0.5, 1, 1.5, 2.0$</td>
</tr>
<tr>
<td>3</td>
<td>Figure 4.19(c)</td>
<td>Reference signal amplification factor $a=0.1, 0.3, 0.5, 1, 1.5, 2.0$</td>
</tr>
</tbody>
</table>
size parameter \( \alpha = 0.5 \) is used in all the following simulations. The converged ERLE values averaged over the last 4000 iterations of a 48000 sample data file are plotted.

For distortion method #1 the coefficients \( b \) and \( c \) produce a signal-to-distortion ratio (SDR) which is computed based on the method presented in Appendix E. The simulation results for distortion methods 1, 2 and 3 are shown in Figure 4.19(a), (b) and (c).

![Graphs showing ERLE performance](image)

**FIGURE 4.19** Simulation results showing ERLE performance of an adaptive FIR filter trained with the DC-compensated NLMS algorithm. (a) Distortion model #1. (b) Distortion model #2. (c) Distortion Model #3.
Effect of Bandlimited Primary Signal for Distortion Method #3. The primary, reference and error signal PSD’s are plotted in Figure 4.20 for the case where \( a=1.5 \). In Figure 4.20 (a), the primary signal is not bandlimited. The error signal closely approximates the primary signal out-of-band, suggesting that the nonlinearity has produced out-of-band distortion products that cannot be removed by the linear adaptive filter. The converged ERLE is 18.14 dB. In Figure 4.20 (b), the primary signal is bandlimited to remove the out-of-band distortion products. However, in-band distortion products still limit the converged ERLE to 18.94 dB.

![Figure 4.20](image)

**FIGURE 4.20** Simulation results showing the PSD of the primary, reference and error signals for distortion model #3, \( a=1.5 \). (a) Unfiltered primary signal. PSD of error signal closely approximates the primary signals at the frequencies beyond the filter cutoff resulting in a converged ERLE of 18.14 dB. (b) Filtered primary signal. PSD of error signal is attenuated due to filter, however, in-band distortion products still limit ERLE to 18.94 dB.

4.6.3 Experimental Results
Figure 4.21 illustrates the PSD of the primary, reference and error signals of the transducer components of HFT #1 (mounted using the standard baffle) inside an anechoic chamber. The volume is 100 dB SPL, as measured 0.5 m from the loudspeaker. The error signal is obtained at the output of an adaptive FIR filter trained with the NLMS algorithm, \( a=0.5 \) and \( N=600 \) taps. Similar to the
4.6 Effect of Transducer Nonlinearities

Computer simulation results shown in Figure 4.20, it can be seen that the error signal closely approximates the out-of-band primary distortion components in the 4-8 kHz frequency range. The converged ERLE for this SPL averaged over the last 4000 samples of a 32000 length training vector is 18.69 dB. These results are very similar to the simulated case.

![Graph showing PSD of primary, reference, and error signals](image)

**FIGURE 4.21** Experimental results showing the PSD of the primary, reference and error signals for HFT #1 transducer components as measured inside the anechoic chamber. The error signal closely approximates the primary signals at the frequencies beyond the filter cutoff.

4.6.4 Effect of Transducer Quality

A simple test showing the effect of a combination of high and low quality loudspeakers and microphones is illustrated in Figure 4.22. The relevant parameters are listed in Table 4.7.

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
<th>Model</th>
<th>Rating</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>HQL</td>
<td>High quality Loudspeaker</td>
<td>AD4061/W8</td>
<td>8Ω·10W</td>
<td>4&quot; dia. round. Large magnet. high quality.</td>
</tr>
<tr>
<td>LQL</td>
<td>Low quality loudspeaker</td>
<td></td>
<td>8Ω·0.25W, 2&quot;round spkr.</td>
<td>Removed from HFT#4</td>
</tr>
<tr>
<td>HQM</td>
<td>High quality microphone</td>
<td>Audio Technica AT831b and power module</td>
<td>Cardioid sens. -44 dBm 200Ω.</td>
<td>&quot;High quality&quot; microphone element.</td>
</tr>
<tr>
<td>LQM</td>
<td>Low quality microphone</td>
<td>Archer 270-090</td>
<td>4.5VDC thru ext. 1kΩ S/N &gt;40 dB, Sens.=-6.5 dB</td>
<td>Electret Microphone (low cost).</td>
</tr>
</tbody>
</table>
FIGURE 4.22 Combination of high or low quality loudspeaker (HQL or LQL) and a high or low quality microphone (HQM or LQM) on the ERLE performance as a function of loudspeaker power. The loudspeaker quality is the major component affecting achievable ERLE and the microphone quality is secondary. Parameters: 1000 tap dc compensated NLMS, $\alpha=0.5$.

The ERLE averaged over the last 4000 iterations of a 48000 sample data file are plotted. The results show that the loudspeaker has the major effect on the achievable steady ERLE performance. The microphone has some effect on achievable ERLE but this is secondary with respect to the loudspeaker. These measurements were made in an anechoic chamber with the loudspeaker mounted in a standard baffle and the microphone placed 30 cm directly in front of the loudspeaker using a microphone boom. This measurement technique removes the effects of room noise, enclosure vibration, echoes and direct coupling. The important observation is that the converged ERLE is low at both the high volume end (where nonlinearity is significant), and the low volume end, where circuit noise and two point suspension effects become significant. The minimum modelling error occurs in the middle volume ranges.
4.7 Summary and Discussion

The chapter has addressed the acoustic echo cancellation problem in HFT’s. A review of the performance requirements has determined the following:

- Both speech enhancement and noise reduction techniques need to be employed to obtain an overall subjective improvement.
- Transducer nonlinearities need to be addressed to obtain the high ERLE values cited in the performance recommendations.
- Acoustic reverberation is shown to be an extremely complicated process that can only be modelled under statistical considerations.

Section 4.2 presents the measurement setup and illustrates that potential problems such as circuit noise, grounding, and cross-talk must be treated before making measurements to ensure that the minimum MSE obtained using a particular algorithm is due to the algorithm itself and not limitations in the test setup. Experiments have confirmed the maximum converged ERLE of the basic setup is approximately 56 dB.

In Section 4.3, the study of the application of IIR structures to the AEC problem concludes as follows:

- IIR model approximations do not lead to significant decreases in the number of parameters (compared to FIR models) necessary to obtain the same ERLE values.
- The Hankel norm approximation error bound for IIR filters has similar characteristics to the TIP/TP ratio, and suggest that IIR structures are not well suited to modelling typical LREM’s.

In Section 4.4, various performance limitations are studied, with conclusions as follows:

- An analysis of noise, finite precision effects and quantization shows that these limitations are generally far below the typical ERLE levels encountered in real world AEC’s.
4.7 Summary and Discussion

- There seems to be a close correlation between the ERLE limits predicted by the TIP/TP ratio and the Hankel error bound.

In Section 4.5, the effects of enclosure vibration and resonances are studied, with the following conclusions:

- Rattling keys and handsets on the terminal have a measurable effect on the achievable ERLE. Limiting the movement of keys and handsets generally improves the achievable ERLE.

- Results have been presented which illustrate that direct coupling between the microphone and loudspeaker due to the enclosure radically changes the impulse response.

- Enclosure vibration can be a more serious problem than loudspeaker distortion as measured in a number of commercially available HFT's. However, it is highly dependent on the type of HFT enclosure.

In Section 4.6, the effects of transducer nonlinearity are studied. The conclusions are summarized as follows:

- In the absence of other limitations, loudspeaker nonlinearity will limit the achievable steady state ERLE in linear structures. The ERLE value will be determined approximately as the ratio of the powers of the nonlinear distortion products to the linear products within the primary signal.

- Distortion products manifest themselves as harmonics which will be generated both within and outside the bandwidth of interest.

- There is a high degree of correlation between the uncancelled error signal and the out-of-band distortion products in both simulated and experimental results using linear structures.

- Given that the vibration and resonances can be removed through proper acoustic design, improvements in transducer quality will also improve the achievable ERLE.
Summary of Steady-State Performance Limitations.

- Undermodelling of the acoustic transfer function is directly related to the number of taps in the adaptive filter, and will be the major limitation when the number of taps is insufficient to cover the LREM impulse response.

- Given a sufficient number of taps, the limitations to the steady-state ERLE are in order of importance: (1) enclosure vibration and/or nonlinear distortion, (2) room noise (in a typical conference room) and (3) DSP and circuit noise.

- The effect that vibration and loudspeaker nonlinearity have on the achievable ERLE is highly dependent on the frequency and volume of the reference signal as well as the type of HFT being used.

An illustration showing the relative contributions is illustrated in Figure 4.23.

![Graph showing the relative contributions of various limitations to ERLE](image)

**FIGURE 4.23** Achievable ERLE as a function of physical limitations.
Chapter 5

Nonlinear Structures For Acoustic Echo Cancellation

This chapter presents several new architectures consisting of cascaded nonlinear and linear sections for the identification of nonlinear systems with memory and dispersion. The specific application is for improved ERLE performance for nonlinear echo cancellation in the HFT domain.

In the first two subsections the Volterra and multi-layer neural network filter models are applied to the identification of both simulated and experimental data. The results presented form a baseline for comparison to the new architectures presented in subsequent sections. Next, the following new structures are developed: (1) A parallel cascaded neural network-FIR structure with a mixed linear-sigmoid activation function; (2) A TDNN structure that uses fully adaptive activation functions in addition to the variable weights in the hidden layers; (3) A cascaded synaptic FIR neural network including the adaptive activation functions described previously. The learning rules are derived by modifying the gradient backpropagation algorithm for the specific architecture.
5.1 The Adaptive Volterra Filter

Adaptive Volterra filters have been applied to compensation of low frequency loudspeaker nonlinearities [4][13], however, all the loudspeakers tested in the literature are "woofer" designs and the author could find no literature on the application of Volterra filters to small loudspeakers, such as those encountered in an HFT. To this end, data obtained from an HFT inside a typical conference room, as well as HFT audio transducers (as measured inside an anechoic chamber) are applied to a nonlinear adaptive Volterra filter to determine how well it is suited for this application. Simulation results comparing the LMS-Volterra and NLMS-FIR filters on the distortion models from Section 4.6.1 are first examined.

5.1.1 Simulation Examples

A 3rd order adaptive LMS-Volterra filter is constructed to study how it behaves while attempting to model nonlinearity as generated using the artificial distortion methods of Section 4.6.1. Tap updates are based on the Volterra algorithm in Appendix D.8. Results are compared to the NLMS-FIR filter. For each of the simulation results, the filters are allowed to train for 64,000 samples and the converged ERLE results are obtained from the average of the last 4000 points.

Distortion Model #1, combined quadratic and cubic distortion. In this test the following parameters are used: \( N_1 = N_2 = N_3 = 10 \), \( \alpha = 0.5 \), \( \mu_1 = 0.1 \), \( \mu_2 = 0.01 \), \( a = 1 \), \( b = c = 0.2 \).

The results are shown in Figure 5.1 with a performance summary given in Table 5.1. The reference signal \( r(n) \) in this test case is a uniformly distributed white noise source with unit variance, which is either filtered first or applied directly to the distortion generator and dispersive LREM, similar to the method in Figure 4.16.
5.1 The Adaptive Volterra Filter

(a) Converged ERLE vs. SDR, unfiltered r(n).

(b) Convergence curve for highest distortion level, unfiltered r(n).

(c) Converged ERLE vs. SDR, filtered r(n).

(d) Convergence curve for highest distortion level, filtered r(n).

FIGURE 5.1 Simulation results. Volterra identification of a nonlinear system generated using distortion model #1 (a) converged ERLE vs. SDR, unfiltered r(n). (b) convergence curve for highest distortion level, unfiltered r(n). (c) converged ERLE vs. SDR, filtered r(n). (d) convergence curve for highest distortion level, filtered r(n).

TABLE 5.1 Summary of simulation results shown in Figure 5.1.

<table>
<thead>
<tr>
<th>Reference Filtering</th>
<th>Figure</th>
<th>Converged ERLE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NLMS-FIR</td>
<td>LMS-Volterra</td>
</tr>
<tr>
<td>Unfiltered</td>
<td>Figure 5.1(a), (b)</td>
<td>17.9 dB</td>
</tr>
<tr>
<td>BPF filtered</td>
<td>Figure 5.1(c), (d)</td>
<td>15.3 dB</td>
</tr>
</tbody>
</table>

The results shown in Figure 5.1 (a) and (b) illustrate the convergence curves for the unfiltered case.
5.1 The Adaptive Volterra Filter

for different distortion parameters, where \( b=c \) are varied. The Volterra filter can easily identify the unknown system. However, the results in Figure 5.1 (c) and (d) also indicate that when the signal is filtered, the performance of the LMS-Volterra filter is degraded.

**Distortion Model #2 and #3.** The simulation results for distortion model #2 and #3 are shown in Figure 5.2(a) and (b). For distortion model #2, \( N_1=N_2=N_3=10 \). For distortion model #3, \( N_1=N_2=N_3=20 \). For both cases, \( \alpha_j=0.5, \mu_2=1e-1 \) and \( \mu_3=1e-2 \).

![Converged ERLF Using a Volterra Filter](image)

**FIGURE 5.2** Simulation results. Converged ERLF using a Volterra filter for the identification of a simulated nonlinear system generated using (a) distortion method #2 (b) distortion method #3.

In the simulations presented above, the Volterra filter consistently outperforms the linear FIR filter.

5.1.2 Experimental Results

A fully connected 3rd order adaptive Volterra filter with \( m_1=1000, m_2=100 \) and \( m_3=40 \) is constructed in an attempt to model real-world loudspeaker nonlinearity in a typical AEC configuration. Tap updates are based on the algorithm in Appendix D.8 with a normalized step size parameter \( \alpha_1=0.5, \mu_2=1e-1 \) and \( \mu_3=1e-2 \). Three experiments are conducted, using data which was recorded in different venues using different HFT’s. For all experiments, the sound source is filtered.
noise, as per the test set-up described in Section 4.2.3, recorded at a volumes ranging from 55 to 100 dB SPL as measured at 0.5 m directly above the loudspeaker.

**HFT #1, Anechoic Chamber, with and without enclosure.** The loudspeaker and microphone from HFT #1 are removed and placed in an anechoic chamber for the first set of measurements. Subsequently, they are placed back inside the HFT enclosure for a second round of measurements. The results shown in Figure 5.3(a) illustrate that approximately 6 dB improvement in converged ERLE can be obtained using the LMS-Volterra filter compared to a1000 tap NLMS-FIR filter for identifying an LREM consisting of the audio transducers only. In Figure 5.3(b) the converged ERLE vs. volume is shown for HFT#1 with the components mounted inside the enclosure. Notice that the LMS-Volterra filter is no longer able to obtain any gains over the FIR filter. This is due to the vibration limits as discussed in Section 4.5. The converged weights of the Volterra filter are illustrated in Figure 5.4(a), (b) and (c) which show the weight values obtained for the linear, quadratic and cubic sections respectively. The *difference* between the linear weights obtained with the
NLMS algorithm is also shown in Figure 5.4(d).

FIGURE 5.4 Volterra tap weights for HFT #1 at 100 dB SPL. (a) Linear tap weights, (b) quadratic tap weights. (c) Cubic tap weights. (d) difference between Volterra and FIR linear tap weights.

**HFT #6 in Conference Room #2.** HFT #6 is stiffened against vibration. The results shown in Figure 5.5 indicate that using this HFT a 7.1 dB performance improvement over the NLMS-FIR filter may be obtained.
5.1 The Adaptive Volterra Filter

![Graphs showing ERLE vs. SPL for HFT #6 results.](image)

**FIGURE 5.5 HFT #6 results.** (a) Converged ERLE vs. SPL. (b) Convergence curves for data at 90 dB SPL.

A summary of the experimental results showing the regions of greatest improvement over the FIR filter structure is shown in Table 5.1.

**TABLE 5.2 Summary of experimental parameters and results for the Volterra filter.**

<table>
<thead>
<tr>
<th>Experiment Location</th>
<th>Figure</th>
<th>Components</th>
<th>SPL Volume ([\text{dB}])</th>
<th>Converged ERLE</th>
<th>Improvement over FIR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anechoic</td>
<td>Figure 5.3 (a)</td>
<td>HFT #1 Components</td>
<td>100</td>
<td>27.3 dB</td>
<td>33.5 dB</td>
</tr>
<tr>
<td>Anechoic</td>
<td>Figure 5.3 (b)</td>
<td>HFT #1 Components</td>
<td>95</td>
<td>28.6 dB</td>
<td>28.8 dB</td>
</tr>
<tr>
<td>Conference</td>
<td>Figure 5.5 (a)</td>
<td>HFT #6 Components</td>
<td>90</td>
<td>21.2 dB</td>
<td>28.3 dB</td>
</tr>
</tbody>
</table>

5.1.3 Discussion

The LMS-Volterra filter is easily able to outperform the NLMS-FIR filter by several 10's of dB when presented with *simulated* data. However, the convergence rate may be severely limited by the increased eigenvalue spread of the input data, which becomes prohibitive as the number of filter parameters increases.

Assuming an LMS update, the Volterra filter will have an approximate complexity of \(2M_e+2\)
where $M_e$ is the dimension of the extended weight vector according to equation (3.59). Assuming a 1000 tap input from which 100 taps and 40 taps are used to obtain the quadratic and cubic sections respectively, the length of the extended vector will be will be 17530 weights, resulting in a staggering complexity of approximately 35062 multiplications per iteration.

For nonlinear AEC's using experimental data, it appears that the nonlinear Volterra filter can improve the ERLE significantly, for example by 7.1dB using HFT #6. However these results are highly dependent on the type of HFT being tested. An improvement could only be obtained when either (i) audio components were isolated (and vibration was not a limiting factor) or (ii) HFT #6 was used (HFT#6 is stiffened against vibration). The experimental results also show that the convergence is quite slow, and given the added complexity, this technique has a disadvantage for real-time applications. These results suggest that alternative models and structures should be investigated.

Experimental results also show that there is very little difference between the linear weights obtained using either the FIR or Volterra filter, and that only a small percentage of the nonlinear weights are significant. Although results are not presented here showing nonlinear tap weights for other volumes, analysis of experimental data at other volumes has determined that they do change significantly with a change in applied volume. A set of nonlinear weights for one volume does not necessarily correspond to the weights obtained at a different volume. This means that a significant number of higher order weights must be retained to obtain a good modelling accuracy, however, this results in an overwhelming number of weights and poor convergence. An "on-line" technique for selecting the significant weights and pruning nonsignificant weights would be advantageous for obtaining reduced complexity, however this is beyond the scope of this thesis.
5.2 Neural Network Adaptive Filter

As an alternative to Volterra filters, artificial neural network (ANN) filters can be used to perform nonlinear adaptive filtering. The "conventional" neural network filter structure uses a tapped delay line (TDL) at the input to a static MLP as illustrated in Figure 5.6. Waibel et al. first proposed this structure in [46] and referred to it as a time delay neural network (TDNN). The structure is defined by the nomenclature \((n_i, h_1, h_2, n_o)\) which refers to the number of input nodes (i.e. number of inputs from the TDL), the number of neurons in the first and second hidden layers, and the number of output nodes respectively. For all the results that follow, \(n_o\) is set equal to 1 and each node in the hidden layer has a sigmoid function defined by equation (3.6) with \(a=1\). The output node is a linear summation and the activation potential to each sigmoid includes an adjustable bias.

![Figure 5.6 Time delay neural network (TDNN) filter.](image)

5.2.1 Computer Simulation Examples

A two layer TDNN filter is applied to the three distortion models described in Section 4.6.1. Tap updates are based on the gradient BP method using a normalized step size (with respect to the power in the TDL) with \(\alpha=0.5\). For each of the simulation results, the algorithms are allowed to
train for 64,000 samples and the converged ERLE results are obtained from the average of the last 4000 points.

**Simulation #1: Selection of number of hidden nodes for a two layer TDNN.** In this test a comparison of the two layer TDNN and FIR filters is presented using a 10 tap input with a normalized step size of 0.2, and distortion model #1. The reference signal $r(n)$ in this test case is a uniformly distributed white noise source with unit variance, which is applied directly to the distortion generator (quadratic and cubic distortion) and dispersive LREM, similar to the method in Figure 4.16. The number of hidden nodes is first selected by trying a number of different architectures based on the models $(10,n_1,1)$ where $n_1$ represents the number of hidden nodes. The results showing converged ERLE are shown in Figure 5.7 for different $n_1$ with quadratic/cubic distortion generated according to method #1, for high level distortion ($a=b=c=1$) and low level distortion ($a=1, b=c=0.2$). Figure 5.7(a) shows the result when an unfiltered reference is used and Figure 5.7(b) show the results for a bandpass filtered reference.

![Unfiltered Reference, Low/High distortion](image1.png)

![Filtered Reference, Low/High distortion](image2.png)

**FIGURE 5.7** Simulation #1 results showing converged ERLE vs. increasing number of hidden nodes for a $(10,n_1,1)$ TDNN. (a) Unfiltered reference (b) Filtered reference.

Based on these results, a $(10,5,1)$ model was chosen for subsequent tests as a good compromise.
between complexity and performance. The results of Figure 5.7 show that there is very little difference between the performance of the TDNN structure when the reference data is filtered or unfiltered. For "low level" distortion with \( b=c=0.2 \), the simulated TDNN performance in Figure 5.7 is between 6 and 7 dB higher than the equivalent FIR structure.

**Simulation #2: Distortion models #1, #2 and #3.** The simulation results shown in Figure 5.8 (a) (b) and (c) are obtained using the methods in Section 4.6 and distortion models #1, #2 and #3 respectively. Again, the TDNN structure obtains a higher ERLE value in the high distortion ranges, but is generally not as good as the FIR-NLMS structure at low distortion levels.

**Simulation #3: Comparison of two and three layer TDNN.** In this simulation, a comparison of the number of parameters for two and three layer TDNNs is done for the case where the artificial room impulse response is of length 500. In this simulation, distortion model #1 is used with \( a=1, \ b=c=0.2 \). Results shown in Table 5.3 summarize the results.

<table>
<thead>
<tr>
<th>LREM order</th>
<th>Type</th>
<th>Size</th>
<th>Step Size</th>
<th>Converged ERLE [dB]</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>FIR</td>
<td>(500 tap)</td>
<td>0.5</td>
<td>16.3</td>
</tr>
<tr>
<td>500</td>
<td>2 hidden layers</td>
<td>TDNN (500,5,5,1)</td>
<td>0.5</td>
<td>15.1</td>
</tr>
<tr>
<td>500</td>
<td>2 hidden layers</td>
<td>TDNN(500,5,2,1)</td>
<td>0.5</td>
<td>15.5</td>
</tr>
<tr>
<td>500</td>
<td>2 hidden layers</td>
<td>TDNN(500,2,2,1)</td>
<td>0.5</td>
<td>16.0</td>
</tr>
<tr>
<td>500</td>
<td>1 hidden layer</td>
<td>TDNN(500,5,1)</td>
<td>0.5</td>
<td>16.9</td>
</tr>
<tr>
<td>500</td>
<td>1 hidden layer</td>
<td>TDNN(500,2,1)</td>
<td>0.5</td>
<td>17.3</td>
</tr>
<tr>
<td>500</td>
<td>1 hidden layer</td>
<td>TDNN(500,1,1)</td>
<td>0.5</td>
<td>17.5</td>
</tr>
</tbody>
</table>

The three layer network consistently performs **worse** than the FIR linear counterpart and suggests that in the context of nonlinear AEC application, **networks with a single hidden layer are suitable to the task.**
5.2 Neural Network Adaptive Filter

FIGURE 5.8 Simulation results for (a) distortion method #1, (b) distortion method #2 and (c) distortion method #3. A $(10,5,1)$ TDNN is used for (a) and (b) and a $(15,5,1)$ TDNN is used for (c) to cover the impulse response generated by method #3.

5.2.2 Development of a Mixed Linear-Sigmoid Activation Function

A neural network filter will generate a finite amount of distortion due to the nonlinear nature of the sigmoid. In some simulations, the TDNN performs worse or only slightly better than a conventional FIR adaptive filter, especially at low distortion values as demonstrated in Figure 5.7 and Figure 5.8. In order to mitigate this effect, a mixed linear-sigmoid activation function is proposed which has a linear portion in the middle. The nonlinear node consists of a linearized hyperbolic
tangent function which is linear for inputs below a user definable amplitude $p$, where $0 \leq p \leq 1$. By setting the parameter $p$ it is possible to reduce the modelling error by a few dB compared to a conventional (i.e. $p=0$) sigmoid\(^1\). The node activation function $\varphi(s,p)$ is defined by:

\[
\varphi(s,p) = \begin{cases} 
  s & |s| \leq p \\
  \text{sign}(s) \left[ (1-p) \cdot \tanh\left( \frac{|s|}{1-p} \right) + p \right] & |s| > p
\end{cases}
\]  

(5.1)

where $s$ is the input. Differentiating (5.37) with respect to $s$, we obtain the slope of the activation function:

\[
\varphi'_s(s,p) = \begin{cases} 
  1 & |s| \leq p \\
  \left[ 1 - \tanh^2(\theta) \right] & |s| > p
\end{cases}
\]  

(5.2)

where $\theta = \left( \frac{|s|}{1-p} \right)$

Figure 5.9 shows the activation function of equation (5.37) with values of $p$ equal to 0.0, 0.5, and 0.9, along with the associated $\varphi'_s(s,p)$ values.

---

\(^1\) Improvements depend on the severity of the nonlinear distortion.
For data that is weakly nonlinear, the weights in the TDNN will adjust to provide an activation in the linear region of the sigmoid, and thus offer improved performance. Simulation results showing the effect of varying the linear region versus converged ERLE are shown in Figure 5.10 using distortion model #1 when $a=1$, $b=c=0.01$ (low distortion) and $b=c=0.5$ (high distortion). The optimum value of linear region is highly dependent on the severity of nonlinearity encountered, however, based on these results, the linear region $p$ was set to either 0.0 or 0.5 in subsequent simulations, as a good compromise between the two extremes. A better solution is to implement an adaptive activation function, and this is discussed further in Section 5.5.

![Linear Region Variation: Distortion Model #1](image)

FIGURE 5.10 Effect of changing linear region in a mixed linear-sigmoidal activation function. (10,5,1) TDNN on distortion model #1 with $a=1$ and $b=c=0.01$ (low distortion) and $b=c=0.5$ (high distortion)

### 5.2.3 Experimental Results

A fully connected two-layer TDNN is used to model the loudspeaker nonlinearity in a typical AEC configuration. Tap updates are based on the BP algorithm with a normalized step size parameter $\alpha=0.5$. For the experiments that follow, the sound source is filtered noise, as per the test set-up
described in Section 4.2.3, recorded at various volumes ranging from 50 to 100 dB SPL as measured at 0.5 m directly above the loudspeaker.

**HFT #3, Anechoic Chamber, no enclosure.** The results in Figure 5.11 indicate that a significant difference in performance comes about as a result of changing the "linear" region of the sigmoid activation function, for the case where 300 taps are used in the input TDL. For example, by changing $p$ from 0.5 to 0.0, an improvement of 4.8 dB ERLE is seen in the 100 dB SPL range, but a degradation of 1.9 dB is observed at 75 dB SPL. Another significant result is that a $(1000,1,1)$ structure performs worse than a $(300,1,1)$ TDNN structure, all other parameters being the same.

![Graph](image)

**FIGURE 5.11 HFT #3 experimental results, anechoic chamber, components only.**

**HFT #1, Anechoic Chamber, with and without enclosure.** The results shown in Figure 5.12 (a) are for the isolated components case and illustrate that a marginal improvement in converged ERLE can be seen using a $(300,1,1)$ TDNN as compared to the standard 1000 tap NLMS-FIR filter. However, at 75 dB SPL, an improvement of approximately 4 dB is obtained. Note that this level of improvement was not achieved using the Volterra filter structure (refer to Figure 5.3 a). The results for the transducers mounted inside the HFT enclosure are shown in Figure 5.12 (b). The
FIGURE 5.12 HFT #1 results, anechoic chamber, TDNN. Converged ERLE vs. SPL for (a) separately mounted transducer components and (b) with components mounted inside the HFT enclosure.

TDNN structure has difficulty in identifying the HFT transfer function, but again, this can be attributed to the vibration limits as discussed in Section 4.5

**HFT #6 in Conference Room #2.** HFT #6 is stiffened against vibration. The results shown in Figure 5.13 indicate that performance is better than the NLMS-FIR filter for volumes greater than 57 dB SPL when using an HFT #6. Over 5 dB improvement is observed at 65 dB SPL.

FIGURE 5.13 HFT #6 results for a 2 layer TDNN. (a) Converged ERLE vs. SPL for the NLMS, (1000,1,1) and (200,1,1) structure. (b) Convergence curve for data at 65 dB SPL.
A summary of the experimental results showing the areas of greatest improvement is given in Table 5.1.

**TABLE 5.4 Summary of experimental results showing regions of greatest improvement.**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Experiment /Location</th>
<th>HFT</th>
<th>TDNN parameters</th>
<th>SPL</th>
<th>Converged ERLE</th>
<th>Improvment over FIR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 5.12(a)</td>
<td>Anechoic</td>
<td>HFT #1 Components</td>
<td>(300,1,1)</td>
<td>0.5</td>
<td>75</td>
<td>1000 tap FIR/NLMS</td>
</tr>
<tr>
<td>Figure 5.12(b)</td>
<td>Anechoic</td>
<td>HFT #1 Components</td>
<td>(1000,1,1)</td>
<td>0.5</td>
<td>70</td>
<td>TDNN/BP</td>
</tr>
<tr>
<td>Figure 5.11</td>
<td>Anechoic</td>
<td>HFT #3 Components</td>
<td>(300,1,1)</td>
<td>0</td>
<td>100</td>
<td>18.9 dB</td>
</tr>
<tr>
<td>Figure 5.13</td>
<td>Conference</td>
<td>HFT #6</td>
<td>(1000,1,1)</td>
<td>0.5</td>
<td>65</td>
<td>32.8 dB</td>
</tr>
<tr>
<td>Figure 5.13</td>
<td>Conference</td>
<td>HFT #6</td>
<td>(200,1,1)</td>
<td>0.1</td>
<td>95</td>
<td>17.1</td>
</tr>
</tbody>
</table>

5.2.4 TIP/TP Performance for the TDNN in the Undermodelled Case

Experimental data was applied to the TDNN structure to determine the optimum length for the TDNN section for the highest volume (100 dB SPL) case. The results shown in Figure 5.14 illustrate that for a system with undermodelling of the impulse length, a TDNN structure has improved ERLE performance compared to the NLMS-FIR filter. The experimental data was obtained from HFT #3 components in an anechoic chamber. The TDNN model is an \( (n_0,2,3,1) \) structure. The best performance occurs when the number of input taps \( n_0 = 150 \) taps. Here the difference between the TDNN and FIR ERLE value is approximately 5.5 dB.
5.2 Neural Network Adaptive Filter

![Graph: Converged ERLE at 100 dB SPL for the Undermodulated Case]

FIGURE 5.14 HFT #3 components (i.e. no enclosure), anechoic chamber. In an undermodelled state a TDNN obtains a higher ERLE as compared with the FIR structure.

5.2.5 Discussion

The simulation results presented in Section 5.2.1 indicate that the TDNN structure is capable of obtaining improved steady state ERLE performance when the nonlinear distortion becomes significant. For low distortion, the TDNN is unable to model the nonlinearity as well as the linear FIR filter.

The complexity breakdown for a two layer TDNN is listed in Table 5.5.

**TABLE 5.5 TDNN complexity assuming a single node output.**

<table>
<thead>
<tr>
<th>Equation Number</th>
<th>Description</th>
<th>Layer</th>
<th>Mults.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(3.60)</td>
<td>vector dot product</td>
<td>Input</td>
<td>$n_0 \times n_1$</td>
</tr>
<tr>
<td>(3.67)</td>
<td>weight update</td>
<td>Input</td>
<td>$n_0 \times n_1 + 2$</td>
</tr>
<tr>
<td>(3.67)</td>
<td>bias update</td>
<td>Hidden</td>
<td>$n_1$</td>
</tr>
<tr>
<td>(3.60)</td>
<td>vector dot product</td>
<td>Hidden</td>
<td>$n_1$</td>
</tr>
<tr>
<td>(3.67)</td>
<td>weight update</td>
<td>Hidden</td>
<td>$n_1 + 2$</td>
</tr>
<tr>
<td>(3.67)</td>
<td>bias update</td>
<td>Output</td>
<td>1</td>
</tr>
<tr>
<td>(3.68)</td>
<td>delta calculation</td>
<td>Output</td>
<td>1</td>
</tr>
<tr>
<td>(3.68)</td>
<td>delta calculation</td>
<td>Hidden</td>
<td>$n_1 + 2$</td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td></td>
<td>$4 \times n_1 + 2 \times n_0 \times n_1 + 8$</td>
</tr>
</tbody>
</table>

Note: $n_0$ equals the number of input nodes (TDL length), and $n_1$ equals the number of hidden layer nodes.
5.2 Neural Network Adaptive Filter

If we assume a 1000 tap input, with a 2 node hidden layer (i.e. \( n_D = 1000 \), \( n_I = 2 \)), the complexity is approximately 4015 multiplications per iteration.

Simulations using LREM's of length 500 show that a TDNN with a single hidden layer is preferable over one with two hidden layers, and that a single node in the hidden layer was sufficient to obtain ERLE improvements. From the perspective of nonlinear AEC application, this is beneficial since such structures have a complexity comparable to the NLMS-FIR filter.

The use of an activation function with a definable linear region is beneficial in low distortion environments. Simulation results have shown that for low distortion environments, a large linear region is preferable, whereas for high distortion environments, the converse is true, and suggests that some form of adaptive control of the activation function may be beneficial.

Experimental results in Section 5.2.3 show that a two layer TDNN can obtain improvements in converged ERLE between 2.8 and 5.6 dB at medium to high volumes where the distortion is greatest, but does not perform as well as the linear FIR filter at low volumes. This is also observed in the simulations.

When HFT's contain significant vibration due to poor enclosure design, the TDNN structure is unable to obtain any clear improvements in ERLE, similar to results presented for the Volterra case. However, where the HFT components are isolated, or where the HFT design mitigates the vibration problem, the TDNN can improve the ERLE by up to 5.6 dB compared to the NLMS-FIR filter.

TDNN performance also depends on the type of HFT being used. For example, at 65 dB SPL, a (1000,1,1) TDNN structure will yield a net ERLE gain of 5.3 dB for HFT #6 but can have a net loss of 3.8 dB for HFT #1.
A surprising new result demonstrated in Section 5.2.4 is that a simple multilayer TDNN structure is capable of obtaining improved ERLE convergence over that of a linear FIR structure for the undermodelled case and suggests an intelligent way of combining low order TDNNs with linear FIR structures for obtaining improved performance in both the low and high volume ranges.

5.3 Comparison of Results - Volterra vs. TDNN

Complexity. For low order systems, the Volterra network is an attractive option for dealing with weak nonlinearities. However, in AEC applications, the system orders make the complexity prohibitively large. The TDNN offers comparable performance at a fraction of the cost of a fully connected Volterra filter. For example, a 1000 tap NLMS-FIR filter will have a complexity of approximately 2004 multiplications per iteration. The TDNN structure has a computational requirement of 4015, approximately twice that of the LMS algorithm. By comparison, the Volterra complexity is 17 to 20 times that of the LMS algorithm.

Convergence. The Volterra algorithm has an initial convergence comparable to the NLMS algorithm but slows down considerably as the MSE decreases. The TDNN structure has a slower initial convergence than the NLMS algorithm but can obtain a steady state MSE faster than the Volterra algorithm. This suggests that a method of improving the convergence, without sacrificing tracking ability would be advantageous.

Modelling Accuracy. The Volterra filter obtains slightly better error performance than the TDNN structure for most of the cases investigated. For low volumes, the Volterra filter obtains the same performance as the linear FIR structure. The TDNN structure on the other hand has difficulty in obtaining the same performance as the linear FIR structure at low to medium volumes as predicted by the computer simulations. However, the TDNN does offer significant improvements in the mid
and high volume ranges and suggests that a combination of linear and nonlinear architectures might be required for obtaining good performance both in low and high volumes. This is the topic of the next section.

5.4 Two Stage Neural Filter

In this section a new nonlinear adaptive filter for improving the echo cancellation performance at both low and high volumes for handsfree telephones is proposed. The proposed structure shown in Figure 5.15 consists of both nonlinear and linear sections and is constructed based on observations from the previous sections. The nonlinear section consist of a two layer neural network that cancels the first part of the AIR where most of the energy is contained. The linear section effectively ensures that residual signals not cancelled by the nonlinear section are accurately modelled. Thus each section looks at a different segment in time.

The weight update equations for the nonlinear portion are based on the gradient backpropagation algorithm with a normalized adaptive step size. The nonlinear node is defined by equation (5.1) and the linear region parameter $p$ was set to 0.2 since it was found that this produced an ERLE approximately 1.5 dB higher\(^1\) than with a conventional (i.e. $p=0$) sigmoid for the lower volumes.

5.4.1 Weight Update Equations

In Figure 5.15, the output $y(k)$ of the neural network portion at time $k$ is defined by:

$$y(k) = w^{(2)}(k)x^{(2)}(k) + w_b^{(2)}(k) \quad (5.3)$$

$$x^{(2)}(k) = \varphi(s(k)) \quad (5.4)$$

\(^1\) Based on several tests varying the linear region using field data collected using HFT #6.
5.4 Two Stage Neural Filter

FIGURE 5.15 Proposed two stage nonlinear AEC structure consists of both nonlinear and linear sections.

\[ s(k) = w^{(1)}(k) T x^{(1)}(k) + w^{(1)}_b(k) \]  

(5.5)

where \( x^{(l)}(k) \) represents the input vector to layer \( l \), \( w^{(l)}(k) \) represents the weight vector in layer \( l \), \( w^{(l)}_b(k) \) represents the single bias weight for layer \( l \), \( s(k) \) represents the input to the nonlinear node and \( T \) is the transpose operator. The weight update equations are described by:

\[ w^{(l)}(k+1) = w^{(l)}(k) - \mu_{TDNN}(k) \delta^{(l+1)}(k) \cdot x^{(l)}(k) \]  

(5.6)

\[ w^{(l)}_b(k+1) = w^{(l)}_b(k) - \mu_{TDNN}(k) \delta^{(l+1)}(k) \]  

(5.7)
\[ \delta^{(l+1)}(k) = \begin{cases} -2e_1(k) & : l = 2, \text{output layer} \\ \varphi'(s(k)) \delta^{(l+2)}(k) w^{(l+1)}(k) & : l = 1, \text{hidden layer} \end{cases} \quad (5.8) \]

where \( e_1(k) = p(k) - y(k) \), \( \varphi'(\cdot) \) represents the derivative of the activation function at the input value \( s(k) \), \( \delta^{(l+1)}(k) \) represents the local gradient "delta" term in layer \( l+1 \), and \( \mu_{TDNN}(k) \) is the normalized step size parameter defined by:

\[ \mu_{TDNN}(k) = \frac{\alpha}{2 + x^{(1)}(k) \frac{\varphi'(s(k))}{\frac{T}{x^{(0)}(k)} + \left[x^{(2)}(k)\right]^2}} \quad (5.9) \]

The parameter \( \alpha \) is a number between 0 and 2, and is set to 0.5. The weights in the linear portion of the proposed structure are updated using the NLMS algorithm:

\[ w_{FIR}(k+1) = w_{FIR}(k) + \left[ \frac{\alpha}{1 + x_{FIR}(k) \frac{T}{x_{FIR}(k)}} \right] e_2(k) \cdot x_{FIR}(k) \quad (5.10) \]

\[ w_b(k+1) = w_b(k) + \left[ \frac{\alpha}{1 + x_{FIR}(k) \frac{T}{x_{FIR}(k)}} \right] e_2(k) \quad (5.11) \]

**Complexity.** The complexity of this algorithm is obtained by summing up the number of calculations in the weight update equations according to Table 5.6. Here, the division needed to compute the normalized step size is counted as one operation, the sigmoid function and deltas are assumed to be obtained from a ROM lookup table in DSP hardware, and \( n_1 \) and \( n_2 \) refer to the lengths of the TDLs in the nonlinear and linear sections respectively.

**TABLE 5.6 Complexity of the two stage neural filter.**

<table>
<thead>
<tr>
<th>Equation Number</th>
<th>Description</th>
<th>Layer</th>
<th>Multiplications or Divisions</th>
</tr>
</thead>
<tbody>
<tr>
<td>(5.3)</td>
<td>output</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>(5.4)</td>
<td>sigmoid</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>(5.5)</td>
<td>vector dot product</td>
<td>1</td>
<td>( n_1 )</td>
</tr>
</tbody>
</table>
5.4 Two Stage Neural Filter

**TABLE 5.6 Complexity of the two stage neural filter.**

<table>
<thead>
<tr>
<th>Equation Number</th>
<th>Description</th>
<th>Layer</th>
<th>Multiplications or Divisions</th>
</tr>
</thead>
<tbody>
<tr>
<td>(5.6)</td>
<td>weight update</td>
<td>Hidden</td>
<td>$n_1+2$</td>
</tr>
<tr>
<td>(5.7)</td>
<td>bias update</td>
<td>Hidden</td>
<td>1</td>
</tr>
<tr>
<td>(5.8)</td>
<td>delta calculation</td>
<td>Hidden</td>
<td>3</td>
</tr>
<tr>
<td>(5.6)</td>
<td>weight update</td>
<td>Output</td>
<td>2</td>
</tr>
<tr>
<td>(5.7)</td>
<td>bias update</td>
<td>Output</td>
<td>1</td>
</tr>
<tr>
<td>(5.8)</td>
<td>delta calculation</td>
<td>Output</td>
<td>1</td>
</tr>
<tr>
<td>(5.9)</td>
<td>normalized step size</td>
<td>TDNN</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>FIR output</td>
<td>FIR</td>
<td>$n_2$</td>
</tr>
<tr>
<td>(5.9)</td>
<td>normalized step size</td>
<td>FIR</td>
<td>2</td>
</tr>
<tr>
<td>(5.10)</td>
<td>weight update</td>
<td>FIR</td>
<td>$n_2+2$</td>
</tr>
<tr>
<td>(5.11)</td>
<td>bias update</td>
<td>FIR</td>
<td>1</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td></td>
<td></td>
<td>$2^*(n_1+n_2)+22$</td>
</tr>
</tbody>
</table>

Note: $n_1$ equals the length of the TDNN portion TDL and $n_2$ equals the length of the FIR portion TDL.

By comparison, the NLMS algorithm requires $2M+1$ operations, where $M$ is the order of the filter.

5.4.2 Experimental Results

**Measurement Setup.** Measurements are performed in Conference Room #2 using HFT #6 which is placed on top of the conference table. The reference source signal consists of white noise which is bandlimited from 300 Hz to 3400 Hz. The filtered reference signal is then amplified such that the loudspeaker produces a sound pressure level (SPL) from 60dB to 95dB as measured 0.5m directly above the loudspeaker. Primary and reference DAT signals are subsequently downloaded to a computer via an ARIEL DSP96 board sampling at 16 kHz. These samples are then applied to both the proposed structure and a 600 tap linear adaptive FIR filter which has DC bias compensation and weights updated in the same fashion as equations (4.4) through (4.7). In the proposed structure, the number of taps in the nonlinear section delay line equals 200 to cover the bulk of the loudspeaker impulse response. The number of taps in the linear section is 400 for a total impulse
length of 600 taps which is sufficient to cover the bulk of the impulse response for Conference Room #2. For each SPL, both algorithms are tested with the same input data of length 80,000 to allow convergence to a steady state at which point the average ERLE is measured and plotted.

Results. In Figure 5.16 (a), experimental results are shown for the case where keys have been taped down to prevent rattling at high volumes. Over 11 dB of improvement can be seen at 95 dB SPL compared to the linear algorithm, and between 0-2 dB improvement is obtained over the rest of the volume range. Figure 5.16 (b) shows the corresponding power spectral density of the primary and reference signals, as well as the error signals for the linear and nonlinear algorithms.

Figure 5.16 (c) shows experimental results for the case where keys are not taped down. Finally, the convergence curves for case (c) are shown in Figure 5.16 (d). At low volumes in the vicinity of 60 dB SPL, the proposed structure improves the ERLE by 3 dB as compared to the NLMS-FIR filter even though there is little nonlinear distortion in this range. In the low volume ranges, room noise and two-point suspension nonlinearities are the dominant limitations and the proposed structure offers some improvement. In the medium volume range from 70-75 dB SPL, the proposed structure performs about 1 dB poorer than the linear filter due to an extra bias weight variance not included in the linear filter, and also because the sigmoid function will generate some small amount of distortion for any \(|s| > p\) even when the inputs are linear. However, in the vicinity of 80 to 95 dB SPL where nonlinear effects dominate, the proposed structure clearly outperforms the linear filter in terms of converged ERLE and demonstrates over 8 dB improvement at 90 dB SPL.

5.4.3 Discussion

A new structure to mitigate nonlinear loudspeaker distortion effects in AEC’s has been presented in this subsection. The architecture is simple and the update algorithms are based on stochastic
FIGURE 5.16 Experimental results showing performance of the proposed structure using HFT #6 in conference room #2. (a) Converged ERLE, keys taped down. (b) plot of PSD of signals for the taped keys case. (c) Converged ERLE, keys not taped. (d) convergence curve, keys not taped down.

gradient methods. Experimental measurements in a conference room indicate that this new structure is capable of improving the ERLE by over 8 dB at high volumes where nonlinear effects are significant and by over 3 dB at low volumes where room noise is significant. Most striking is the difference in performance between data sets that have been collected with and without the keys taped down (i.e. to prevent rattling). By taping the keys down, the proposed structure will achieve 1-2 dB improvement in converged ERLE in the low-medium volume ranges and a 4 dB improve-
ment in converged ERLE at 95 dB SPL compared to the case where the key movement is not restricted.

For the experimental results presented, the complexity of the two stage neural filter is only 2% greater than the NLM-FIR filter. This is a marginal increase in complexity for the improvements obtained.

Plots of the power spectral density of the signals (See Figure 5.16 (b)) also show that the error signal is reduced in amplitude evenly across the full bandwidth, as compared to the error obtained with the FIR/NLMS algorithm which closely follows the out-of-band primary signal PSD. For the proposed structure, the error spectral density near the nyquist sampling frequency does not follow the primary signal amplitude and this is due to the distortion “regeneration” phenomenon associated with passing a signal through a nonlinear sigmoid.

5.5 Variable Activation Function

In this section, the mixed linear-sigmoid activation function described previously in Section 5.2.2 is modified so that it is fully adaptive. The motivation for pursuing this idea is based on both simulation results (see Figure 5.10) and experimental results (see Section 5.2.3) which show that improvements can be made by changing the size of the linear region in the activation function.

The idea of using an adaptive activation function in the realm of adaptive filtering has been previously proposed by Zhan and Li [149] as a method for realizing arbitrary nonlinear filters. However, the adaptive neural filter algorithm that [149] proposes is limited in application since the activation function is placed at the output only and does not allow for placement in a hidden layer. As well, the method proposed in [149] requires that training of the activation function be performed with all other weights being held constant. In this section, these restrictions are removed by
developing a training algorithm that allows the activation function to assume a fully adaptive role regardless of its placement within the network, and without limitations on the training of the network weights.

### 5.5.1 Development of the Learning Algorithm

We define the sigmoid function as given previously in equation (5.1), repeated here for convenience.

$$\varphi(s, p) = \begin{cases} s & |s| \leq p \\ \text{sign}(s) \left[ (1 - p) \cdot \tanh \left( \frac{|s| - p}{1 - p} \right) + p \right] & |s| > p \end{cases} \quad (5.12)$$

where $s$ is the input and $p$ defines the linear region. The instantaneous cost function $J_{\text{inst}}$ at time $n$ is defined as:

$$J(n) = J_{\text{inst}}(n) = \frac{1}{2} \sum_{i=1}^{N_L} e_i^2(n) \quad (5.13)$$

where:

$$e_i(n) = d_i(n) - y_i(n) \quad (5.14)$$

The algorithm attempts to minimize the $J$ by the delta rule [8] for the vectors $\mathbf{w}$ and $\mathbf{p}$ by incrementing at each step towards the optimum vector using the negative gradient at that point. The weight update equations for the weight vector $\mathbf{w}$ are shown in Section 3.7.1. For $\mathbf{p}$ we have:

$$\mathbf{p}(n+1) = \mathbf{p}(n) - \eta \frac{\partial J}{\partial \mathbf{p}} \quad (5.15)$$

$\mathbf{p}$ is a vector consisting of all the $p$ parameters for the sigmoids.
\[ p(n) = [p_1^{(l)}(n), p_2^{(l)}(n), \ldots, p_{n_1}^{(l)}(n), \ldots, p_{n_L}^{(l)}(n)] \]  

(5.16)

and \( \eta \) is a fixed step size. The output of node \( j \) in layer \( l \) is:

\[ x_j^{(l)}(n) = \varphi(s_j^{(l)}(n), p_j^{(l)}(n)) \]  

(5.17)

where \( \varphi \) represents the nonlinear activation function and \( s_j^{(l)}(n) \) is the node activation input.

The derivation of the correction applied to vector \( p \) may be done by expanding the gradient as follows using the chain rule:

\[ \frac{\partial J(n)}{\partial p_j(n)} = \frac{\partial J(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial x_j(n)} \frac{\partial x_j(n)}{\partial p_j(n)} \]  

(5.18)

Differentiating (5.13) with respect to \( e_j(n) \), we get

\[ \frac{\partial J(n)}{\partial e_j(n)} = e_j(n) \]  

(5.19)

Differentiating (5.14) with respect to \( x_j(n) \), we get

\[ \frac{\partial e_j(n)}{\partial x_j(n)} = -1 \]  

(5.20)

Differentiating (5.17) with respect to \( p_j(n) \), we get

\[ \frac{\partial x_j(n)}{\partial p_j(n)} = \varphi'_p(s_j(n), p_j(n)) \]  

(5.21)

where \( \varphi'_p(s_j^{(l)}(n), p_j(n)) \) signifies derivative of the activation function output with respect to the argument \( p_j(n) \). Differentiating (5.12) with respect to \( p \), and dropping reference to \( n \), we get:
\[ \varphi'_p(s, p) = \begin{cases} 
0 & ;|s| \leq p \\
\text{sign}(s) \left\{ -\tanh(\theta) + (1 - p) \left[ 1 - \tanh(\theta) \right]^2 \left[ \frac{\theta - 1}{1 - p} \right] + 1 \right\} & ;|s| > p
\end{cases} \]

where \( \theta = \left( \frac{|s| - p}{1 - p} \right) \) \hspace{1cm} (5.22)

Figure 5.17 shows the activation function of equation (5.12) with values of \( p \) equal to 0, 0.5 and 0.9, along with the associated \( \varphi'_s(s, p) \) and \( \varphi'_p(s, p) \) values.

![Graph showing activation function and derivative with respect to \( p \) as the input \( s \) varies for values of \( p=0.0, 0.5 \) and 0.9.](image)

**FIGURE 5.17** Adaptive activation function and derivative with respect to \( p \) as the input \( s \) varies for values of \( p=0.0, 0.5 \) and 0.9.

The correction \( \Delta p_j(n) \) applied to \( p_j(n) \) can now be redefined using the delta rule as:

\[ \Delta p_j(n) = \eta \xi_j(n) \] \hspace{1cm} (5.23)

where the local gradient \( \xi_j(n) \) for the parameter \( p \) is defined by

\[ \xi_j(n) = \frac{\partial J(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial x_j(n)} \frac{\partial x_j(n)}{\partial p_j(n)} = -e_j(n) \varphi'_p(s_j(n), p_j(n)) \] \hspace{1cm} (5.24)
Equation (5.24) is derived based on the adaptive activation function being in the output layer. If the neuron is located in a hidden layer of the network, we must redefine the local gradient for neuron \( j \) in the same way as the BP algorithm as follows:

\[
\xi_j(n) = -\frac{\partial J(n)}{\partial x_j(n)} \frac{\partial x_j(n)}{\partial p_j(n)} = -\frac{\partial J(n)}{\partial x_j(n)} \varphi'_p(s_j(n), p_j(n)) \tag{5.25}
\]

where

\[
\frac{\partial J(n)}{\partial x_j(n)} = -\sum_{k=1}^{N_i} e_k(n) \varphi'_k(s_k(n)) w_{jk}(n) = -\sum_{k=1}^{N_i} \delta_k(n) w_{jk}(n) \tag{5.26}
\]

Using (5.26) in (5.25), we obtain the local gradient \( \xi_j(n) \) for a hidden neuron \( j \) in layer \( l \),

\[
\xi_{j}^{(l)}(n) = \varphi'(p_{j}^{(l)}(n)) \sum_{k=1}^{N_i} \delta_k^{(l+1)}(n) w_{jk}^{(l)}(n) \tag{5.27}
\]

Equation (5.27) shows that the local gradient \( \xi_j(n) \) is dependent on both the derivative (with respect to \( p \)) of the associated activation function as well as the weighted sum of the \( \delta \)'s computed for the neurons in the next hidden or output layer which is connected to neuron \( j \).

It should be noted that there is no restriction on the type of nonlinearity used. For example, we may define an alternate hyperbolic tangent function such as:

\[
\varphi(s, p) = \frac{1}{p} \tanh(sp) \\
\varphi'_s(s, p) = 1 - (\tanh(sp))^2 \\
\varphi'_p(s, p) = \frac{s}{p^2} [1 - (\tanh(sp))^2] - \frac{1}{p^3}\tanh(sp) \tag{5.28}
\]
Yamada et al. [150] have used a similar unit function to construct a direct neural network controller for robot manipulators, but with a different definition from that described above and without giving the learning algorithm of the parameter \( p \). A plot of the function and slope of (5.28) with respect to the activation input \( s \) and the variable \( p \) is shown in Figure 5.18 for values of \( p = 0.5 \) and 2. It differs somewhat from that of (5.12) in that the function becomes linear as \( p \) approaches 0 and highly nonlinear as \( p \) approaches infinity.

![Figure 5.18](image)

**Figure 5.18** Alternate activation function and derivative with respect to \( s \) as the input \( s \) varies (b) derivative with respect to \( p \) as the input \( s \) varies.

The advantage of the sigmoid of equation (5.28) is that a two layer TDNN structure with fixed activation functions and a single weight on the output is similar to a single layer TDNN with a
variable activation function. The complete algorithm is summarized below:

---

**BP Algorithm with Variable Activation (VA) Functions:**

**Step 1: Initialization.** Set all synaptic weights in input layer to zeros, and all other weights and threshold levels to small random numbers that are uniformly distributed.

**Step 2: Forward Computation.** For all training samples, compute the activation potentials and outputs of the networks forward layer by layer using the following equations:

\[
s_j^{(l+1)}(n) = \sum_{i=0}^{N_l} w_{ij}^{(l)}(n) x_i^{(l)}(n)
\]  

(5.29)

where \(x_i^{(l)}(n)\) represents both the output from the previous layer and the input to weight matrix elements \(w_{ij}^{(l)}(n)\) at time \(n\). The output of node \(j\) in layer \(l\) is:

\[
x_j^{(l)}(n) = \varphi(s_j^{(l)}(n), p_j^{(l)}(n))
\]  

(5.30)

Compute the error signal \(e(n)\) produced at the output layer (i.e. \(l=L\)) of the network:

\[
e(n) = d(n) - x_i^{(L)}(n)
\]  

(5.31)

**Step 3: Backpropagation of Errors.** Compute the local gradients \(\delta\)'s and \(\xi\)'s of the network by proceeding backward, layer by layer:
\[ a_j^{(l)}(n) = \frac{\partial J(n)}{\partial x_i^{(L)}(n)} = \begin{cases} -e_i(n) & \text{for } l = L \\ \sum_{j=1}^{N_{i+1}} \delta_j^{(l+1)}(n) w_{ij}^{(l)}(n) & \text{for } 1 \leq l \leq L - 1 \end{cases} \quad (5.32) \]

\[ \delta_j^{(l)}(n) = a_j^{(l)}(n) [\varphi'(s_j^{(l)}(n), p_j^{(l)}(n))] \quad (5.33) \]

\[ \xi_j^{(l)}(n) = a_j^{(l)}(n) [\varphi'(s_j^{(l)}(n), p_j^{(l)}(n))] \quad (5.34) \]

**Step 4: Update Parameters.**

\[ w_{ij}^{(l)}(n+1) = w_{ij}^{(l)}(n) - \mu \delta_j^{(l+1)}(n) \cdot x_i^{(l)}(n) \quad (5.35) \]

\[ p_j^{(l)}(n+1) = p_j^{(l)}(n) - \eta \xi_j^{(l+1)}(n) \quad (5.36) \]

The value of \( p_j^{(l)}(n+1) \) should be clamped above 0 for the sigmoid of equation (5.28) and between 0 and 1 for the sigmoid function of equation (5.12).

### 5.5.2 Simulation Examples

In this section, the proposed algorithm is applied to the identification of a nonlinear system constructed using simulation method #1. The model is a (10,5,1) two layer network with a variable activation function in the hidden layer, and a linear activation function in the output layer. A normalized step size of 0.1 is used for updating the layer weights and the step size for adapting the activation function is set to 0.001. Also shown for comparison is the NLMS-FIR filter using \( \alpha=0.1 \).
5.5 Variable Activation Function

![Graphs showing simulation results for distortion method #1, (10,5,1) variable activation TDNN, filtered reference signal. (a) converged ERLE vs. SDR (b) convergence curve for the highest distortion level.](image)

FIGURE 5.19 Simulation results for distortion method #1, (10,5,1) variable activation TDNN, filtered reference signal. (a) converged ERLE vs. SDR (b) convergence curve for the highest distortion level.

Simulation results in Figure 5.19 show that the variable activation TDNN is capable of achieving a higher ERLE over the NLMS-FIR filter from 0-45 dB SDR and also achieves better steady state ERLE compared to the conventional (i.e. fixed activation function) model of Section 5.2.

5.5.3 Experimental Results

**HFT #6 in Conference Room #2.** In this experiment, a single hidden layer TDNN with variable activation (VA) function is applied to the measured data. A normalized step size of $\alpha=0.5$ is used for the weight updates, and a fixed step size of 0.001 is used for the update of the $p$ parameter.

The results shown in Figure 5.20 (a) indicate that the single layer VA TDNN filter has similar performance to the regular TDNN structure results shown in Figure 5.13. However, the VA TDNN filter steady state ERLE performance is between 1 and 5 dB better than the fixed activation function TDNN (See Figure 5.13) in the 57-95 dB SPL ranges. In the 50-55 dB range the performance is still poorer than the FIR structure. Figure 5.20 (b) shows the corresponding convergence for the 95 dB SPL case.
5.5.4 Discussion

Both simulation and experimental results show that the VA-TDNN is capable of achieving lower modelling error for the identification of a nonlinear system that has a widely varying range of nonlinearity (for example a loudspeaker going into and out of the saturation range) compared to a network using a conventional fixed sigmoid activation function. However, the convergence rate is slower than the fixed activation TDNN especially for higher distortion values near 95 dB SPL. In terms of application to AEC, the slowing of convergence is an undesirable feature, however, methods exist to improve the convergence performance of neural networks, and this is discussed further in Chapter 6.

The results shown in Figure 5.20 (c) indicate that a single hidden layer TDNN can still achieve significant modelling accuracy improvements as compared to the FIR structure, although they are almost identical in architecture, save for the activation function. The important point to stress is that this architecture has a complexity only slightly more than the NLMS algorithm since back-
propagation through several layers is no longer necessary and suggests that for the nonlinear AEC problem, it is possible to obtain significant improvements in converged ERLE using simple nonlinear mechanisms.

Similar to the conventional TDNN case, there is a significant improvement in converged ERLE in the 60-75 dB SPL volume range. It suggests even in the so-called “linear” range of the loudspeaker, there is small level of nonlinear distortion in the LREM that can be compensated, resulting in improved performance.

The VA structures presented in this section still do not model the low volume ranges very well. The performance, for example at 55 dB SPL is still approximately 9 dB worse than the linear FIR structure. Various parameter changes were made to see if this would make a difference, however, no conclusive results were obtained to demonstrate a clear reason why this phenomenon occurs.

5.6 MLP with FIR Synapses and Variable Activation Function

Recent interest in deriving temporal neural network structures for modelling time-dependent signals has resulted in MLP structures with synapses described by FIR filters [50][47] and IIR filters [42][43]. The architecture can be considered an extension of the classical feedforward structure described previously in Section 3.7. In this section we examine the FIR MLP, which can be considered an ideal architecture for modelling cascaded linear-nonlinear-linear temporal structures of the type illustrated in Figure 5.21. The FIR MLP is generalized to include the variable activation (VA) functions developed in Section 5.5 and is then applied to simulated and experimental data.

The motivation for pursuing this line of attack is based on the encouraging results in [151] which document a number of techniques for application in Time Series Prediction. The two most successful techniques outlined in [151] are based on function approximation using FIR and IIR filters
FIGURE 5.21 Using the FIR MLP to represent a cascaded linear-nonlinear-linear subsystem (i.e. nonlinear HFT).

between synapses of a neural network and a state space modelling to develop a representation of the system without knowing the system equations.

State space modelling has already found success in nonlinear system identification and filtering (see for example [80] [82]), and encouraging results are presented for the application of FIR MLP architectures to time series prediction [50][47][152], however, no application of the FIR MLP method has been found in the literature in the realm of real-time nonlinear adaptive filtering. Consequently, the application of VA FIR MLPs to nonlinear AEC is examined here.

5.6.1 Network Architecture

The FIR MLP architecture is shown in Figure 5.22. Define the internal states of the network:

\[ x_j^{(l)}(n), \quad 1 \leq j \leq N_l \]  \hspace{1cm} (5.37)

where

- \( x_j^{(l)}(n) \) represent the input to the \( j \)-th FIR synaptic filter in the \( l \)-th input layer
- \( N_l \) represents the number of nodes in layer \( l \).
For the input layer, \( l = 0 \), and for the output layer, \( l = L \). The output of the \( i^{th} \) FIR synapse is:

\[
z_{ij}^{(l+1)}(n) = [w_{ij}^{(l)}(n) \cdot x_i^{(l)}(n)]
\] (5.38)

where \( \cdot \) represents the dot product of vectors

\[
w_{ij}^{(l)} = [w_{ij}^{(l)}(0), w_{ij}^{(l)}(1), \ldots, w_{ij}^{(l)}(T^{(l)})]^T
\] (5.39)

represents the FIR weight vector connecting the output of neuron \( i \) in the \( l^{th} \) layer to the input of neuron \( j \) in the \( l+1^{th} \) layer and

\[
x_i^{(l)} = [x_i^{(l)}(n), x_i^{(l)}(n-1), \ldots, x_i^{(l)}(n-T^{(l)})]^T
\] (5.40)

represents the vector formed from the output \( x_i^{(l)}(n) \) from the previous layer and the TDL of the FIR connecting node \( i \) in the \( l^{th} \) layer to node \( j \) in the \( l+1^{th} \) layer. The activation level at the input to the sigmoid nonlinearity of neuron \( j \) in layer \( l \) is

\[
s_j^{(l)}(n) = \sum_{i=0}^{N_i} z_{ij}^{(l)}(n) - \theta_j^{(l)}(n)
\] (5.41)

where \( \theta_j^{(l)}(n) \) is the bias added to the net input. The output at the \( l^{th} \) layer is obtained by putting \( s_j^{(l)}(n) \) through the activation function.

\[
x_j^{(l)}(n) = \varphi(s_j^{(l)}(n), p_j^{(l)}(n))
\] (5.42)

### 5.6.2 Derivation of the Modified Temporal BP Learning Algorithm

An algorithm for training networks having FIR synapses was first published by Wan [47] which is based on the total squared error over the entire sequence of inputs as opposed to the "instantaneous error". The calculations for the deltas \( \delta_j^{(l)}(n) \) in [47] are in fact non-causal. Since it takes time
5.6 MLP with FIR Synapses and Variable Activation Function

FIGURE 5.22 Forward signal propagation in the FIR MLP.

for the output of any internal neuron to completely propagate through the network, the change in the total error due to a change in an internal state is a function of future values within the network. The solution is to add a finite number of delay operators into the network states $x_j^{(l)}(n)$ and propagate the deltas backwards without delay. The result is that the internal weights are updated at time $(n)$ based on the deltas and internal states at time $(n-D)$ where $D$ is some fixed delay. This is
analogous to the delayed LMS algorithm (Kabal [153]) which exhibits a slower convergence (but similar misadjustment) as compared to the standard LMS algorithm.

In [154], several training algorithms are presented for the FIR MLP based on whether the performance criteria is obtained using an instantaneous error, or the total error by summing the instantaneous error over all time steps in a training sequence. The authors conclude that algorithms based on minimization of the total error are very inefficient for networks of more than two layers. Wan’s original temporal algorithm in [47] uses a total cost function and has an update delay to maintain causality. Since the system we are trying to identify has a large order for FIR$_2$, which is representative of a typical room response (See Figure 5.21), the update algorithms based on the total cost function are not investigated since this would involve a large delay in the tap weight update, and consequently slower convergence.

The parameter update equations are derived in a way similar to Wan’s method [47], but with modifications as follows:

- We use an instantaneous cost function (rather than total cost function over all time).
- The adaptive activation function derived in Section 5.5 is used.

We first consider the weight updates, and then the updates of the variable sigmoid parameter $p$.

**Weight Update Derivation.** The most straightforward way of updating the weight vectors is to minimize the instantaneous cost function $J$ (See equation (5.13)) using the stochastic gradient descent algorithm at each increment of time $n$ according to [47]:

```plaintext
```
\[
\begin{align*}
\mathbf{w}^{(l)}_{ij}(n+1) & = \mathbf{w}^{(l)}_{ij}(n) - \mu \frac{\partial J}{\partial \mathbf{w}^{(l)}_{ij}(n)} \\
& = \mathbf{w}^{(l)}_{ij}(n) - \mu \frac{\partial J}{\partial s^{(l+1)}_j(n)} \frac{\partial s^{(l+1)}_j(n)}{\partial \mathbf{w}^{(l)}_{ij}(n)} 
\end{align*}
\] (5.43)

The partial derivative of \( s^{(l+1)}_j(n) \) with respect to the weight vector \( \mathbf{w}^{(l)}_{ij}(n) \) is given by

\[
\frac{\partial s^{(l+1)}_j(n)}{\partial \mathbf{w}^{(l)}_{ij}(n)} = \mathbf{x}^{(l)}_i(n) \] (5.44)

The local gradient for neuron \( j \) in layer \( l \) is defined as

\[
\delta^{(l)}_j(n) = \frac{\partial J(n)}{\partial s^{(l)}_j(n)} \] (5.45)

Hence (5.43) may be written in the familiar form

\[
\mathbf{w}^{(l)}_{ij}(n+1) = \mathbf{w}^{(l)}_{ij}(n) - \mu \delta^{(l+1)}_j(n) \mathbf{x}^{(l)}_i(n) \] (5.46)

**Case 1: Output Layer Weights.**

\[
\delta^{(L)}_j(n) = \frac{\partial J(n)}{\partial s^{(L)}_j(n)} = \frac{1}{2} \frac{\partial e^2_j(n)}{\partial s^{(L)}_j(n)} = -e_j(n) \varphi'(s^{(L)}_j(n), p^{(L)}_j(n)) \] (5.47)

**Case 2: Hidden Layer Weights.** For a hidden layer, we use the chain rule, expanding over all \( N_{l+1} \) inputs \( s^{(l+1)}(n) \) in the next layer. However, instead of expanding over all time as is done in [47], the expansion is only done at time \( k=n \) since we are only concerned with the instantaneous error.
\[ \delta_j^{(l)}(n) = \frac{\partial e^2(n)}{\partial s_j^{(l)}(n)} = \sum_{k=1}^{N_{l+1}} \sum_{l=1}^n \frac{\partial f(n)}{\partial s_j^{(l+1)}(t)} \frac{\partial s_k^{(l+1)}(t)}{\partial s_j^{(l)}(n)} \]

\[ = \sum_{k=1}^{N_{l+1}} \frac{\partial J(n)}{\partial s_j^{(l+1)}(n)} \frac{\partial s_k^{(l+1)}(n)}{\partial s_j^{(l)}(n)} = \sum_{k=1}^{N_{l+1}} \delta_j^{(l+1)} \frac{\partial s_k^{(l+1)}(n)}{\partial s_j^{(l)}(n)} \]  

\[ = \varphi'_s(s_j^{(l)}(n), p_j^{(l)}(n)) \sum_{k=1}^{N_{l+1}} \delta_j^{(l+1)}(n) \frac{\partial z_{jk}^{(l+1)}(n)}{\partial x_j^{(l)}(n)} \]  

Recall that

\[ z_{jk}^{(l+1)}(n) = \left[ w_{jk}^{(l)}(n) * x_j^{(l)}(n) \right] = \sum_{t=0}^{T(n)} w_{jk}^{(l)}(t) x_j^{(l)}(n-t) \]  

where \( T(n) \) is the number of delays in the FIR sections in layer \( l \). Thus for the instantaneous case we obtain

\[ \frac{\partial z_{jk}^{(l+1)}(n)}{\partial x_j^{(l)}(n)} = w_{jk}^{(l)}(n) \]  

\[ \text{Algorithm 1: Instantaneous Gradient.} \] Substitution of (5.50) into (5.48), we get the delta update for the hidden layer,

\[ \delta_j^{(l)}(n) = \varphi'_s(s_j^{(l)}(n), p_j^{(l)}(n)) \sum_{k=1}^{N_{l+1}} \delta_j^{(l+1)}(n) w_{jk}^{(l)}(n) \]  

This can be considered an \textit{approximate} instantaneous gradient. The \( \delta \) terms are calculated using standard backpropagation through the first weight of each FIR synapse, and the rest of the coefficients are ignored.
Algorithm 2: Accumulated Gradient. A different form is achieved if the gradient is calculated over a short time window $1 \leq n_w \leq T^{(l)}$ by delaying the calculation of the gradient until all contributions from feedforward delay elements can be combined.

$$
\delta_j^{(l)}(n) = \varphi_s'(z_j^{(l)}(n)) \sum_{k=1}^{N_{l+1}} v_{jk}^{(l)}(n) \tag{5.52}
$$

$$
v_{jk}^{(l)}(n) = [\Delta_k^{(l+1)}(n) \ast \bar{w}_{jk}^{(l)}(n)] \tag{5.53}
$$

where the quantities $\Delta_k^{(l+1)}(n)$ and $\bar{w}_{jk}^{(l)}(n)$ define the length of the accumulated gradients, defined by

$$
\Delta_k^{(l+1)}(n) = [\delta_k^{(l)}(n), \delta_k^{(l)}(n-1), ..., \delta_k^{(l)}(n-n_w)]^T \tag{5.54}
$$

$$
\bar{w}_{jk}^{(l)}(n) = [w_{jk}^{(l)}(0), w_{jk}^{(l)}(1), ..., w_{jk}^{(l)}(n_w)]^T \tag{5.55}
$$

In this case, the backpropagated error is obtained from a backward filter, and all the coefficients up to the $n_w$th coefficient will have an influence on the $\delta$ value. Figure 5.23 illustrates this process.

Algorithm 1 was first proposed by Back and Tsoi [50]. The tapped delay line in the FIR synapse allows a number of options in calculating the gradient. However, the gradient can be obtained from an instantaneous estimate, i.e. using only the first weight in the backward filter, or from $T^{(l)}$ delay sections, i.e. using the entire FIR synapse. It may be observed that Algorithm 1 is just a special case of Algorithm 2, since it can be obtained by fixing the gradient window parameter $n_w=1$. However, rather than fixing $n_w=1$ as is done in [50], the value of $n_w$ in the above derivation is allowed to assume any value between 1 and $T^{(l)}$ and thus offers an additional degree of flexibility not offered in the algorithm presented in [154].
FIGURE 5.23 Backward filter propagation of "accumulated" gradient terms.
Modification for the Variable Activation Function. The inclusion of a variable activation function can be done in the same manner as outlined in Section 5.5. The adaptation of $p$ is done according to the stochastic gradient update:

$$p(n+1) = p(n) - \eta \frac{\partial J}{\partial p}$$  \hspace{1cm} (5.56)

$$\frac{\partial J}{\partial p_j(n)} = z_j^{(l)}(n) = \frac{\partial \varphi}{\partial p_j(n)}\sum_{k=1}^{N_l} \left[ \Delta_k^{(l+1)}(n) \ast \bar{w}_j^{(l)}(n) \right]$$

$$= \varphi'(s_j^{(l)}(n), p_j^{(l)}(n)) \sum_{k=1}^{N_l} \left[ \Delta_k^{(l+1)}(n) \ast \bar{w}_j^{(l)}(n) \right]$$  \hspace{1cm} (5.57)

Essentially, the derivative of the activation function is computed with respect to $p$ and then it is multiplied by the filtered delta vector, which is the quantity in brackets. Equation (5.57) has the same format as (5.27) with the exception deltas and weights are now vector operator due to the FIR synapses.

Summarizing, the complete adaptation algorithm for the parameter updates can be expressed as follows:

---

**FIR MLP Algorithm with Variable Activation Functions:**

**Step 1: Initialization.** Set all synaptic weights in the input layer to zeros, and all other weights and threshold levels to small random numbers that are uniformly distributed.

**Step 2: Forward Computation.** For all training samples, compute the activation potentials and outputs of the networks forward layer by layer using the following equations:
\[ s_{j}^{(l+1)}(n) = \sum_{i=0}^{N_l} [w_{ij}^{(l)}(n) \cdot x_{i}^{(l)}(n)] - \theta_{j}^{(l)}(n) \] 

(5.58)

Compute the output of node \( j \) in layer \( l \) using:

\[ x_{j}^{(l)}(n) = \varphi(s_{j}^{(l)}(n), p_{j}^{(l)}(n)) \]

(5.59)

Compute the error signal \( e(n) \) produced at the output layer (i.e. \( l=L \)) of the network:

\[ e(n) = d(n) - x_{1}^{(L)}(n) \]

(5.60)

**Step 3: Backpropagation of Errors.** Compute the local gradients \( \delta \)'s and \( \xi \)'s of the network by proceeding backward, layer by layer:

\[ a_{j}^{(l)}(n) = \begin{cases} 
-e_{i}(n) & \ldots l = L \\
\sum_{j=1}^{N_{l}+1} \left[ \Delta_{j}^{(l+1)}(n) \cdot \bar{w}_{ij}^{(l)}(n) \right] & \ldots 1 \leq l \leq L-1
\end{cases} \]

(5.61)

\[ \delta_{j}^{(l)}(n) = a_{j}^{(l)}(n) [\varphi_{x}'(s_{j}^{(l)}(n), p_{j}^{(l)}(n))] \]

(5.62)

\[ \xi_{j}^{(l)}(n) = a_{j}^{(l)}(n) [\varphi_{p}'(s_{j}^{(l)}(n), p_{j}^{(l)}(n))] \]

(5.63)

where \( \Delta_{j}^{(l+1)}(n) \) and \( \bar{w}_{ij}^{(l)}(n) \) are vectors of length \( 1 \leq n_{w} \leq T^{(l)} \).

**Step 4: Update Parameters.**

\[ w_{ij}^{(l)}(n+1) = w_{ij}^{(l)}(n) - \mu \delta_{j}^{(l+1)}(n) x_{i}^{(l)}(n) \]

(5.64)
\[ p^{(l)}_j (n+1) = p^{(l)}_j (n) - \eta \xi^{(l+1)}_j (n) \] (5.65)

The value of \( p^{(l)}_j (n+1) \) should be clamped above 0 for the sigmoid of equation (5.28) and between 0 and 1 for the sigmoid function of equation (5.12). If the values of \( p^{(l)}_j \) are fixed (i.e. not updated) and \( n_w \) is set equal to \( T^{(l)} \) for all sections, the above algorithm defaults to the instantaneous cost accumulated gradient algorithm proposed in [154].

### 5.6.3 Simulation Results

In this section we apply the proposed structure to the identification of a nonlinear system as shown in Figure 5.24 with the following parameters: Number of taps in first FIR section = 5, the activation function used was defined by equation (5.12) with parameter \( p = 0.5 \), and the number of taps in the 2nd FIR section is 10. Both of the FIR sections have the weights and biases randomly assigned.

*FIGURE 5.24 System identification using the proposed model.*
Three structures were tested. The first structure (called ‘FIR’) is a conventional FIR structure consisting of 15 taps with the inclusion of a bias weight to compensate for output bias. Fifteen taps were chosen to accommodate the impulse length of the “unknown” system. The normalized step size alpha was set to 0.1 and it was trained with the NLMS algorithm.

The second structure tested (called ‘IC’) consists of two FIR sections with a fixed sigmoidal activation function between them, essentially equation (5.12) with $p=0$. The number of taps in the FIR sections is set to 5 and 10 respectively, and the gradient accumulation $n_w$ is set to 1. This is equivalent to IC-2 in [154]. The normalized step size alpha was set to 0.1.

The third structure is the proposed architecture with variable activation function and gradient accumulation. It has a similar architecture to the second structure except we allow the activation function to adapt parameters $p$ according to the proposed training algorithm, and set the gradient accumulation window $n_w$ to 1 or 3. The normalized step size alpha was set to 0.1. This algorithm is called ‘ICVA’.

The training sequence consists of 8000 randomly generated data points. For all the algorithms, the normalized mean square error (NMSE) is plotted according to the formula:

$$NMSE(n) = 10 \log \left( \sum_{r=0}^{500} \left[ e_r(k) \right]^2 \right) / \sum_{r=0}^{500} \left[ d_r(k) \right]^2 dB$$  \hspace{1cm} (5.66)

where $e_r(k)$ and $d_r(k)$ represent the averaged error and desired signals and $r$ represents the window values over which these averages are then smoothed, in this case equal to 500. The convergence results are shown in Figure 5.25. The FIR structure trained with the NLMS algorithm is
FIGURE 5.25 Comparison of convergence using the proposed algorithms.

clearly unable to identify the unknown system accurately, and obtains an average NMSE of only -- 11 dB. Both the IC structure and ICVA structure with $n_w=1$ perform considerably better than the FIR structure. The ICVA structure is able to achieve a slightly faster convergence in the 0-2000 iteration range, and both converge to approximately -24 dB NMSE. By increasing the gradient accumulation window, an increase in convergence speed is seen to occur. With $n_w=1$, ICVA achieves -20 dB NMSE after 1500 iterations. With $n_w=3$, the same NMSE is achieved after only 1000 iterations.

5.6.4 Experimental Results

*HFT #6 in Conference Room #2.* The ICVA architecture is applied to experimental data collected in this venue. Two different architectures are evaluated. The first architecture is similar to that shown in Figure 5.24 and has 150 taps in the first FIR section, followed by 850 taps in the second FIR section. The gradient accumulation window is equal to 100, the normalized step size = 0.5 and
the adaptive parameter step size is 0.01. The sigmoid of equation (5.28) is used. This architecture is called "150/850 ICVA". The second architecture is simplified by using a single variable activation function in front of a 1000 tap FIR structure, i.e. the first FIR section is absent. This architecture is called "0/1000 ICVA" and has the same parameters as the first architecture. Note that this architecture is almost identical to a conventional FIR structure, however, it is still necessary to use the temporal backpropagation algorithm due to the TDL in the FIR portion.

The results shown in Figure 5.26 indicate that the 150/850 ICVA obtains a lower converged ERLE with respect to the conventional two layer TDNN (see for comparison results in Figure 5.13) but obtains a converged ERLE approximately 1 dB better than the conventional TDNN at 95 dB SPL. The 0/1000 ICVA structure obtains 2 to 3 dB better performance than the FIR model at SPL levels between 60 and 70 dB, but is worse at other volumes.

![Graph](image)

**FIGURE 5.26** Experimental results. HFT #6 in conference room 2 showing a comparison of FIR, VA-FIR MLP using 150/800 taps in first/second FIR sections, and simplified VA-FIR MLP with only one FIR section following adaptive sigmoid.
5.6.5 Discussion

Simulation results presented in Section 5.6.3 illustrate that improved convergence can be obtained by utilizing an adaptive activation function with gradient accumulation window. The proposed structure can be considered as a new alternative architecture to conventional MLPs which utilize fixed sigmoidal activation functions only. By selecting the size of the gradient accumulation window, a trade-off between convergence performance and complexity can be achieved, and that for low order systems, the gain is impressive. Plots of the converged MSE also show that the excess MSE is higher when gradient accumulation is used, compared to when the accumulation is not used, i.e. $n_w=1$. The experimental results presented in Section 5.6.4 indicate that the proposed VA-FIR-MLP structure is not as good as the conventional TDNN utilizing an *adaptive* activation function (See Section 5.5.3). As well, the performance at low volumes does not equal that of the linear FIR, although at medium and high volumes it is 1-2 dB better.

5.7 Summary

This chapter has addressed methods to combat nonlinear loudspeaker distortion in AEC's, by the application of some known and newly proposed nonlinear structures. In Section 5.1 results using a 3rd order Volterra structure are presented. The following conclusions can be summarized:

- Although the Volterra filter can obtain a high degree of modelling accuracy in simulation examples (for example 10’s of dB better than a linear model), the performance obtained using experimentally obtained data was typically only one or 2 dB better than the linear models.

- The convergence is slow since the number of taps required to accurately model a physical HFT is large (several tens of thousands).

The TDNN was proposed as an alternative nonlinear model in Section 5.2 since it has the ability to generalize a wide range of nonlinear functions, and does not suffer from the curse of dimensional-
ity. The following conclusion can be made:

- Several computer examples showed that the *simulated* performance gains of the TDNN structure would not be as significant as obtained with the Volterra filter.

- When applied to *experimentally* obtained data, the TDNN models attained up to 5.6 dB higher converged ERLE than the Volterra models (Refer to Table 5.1).

- Structures with a single hidden layer with a small number of hidden nodes (i.e. one or two) are adequate for modelling the AEC process. In terms of complexity, this is welcome news, since the number of weights is a multiplicative function of the number of nodes between layers.

- A mixed linear sigmoid activation function was subsequently developed and it was demonstrated that by varying the linear range, several dBs of improved ERLE could be obtained (Refer to Figure 5.10).

- A TDNN is capable cf several dB’s of improvement in converged ERLE in the undermodelled case\(^1\) compared to an FIR structure with an equivalent number of taps in the TDL as measured using HFT #3 (Refer to Figure 5.14).

The conventional TDNN also has some detrimental effects, for example, poorer performance than the FIR structure at low and medium volumes. As a result, several new TDNN based architectures were developed to try to mitigate some of the problems associated with the conventional TDNN.

A new structure for nonlinear AEC, the *two stage neural filter*, was presented in Section 5.4.

- A simple gradient based learning algorithm was presented which has a complexity of \(2(n_1 + n_2) + 22\) operations per iteration, where \(n_1\) and \(n_2\) refer to the lengths of the TDLs in the nonlinear and linear sections respectively.

- Experimental results on HFT #6 indicate that this new structure is capable of 3 dB improved ERLE at the low volumes, and up to 11 dB improvement at high volume ranges, compared to the equivalent length FIR structure

---

1. Typically the TIP/TP ratio (see Section 4.4.2) is the limiting factor for undermodelled linear systems, however, it appears that for nonlinear systems, this may not be the case.
• The performance of proposed structure matches the linear FIR structure in the medium volume range.

It was demonstrated in Section 5.2 that by varying the linear region of the mixed linear-sigmoid activation functions, improved performance could be obtained. This was the motivation for developing a the VA-TDNN structure using a fully adaptive activation function, as discussed in Section 5.5. The VA-TDNN structure obtains the best overall performance of all the models tested on the experimental data (See Figure 5.20).

Finally, in Section 5.6, the VA-FIR-MLP was presented as a method for identification of cascaded nonlinear/linear/nonlinear systems. Computer simulations showed the efficacy of this model for the identification of low-order nonlinear cascaded systems, however the experimental results show that it is not as effective as the methods presented in the previous sections, although some gains could be obtained in the high volume ranges.

All of the structures developed in this chapter were tested using filtered noise as the input, even though the final application is for nonlinear acoustic echo cancellation, where the input signal is speech. It is well known that instantaneous gradient based learning algorithms suffer from slow convergence when coloured signals like speech are applied. In this chapter, the main performance criterion is the the steady-state ERLE value, so given sufficient training time, filtered noise was an appropriate input. However, for applications using speech as the input signal, it is necessary to invoke nonlinear training algorithms that are less sensitive to the characteristics of the input signals. This is the focus of Chapter 6.
Chapter 6

Conjugate Gradient Methods for Improved Performance

In previous chapters it was shown that several TDNN based structures can be applied to improve the overall steady-state MSE. However, the convergence rate in general was not as fast as the FIR structure trained with the NLMS algorithm. In this chapter, a new training algorithm is developed to improve the convergence rate without affecting the tracking ability. The algorithm is based on the fast conjugate gradient (FCG) algorithm which is modified for application to MLPs using the gradient backpropagation algorithm. In Section 6.1 the linear FCG algorithm is presented and then extended to neural networks. It is then used on the two stage neural filter developed in Section 5.4 and applied to both simulated and experimental data, including speech, to illustrate the performance advantages that can be obtained.

In Section 6.2, a variation on the FCG algorithm employing gradient reuse and a variable step-size line search algorithm is presented. This variation is for linear structures only. Section 6.2.9 presents a summary and discussion of the results.
6.1 Fast Conjugate Gradient Backpropagation

The conventional backpropagation (BP) algorithm presented in Section 3.7.1 is probably the most widely used supervised learning algorithm in neural network applications. However, with a large number of weights, the BP learning time is excessively long and its use become impractical. The conjugate gradient algorithm is well suited for the neural network learning problem since it is fast, simple and requires little additional storage space (only the current and previous gradient and search vectors and weights must be stored). The CG method speeds up the BP learning time significantly and does not suffer from the inefficiencies and possible instabilities that arise using the BP with a fixed step size. In fact, the CG algorithm has been found in some studies [72] to be an order of magnitude faster than the conventional BP using momentum.

Partial CG methods introduced in Section 3.4 allow further simplification of the CG algorithm by restricting the weight updates for a number of iterations $k<m$, where $m$ is the filter order. Partial CG algorithms provide a stepping point for the formulation of fast (i.e. numerically less intensive) versions of the CG algorithm.

6.1.1 Fast Conjugate Gradient Algorithm for Linear Adaptive Filters

Boray and Srinath [74] recently developed a fast conjugate gradient algorithm (FCG) for linear adaptive filtering using an averaged instantaneous gradient over a window of past sample values. They showed that the advantages of this windowed approach are (i) better tracking and convergence is achieved in nonstationary environments with correlated data compared to the RLS algorithm, and (ii) there are no stability problems associated with an exponential forgetting factor as in the RLS algorithm. The CG algorithm achieves convergence speed comparable to the RLS algorithm even when the input signal autocorrelation matrix is ill conditioned [74]. However, the CG
computational burden is still high compared to variations based on the LMS algorithm [155] hence the FCG algorithm is a welcome method to simplify the CG further.

In the conventional CG algorithm the gradients are normally calculated as true gradients meaning that at least \( m \) conjugate directions can be calculated, and that all the training data is available for calculation of the gradient. However, in real time filtering and system identification an on-line method of approximating the gradient is required. If we use an instantaneous gradient estimate, as is done in the LMS algorithm, the CG algorithm will terminate in one step and essentially defaults to the LMS algorithm. This is because there will not be any more directions conjugate to the initial direction vector. However, a better approximation to the gradient can be obtained by calculating the estimate based on a window \( n_w \) of past values of inputs [74]. The algorithm tries to minimize a partial cost function constructed by summing \( n_w \) instantaneous cost functions using the current weight vector \( w(n) \):

\[
J_{partial}(n) = \sum_{i=0}^{n_w-1} J_{inst}(n-i) \bigg|_{w(n)}
\]  

It can be shown [74] that there will be at least \( \min(m,n_w) \) linearly independent direction vectors in the gradient estimate, where \( m \) is the filter order. Specifically the instantaneous gradient estimate at time \( n \) is replaced by a windowed estimate as follows:

\[
g_k(n) = \left[ \nabla f(w_k(n)) \right] = \left( \frac{2}{n_w} \right)^{n_w-1} \sum_{i=0}^{n_w-1} \left\{ \left[ w_k^T(n) x(n-i) - d(n-i) \right] x(n-i) \right\}
\]  

The linear FCG algorithm is the same as that listed in Section 3.4.4 except that the computation of \( g_k(n) \) is done according to (6.2) and the iteration count is terminated when \( k=n_w \).
Simplification of the FCG algorithm. The computation of the optimum step size $\alpha_k$ in (3.44) still requires $2mn_w$ multiplies and one division. By removing the calculation of $\alpha_k$ and replacing it with a constant value, the calculation of $p$ and $y$ are also no longer required, thus simplifying the algorithm further. However, instead of using a fixed step size as proposed in [74], a normalized step size is used in all subsequent simulations, defined by

$$\tilde{\alpha}(n) = \frac{\alpha}{x^T(n)x(n) + \varepsilon}$$ \hfill (6.3)

where $0 < \alpha < 2$ and $\varepsilon$ is some small value. This slight variation of the algorithm is used in all the simulations. It should be pointed out that by avoiding the calculation of $\alpha_k$ at each iteration, there is no guarantee that the successive direction vectors will be truly conjugate. This will result in reduced convergence rates over that of the CG algorithm. The FCG algorithm for linear FIR filters is summarized below:

---

**Fast Conjugate Gradient Algorithm for Linear FIR Filters:**

**Initialization:** $w_0(0) = 0$

For each iteration $n$, do steps 1, 2 and 3.

**Step 1:**

a) Starting with an initial weight vector $w_0(n)$ compute the following:

$$g_0(n) = \frac{\nabla f(w_0(n))}{\|\nabla f(w_0(n))\|}$$ \hfill (6.4)

b) set $d_0(n) = g_0(n)$

c) compute the normalized step size $\tilde{\alpha}(n)$ according to (6.3)
6.1 Fast Conjugate Gradient Backpropagation

**Step 2:**

Repeat for $k=0,1,\ldots,n_w-1$

1. Set $w_{k+1}(n) = w_k(n) + \tilde{\alpha}d_k(n)$

2. Compute the gradient estimate at the new weight vector $w_{k+1}(n)$

$$g_{k+1}(n) = \left[ \nabla f(w_{k+1}(n)) \right]$$ (6.5)

3. Unless $k=n_w-1$, obtain the new direction vector

$$d_{k+1}(n) = -g_{k+1}(n) + \beta_k d_k(n)$$ (6.6)

where

$$\beta_k = \frac{g_{k+1}(n)^T g_{k+1}(n)}{g_k(n)^T g_k(n)}$$ (6.7)

and repeat Step 2 (a).

**Step 3:**

Replace $w_0(n)$ by $w_m(n)$ and go back to Step 1.

---

6.1.2 Extension of the FCG Algorithm to Neural Networks

We can extend the FCG algorithm to the nonlinear case, for neural networks. The nonlinear FCG (NFCG) algorithm is similar to the algorithm presented in the previous section. The differences are

1. the network is nonlinear
2. the errors must be computed for hidden layers and not just the output layer
3. the previous values of the hidden layer outputs must be retained as well as the output layers in order to compute the windowed gradient.

The cost function to be minimized takes the form of equation (6.1) however, the vector $\mathbf{w}(n)$ is now an $M^{\text{th}}$ order supervector defined by

$$\left[ \mathbf{w}(n) \right]^T = \left[ \left[ \mathbf{w}^{(0)}(n) \right]^T , \left[ \mathbf{w}^{(1)}(n) \right]^T , \left[ \mathbf{w}^{(2)}(n) \right]^T , \ldots, \left[ \mathbf{w}^{(L)}(n) \right]^T \right]$$ (6.8)

where
\[ w^{(l)}(n) = \begin{bmatrix} w^{(l)}_{11}(n), w^{(l)}_{12}(n), \ldots, w^{(l)}_{N_l N_{l-1}}(n) \end{bmatrix}^T \]  \hspace{1cm} (6.9)

is the weight vector connecting layer \( l \) to layer \( l+1 \) at time \( n \), \( M \) equals the total number of weights in the network and \( L \) is the total number of layers in the network. The windowed gradient vector \( g(n) = [\nabla f(w(n))] \) is also now of length \( M \), with individual elements corresponding to the weights listed in equation (6.9).

The gradient is computed using the average squared error of a window of training input/output pairs. Similar expressions for the CG and BP algorithms have been developed by several authors, including Charlamous [73], Johansson et. al. [72], as well as Adeli and Hung [156]. However, these expressions are based on the batch training mode using the full set of input/output pairs, and not the windowed method proposed here.

Errors are backpropagated to previous layers in the same way as the conventional BP algorithm.

The important point is that the window is moved for each new sample of the input that comes in i.e. it is a sliding window of past input/output pairs. The NFCG is summarized below:

Nonlinear FCG (NFCG) Algorithm

**Initialization:** Set weights and biases to random values between -1 and +1.

For each iteration \( n \), do Steps 1, 2, and 3.

**Step 1.** a) Starting with an initial weight vector \( w_0(n) \), compute the following:

\[ g_0(n) = [\nabla f(w_0(n))] = \left( \frac{2}{n_w} \right)^{\frac{1}{2}} \sum_{i=0}^{n_w-1} g_{inst}(n-i) \begin{bmatrix} w_0(n), u^{(0)}(n-i), d(n-i) \end{bmatrix} \]  \hspace{1cm} (6.10)
6.1 Fast Conjugate Gradient Backpropagation

b) set \( d_0(n) = -g_0(n) \)

c) compute the normalized step size parameter \( \alpha \) according to:

\[
\tilde{\alpha} = \frac{\gamma}{\varepsilon + \| \mathbf{u}^{(0)}(n) \|_2^2} = \frac{\gamma}{\varepsilon + \mathbf{u}^{(0)}(n)^T \mathbf{u}^{(0)}(n)}
\]  

(6.11)

Note that \( \tilde{\alpha} \) could be replaced by a fixed step size here if desired:

**Step 2.** Repeat for \( k=0,1,..,n_w-1 \) where \( n_w \leq m \)

\( a) \) set \( \mathbf{w}_{k+1}(n) = \mathbf{w}_k(n) + \alpha \mathbf{d}_k(n) \)

\( b) \) compute an estimate of the gradient at \( \mathbf{w}_{k+1}(n) \):

\[
g_{k+1}(n) = \left[ \nabla f(\mathbf{w}_{k+1}(n)) \right] = \left( \frac{2}{n_w} \right) \sum_{i=0}^{n_w-1} g_{\text{inst}}(n-i) \bigg|_{\mathbf{w}_{k+1}(n), \mathbf{u}^{(0)}(n-i), d(n-i)}
\]  

(6.12)

c) Unless \( k=n_w-1 \), set \( \mathbf{d}_{k+1}(n) = -g_{k+1}(n) + \beta_k \mathbf{d}_k(n) \), where:

\[
\beta_k = \frac{g_{k+1}^T(n) \mathbf{g}_{k+1}(n)}{g_k^T(n) \mathbf{g}_k(n)}
\]  

(6.13)

Note that if \( \beta_k > 1 \), go directly to Step three.

Repeat Step 2 a).

**Step 3.** Replace \( \mathbf{w}_0(n) \) by \( \mathbf{w}_k(n) \) and go back to Step 1.

\( g_{\text{inst}}(n-i) \) is the instantaneous gradient calculated with the current network weight vector \( \mathbf{w}_0(n) \) and past inputs \( \mathbf{u}^{(c)}(n-i) \) and \( d(n-i) \). Both \( g_{\text{inst}}(n-i) \) and \( \mathbf{w}_0(n) \) are vectors of length \( M \), where \( M \) is the total number of weights in the network. The calculation of individual elements of the instantaneous gradient vector \( g_{\text{inst}}(n-i) \) is done by performing the following steps:
\[ g_{ij}^{(l)}(n-i) = \delta_j^{(l+1)}(n-i) \cdot u_i^{(l)}(n-i) \] (6.14)

where \( g_{ij}^{(l)}(n-i) \) is the instantaneous gradient from the data \( i \) time steps in the past for weight \( w_{ij}^{(l)}(n) \) in the \( l \)-th layer, and:

\[
\delta_j^{(l)}(n-i) = \begin{cases} 
-2e(n-i)\phi'(s_j^{(L)}(n-i)) & \text{...} l = L \\
\phi'(s_j^{(l)}(n-i)) \cdot \sum_{k=1}^{N_{l-1}} \delta_k^{(l+1)}(n-i) \cdot w_{jk}^{(l)}(n) & \text{...} 1 \leq l \leq L - 1
\end{cases}
\] (6.15)

\[ e(n-i) = d(n-i) - N[w_{k+1}(n), u^{(0)}(n-i)] \] (6.16)

Note that \( \phi[w_{k+1}(n), u^{(0)}(n-i)] \) represents the nonlinear output of the neural network at time \( n \) using the current weight vector \( w_{k+1}(n) \) with past input vectors \( u^{(0)}(n-i) \). An illustration of the terms is shown in Figure 3.7 and Figure 3.8.

**Complexity.** The choice of \( n_w = 1 \) implies no averaging in the gradient estimate and the NFCG algorithm reverts to the BP algorithm. For higher values of \( n_w \) the complexity approaches that of algorithms that use the second derivative for obtaining the optimum step size and direction which have complexity \( O(m^3) \)[75]. The complexity of the NCG algorithm is \( O(mn_{n_w}^2) \) since in Step 2, the weights are updated \( n_w \) times per iteration and the calculation of the averaged gradient is \( O(mn_{n_w}) \).
6.1.3 Computer Simulation

In this section, we apply the NFCG algorithm to the identification of a nonlinear system constructed by generating a signal which is hard limited and convolved with an exponentially decaying 50 tap impulse. The system is illustrated in Figure 6.1. The input signal $x(n)$ is obtained by a first order autoregressive (AR) process according to:

$$ y(n) = 0.9y(n-1) + v(n) \quad (6.17) $$

where $v(n)$ is a unit variance white noise sequence. The hard limiter has a linear region up to 0.5, beyond which the output is clipped with a slope of 0.2. Two hundred independent trials are used in the averaging of the normalized MSE.

![Diagram](image)

**FIGURE 6.1 System identification model.**

The results in Figure 6.2 show that for the AR input, the NFCG algorithm converges at a rate much faster than the conventional BP algorithm, depending on the size of the gradient averaging window.
$n_w$. The larger the choice of $n_w$, the higher the convergence rate. The final misadjustment is approximately -18 dB for all cases.

![Simulation Results: BP and NFCG Algorithms](image)

**FIGURE 6.2** Simulation results showing the averaged NMSE performance of the BP and NFCG algorithms with $n_w=2$, 5, and 10 for the system identification model of Figure 6.1. Two hundred independent trials are used in the averaging process.

*Convergence Rate Improvement*. The convergence rate improvement is not a linear function of the window size. For example, the BP algorithm, which is equivalent to the NFCG with $n_w=1$, takes approximately 1400 iterations to reach -15 dB NMSE. For window sizes $n_w=2$, 5, and 10, the number of iterations required to reach the same NMSE are approximately 600, 200 and 150 respectively. As a result, it can be seen that the convergence rate improvement becomes progressively smaller for large window sizes, and that for $n_w>5$, the convergence rate improvements are small.
6.1.4 Experimental Results

In this section two experiments are performed using data collected from actual LREM and HFT components. In the first experiment, a filtered noise signal is applied to loudspeaker SPK#2 (refer to Appendix B) which is mounted in a standard baffle and placed inside an anechoic chamber. This is the reference signal. The primary signal is picked up by MIC#1. The primary and reference signals are then applied to a conventional TDNN structure which is trained with the BP and NFCG algorithms.

In experiment #2, data is collected inside Conference Room #2 using HFT#6. Real speech signals are applied as the reference signal. The primary and reference signals are then applied to the two stage neural filter (See Section 5.4) which is trained with the BP and NFCG algorithms. For comparison purposes, the performance of an FIR filter trained with the accelerated SFTF algorithm (see Appendix D.5) is also shown. The accelerated SFTF algorithm is used to remove the long training time associated with LMS based training algorithms when using speech inputs, which may be as long as 10 seconds [157].

Experiment #1, Noise Input. The volume is 100 dB SPL as measured at 0.5 meters from the loudspeaker. The microphone is placed 15 cm. from the loudspeaker output. The signals are sampled at 16 kHz and are later transferred to a computer for off-line analysis. Two adaptive filter structures were tested to identify the system (i) a 150 tap linear transversal filter trained using the NLMS algorithm (ii) a 3 layer TDNN with 150 input taps trained with both the BP and NFCG algorithms. The experimental results shown in Figure 6.3 show the results for all cases. The NLMS has fast convergence but is incapable of obtaining an ERLE of greater than 19 dB due to the nonlinear loudspeaker. The TDNN trained with the BP algorithm is capable of identifying the system more effectively and achieves 25 dB ERLE but the initial convergence is much slower than the NLMS
algorithm. Training the TDNN using the NFCG with a window size \( n_w = 5 \) results in convergence speed equivalent to the NLMS structure as well as obtaining 24 dB ERLE.

![ERLE Convergence at 100 dB SPL](image)

**FIGURE 6.3** Experiment #1 results comparing converged ERLE curves of a 150 tap FIR structure trained using the NLMS algorithm with that of a TDNN trained with the BP and NFCG algorithm.

**Experiment #2, Speech Input.** The average volume of the speech signal as measured 0.5 m from the loudspeaker is 95 dB SPL, which is a comfortable listening level 6-10 ft. from the HFT. The HFT is placed in the middle of the conference table. The parameters are listed in Table 6.1.

**TABLE 6.1** Experiment #2 parameters.

<table>
<thead>
<tr>
<th>Item</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>160,000 samples @16 kHz sampling, 95 dB SPL average volume at 0.5 m.</td>
</tr>
<tr>
<td>FIR Trained with Accelerated SFTF</td>
<td>( N = 600, \lambda = 0.9998 ), acceleration factor=0.95, soft initialization constant=200.</td>
</tr>
<tr>
<td>Two stage neural filter trained with NFCG algorithm</td>
<td>( N_1 = 150, N_2 = 450 ), number of hidden nodes=1, neural network normalized step size ( \alpha = 0.5 ), nlms step size ( \alpha = 0.5 ), window size ( n_w = 5 ) for TDNN section</td>
</tr>
</tbody>
</table>
Figure 6.4 shows the speech signal amplitude as a function of time. The converged ERLE results shown in Figure 6.5 and Figure 6.6 indicate that the proposed structure/algorithm outperforms the FIR structure trained with the accelerated SFTF algorithm by approximately 5 dB.

FIGURE 6.4 Reference signal speech signal.
FIGURE 6.5 Experiment #2 results. Converged ERLE results with speech input. Gaps show where pauses in speech are located.

FIGURE 6.6 Experiment #2 results. Close up of speech period between 6 and 8 seconds. The two stage neural filter trained with proposed algorithm achieves approximately 5 dB higher ERLE that the FIR filter trained with stabilized SFTF algorithm.
6.1.5 Discussion

The results presented in this section have shown that the NFCG algorithm is capable of improving the convergence rate of neural network based adaptive filters. When applied to the two stage neural filter from Section 5.4, the NFCG algorithm achieves a 5 dB improvement in ERLE compared to the accelerated SFTF algorithm when trained with real speech signals at loud volumes where loudspeaker nonlinearities become significant. Simulation results in Section 6.1.3 also indicate that by varying the size of the gradient window $n_w$, we can obtain improved convergence speed with a corresponding increase in complexity. A window size of $n_w=5$ was found sufficient to speed the initial convergence rate of the two stage neural filter to be no worse than the linear FIR trained with the NLMS algorithm, when applied to data collected from a loudspeaker/microphone placed in an anechoic chamber.

One of the important features of the NFCG algorithm is that the gradient window can be made arbitrarily small to "tailor" the algorithm to a particular application. Thus, where a modest increase in convergence is desired without compromising tracking ability, a small $n_w$ can be chosen. Low values of $n_w$ will result in slower convergence, however, the advantages are reduced complexity and faster tracking capability. This is important for AEC applications where reduced complexity is of utmost importance. It has been shown that linear algorithms based on instantaneous gradient LMS type updates have superior tracking ability compared to faster algorithms based on least squares minimization [96]. The performance of partial CG algorithms falls between algorithms based on instantaneous and full gradient (i.e. least squares) methods [71]. Hence, as the window size becomes large and the convergence improves, the dynamic tracking performance will suffer. It is therefore reasonable to assume that a similar performance trade-off will be noticed in extensions of CG methods to the nonlinear domain, like the NFCG.
6.2 Conjugate Gradient Reuse Algorithm with Dynamic Line Search

The fast CG algorithm presented in Section 6.1.1 was obtained by substituting the optimum step size $\alpha_k$ with a fixed step size, thus avoiding the computation of the difference vector $y_k(n)$ and its gradient $p_k(n)$. It is possible however, to estimate a reasonable value of $\alpha_k$ using an inaccurate line search technique.

In this section a new linear algorithm is presented which combines the FCG algorithm and the Modified Variable Step Size (MVSS) algorithm [56] to provide a convergence/tracking performance/complexity trade-off. The MVSS algorithm is used to perform an inaccurate one dimensional line search along the conjugate direction vector $d_k(n)$ at each iteration $k$. By selecting an appropriate number of steps performed during the line search, it is possible to achieve the same performance as the FCG algorithm but using a smaller window size, and therefore reduced complexity. This algorithm is called the Variable step size CG (VCG) algorithm.

A simplified version of the VCG algorithm is also presented that reuses weight updates (i.e. gradient reuse) to avoid calculating gradients and conjugate directions at every sample $n$. This simplified algorithm only invokes the conjugate gradient update every $P$th sample resulting in an overall complexity reduction by a factor of $P$ as compared to the FCG algorithm. This new algorithm is called the conjugate gradient reuse (CGR) algorithm.

Simulation results show that improved convergence and tracking is obtained compared to the NLMS, RLS, FCG and MVSS algorithms when the input data is correlated and the environment is nonstationary.
6.2.1 Inaccurate Line Search

There are two kinds of line searches, accurate line searches and inaccurate line searches. Accurate line searches are very attractive theoretically, however, they are very expensive to carry out and the algorithm may spend a considerable amount of computing effort locating the exact minimum point on a descent direction.

Typically, an accurate line search involves bracketing or straddling the minimum and then using a cubic interpolation algorithm to determine the exact minimum [72]. A stochastic line search algorithm has been presented in [158] which recursively minimizes the sum of squared errors on a linear manifold. It is similar to fast RLS algorithms since it iteratively calculates the optimum step size parameter. Other line search techniques such as the scaled conjugate gradient (SCG) algorithm [159] have been proposed in the literature, however, all the aforementioned accurate line searches are formulated for full gradients and do not work for partial CG methods.

Inaccurate line searches typically terminate a search before it has converged. Several popular techniques are Armijo’s Rule [160] and the Goldstein Test [71]. The basic idea is to guarantee a proper, namely not to large and not too small, step size is selected. The FCG algorithm is the extreme case of an inaccurate line search, where only a single fixed step is taken toward the minimum. The MVSS algorithm may also be applied as an inaccurate line search algorithm and this is described next.

6.2.2 The MVSS Line Search Algorithm

The MVSS algorithm [56] is a variable step size (VSS) LMS type algorithm that dynamically adjusts the step size during the search for the minimum of a performance surface. The MVSS calculates at each iteration a step size based on the autocorrelation of adjacent error samples. When
the error correlation is large, dynamic step size control adjusts the step size to be large thus speeding up the convergence. Thus, when the filter is far from the optimum and the autocorrelation of the error signal is large, the step size is large. This has the effect of reducing the number of iterations needed to find the minimum of a particular conjugate direction by increasing the line search step size where appropriate and in this respect is similar to the line search algorithm in the SCG algorithm and Armijo's rule. Specifically, the step size is updated by the following formulas:

\[
\mu(k+1) = \begin{cases} 
\mu_{\text{max}} & \mu(k+1) \geq \mu_{\text{max}} \\
\mu_{\text{min}} & \mu(k+1) \leq \mu_{\text{min}} \\
\zeta \mu(k) + \gamma \rho^2(k) & \mu_{\text{min}} < \mu(k+1) < \mu_{\text{max}}
\end{cases}
\] (6.18)

where \( \rho(k) = \Gamma \rho(k-1) + (1 - \Gamma) e(k) e(k-1) \) (6.19)

\[
\text{and} \begin{cases} 
0 < \zeta < 1 \\
\Gamma < 1 \\
\gamma > 0
\end{cases}
\] (6.20)

The parameter \( \gamma \) controls the convergence time. The parameter \( \zeta \) controls the averaging of the step size update and \( \Gamma \) controls the averaging time constant of the filtered error update. The parameter \( \rho \) gives a short time estimate of the error signal autocorrelation. Typical parameter values are \( \zeta=0.97, \Gamma=0.99, \gamma=10^{-5} \) [56]. The advantage of the MVSS algorithm over the standard VSS algorithm is its relative *insensitivity* to noisy signals due to the time average autocorrelation process.

This technique is adopted in the FCG algorithm to dynamically control the step size update during the line search. A measure of the "minimum" of the line search can be obtained by examining \( e(j) \) and \( e(j-1) \) where \( j \) is the line search iteration count. If \( e(j) < e(j-1) \) then the algorithm is still searching for the minimum of the performance surface along this particular direction. If
\[ e(j) \geq e(j-1) \]
then the minimum has been reached, at which point we exit the search and replace the initial weight vector \( w_0 \) with the one used to generate \( e(j-1) \).

### 6.2.3 Maximum and Minimum Step Sizes

Due to the inaccuracy of the line search, the step size \( c_k \) will not be exact and the direction vectors \( d_k \) will not be Q-conjugate. For general nonlinear problems, an indication that the algorithm is getting stuck, or near the minimum, is that very small steps are being taken. Subsequently orthogonality between successive gradients is lost, \( g_{k+1} = g_k \) [72] and it is possible that the calculation of \( \beta_k \) will be close to or larger than 1 and cause instability.

Proakis [161] demonstrated that the conventional CG algorithm resembles the operation of a first-order recursive filter whose output \( d_k \) is given by equation (6.6). An \( m \)-dimensional filter is in effect a set of \( m \) identical single-pole (low pass) filters operating in parallel which corresponds to filtering the gradients with a time-variant filter. This can provide faster convergence than the conventional LMS algorithm, however, if \( \beta_k \) is close to or larger than 1, as may happen with partial CG methods using inaccurate line searches, successive iterations will only serve to move the weight vector away from the optimum value and make the algorithm unstable. Proakis found it necessary to limit the value of \( \beta_k < 1 \) in a channel equalization experiment and obtained the conditions for stability which can be expressed as follows [161]:

\[
0 < \mu_k < \frac{2(1+\beta_k)}{\lambda_{\text{max}}} \tag{6.21}
\]

where \( 0 < \beta_k < 1 \) and \( \lambda_{\text{max}} \) is the maximum eigenvalue of the input data. Specifically, the gradient averaging extends the upper limit of the region of stability of \( \mu_k \) from \( 2\lambda_{\text{max}} \) to \( 2(1+\beta_k)\lambda_{\text{max}} \) but \( \beta_k \) must be kept below 1.
6.2 Conjugate Gradient Reuse Algorithm with Dynamic Line Search

In [162] Shanno shows that the use of inaccurate line searches using the Polak-Ribiere or Fletcher-Reeves method may yield conjugate directions $d_k$ which are not descent directions, resulting in numerical instability. A method for calculating $d_k$ is derived which guarantees a descent direction. Since Shanno's method is not used here for computing successive $d_k$, the maximum step size boundary in (6.21) is checked each iteration and then a limit $\mu_{max}/10$ is imposed on the minimum step size.

6.2.4 Variable Step Size CG Algorithm using MVSS Line Search

The complete algorithm is summarized below and uses triply indexed parameters. The parameter $n$ refers to the main iteration number, where the data is shifted in on a sample by sample basis, $k$ represents the conjugate direction iteration count and $j$ represents the line search iteration count.

Conjugate Gradient Algorithm with MVSS Line Search (VCG)

**Initialization:** $w_0(0)=0, \beta_k(0)=0$.

For each iteration $n$, do steps 1,2 and 3.

**Step 1:**

1a) Shift in new data into vector $x(n)$

1b) Starting with an initial weight vector $w_0(n)$, compute the initial error:

$$e(n) = w_0^T(n) x(n) - d(n) \quad (6.22)$$

1c) Compute the maximum step size according to

$$\mu_{max}(n) = \frac{1 + \beta_k(n)}{x^T(n)x(n)} \quad (6.23)$$
1d) Compute the initial windowed gradient estimate:

\[ g_0(n) = \left( \frac{2}{n_w} \right)^{n_w - 1} \sum_{i = 0}^{n_w - 1} \left\{ \left[ w_0^T(n) x(n - i) - d(n - i) \right] x(n - i) \right\} \]  

(6.24)

1e) set \( d_0(n) = -g_0(n) \)

Step 2: Repeat for \( k = 0, 1, \ldots, n_w - 1 \) where \( n_w \leq m \)

2a) set \( \mu_{k, 0}(n) = \mu_{max}(n) \) and \( w_{k, 0}(n) = w_k(n) \)

Repeat Steps 2b-1) through 2b-4) for \( j = 1, \ldots, n_w \) where \( n_w \leq m \)

2b-1) Set \( w_{k, j}(n) = w_{k, j-1}(n) + \mu_{k, j}(n) d_k(n) \)

2b-2) Compute the new error output using

\[ e_{k, j}(n) = w_{k, j}^T(n) x(n) - d(n) \]  

(6.25)

2b-3) Adjust the step size

\[ \mu_{k, j}(n) = \begin{cases} 
\mu_{max}(n) & ; \mu_{k, j}(n) \geq \mu_{max}(n) \\
\mu_{min}(n) & ; \mu_{k, j}(n) \leq \mu_{min}(n) \\
\zeta \mu_{k, j-1}(n) + \gamma \rho_j^2(n) & ; \mu_{min}(n) < \mu_{k, j}(n) < \mu_{max}(n) 
\end{cases} \]  

(6.26)

where \( \rho_j(n) = \Gamma \rho_{j-1}(n) + (1 - \Gamma) e_j(n) e_{j-1}(n) \)  

(6.27)

2b-4) if \( e_{k, j}(n) > e_{k, j-1}(n) \) then proceed to Step 2c), else goto Step 2b-1).

2c) restore the “optimum” weight vector \( w_{k+1}(n) = w_{k, j}(n) \) for this direction.

2d) Unless \( k = m_w - 1 \), set \( d_{k+1}(n) = -g_{k+1}(n) + \beta_k(n) d_k(n) \), where:
6.2 Conjugate Gradient Reuse Algorithm with Dynamic Line Search

\[ \beta_k(n) = \frac{\mathbf{g}_{k+1}(n) \mathbf{g}_{k+1}(n)^T}{\mathbf{g}_k(n) \mathbf{g}_k(n)} \quad \text{and} \]

\[ \mathbf{g}_{k+1}(n) = \left[ \nabla f(\mathbf{w}_{k+1}(n)) \right] \]

\[ = \left( \frac{2}{n_w} \right) \sum_{i = n - n_w + 1}^{n} \{ \mathbf{w}_{k+1}^T(n) \mathbf{x}(i) - d(i) \} \mathbf{x}(i) \]  

(6.28)  

(6.29)

If \( \beta_k(n) > 1 \), go directly to Step 3, otherwise go to Step 2.

**Step 3.** Replace \( \mathbf{w}_0(n+1) \) by \( \mathbf{w}_k(n) \), and go back to Step 1.

The MVSS line search is performed in Step 2b-1) where the newly computed direction vector \( \mathbf{d}_k \) is used to successively update \( \mathbf{w}_k \).

Note that \( \mathbf{d}_k(n) = -\mathbf{g}_k(n) = -[\nabla f(\mathbf{w}_k(n))] \) only during the first iteration of the line search when \( k=0 \) and that during successive iterations, \( \mathbf{d}_k \) will change. We have imposed an allowed limit of \( n_w \) steps of loop 2b) to reach the minimum of a particular conjugate direction during the line search. This factor was chosen to place a limit on the number of steps taken should the direction estimate be in the wrong direction. If the minimum of a particular conjugate direction has not been reached before the next conjugate direction is calculated, or if the gradient estimate is poor, there is no guarantee that the new directions will be conjugate with respect to one another. This will slow convergence, however, it is still superior to the NLMS algorithm.
6.2.5 Gradient Reuse

The VCG provides an averaged gradient which also points in the optimum direction towards the minimum of the performance surface based on the available information in the gradient window. If we assume that the performance surface does not change too rapidly, then it is safe to assume that by reusing the conjugate gradient updates (as opposed to conjugate direction updates), we can still step in the right direction and at the same time avoid the calculation of the true gradient. If we only allow a gradient calculation every $P$ input samples, we obtain a reduction in the complexity by a factor of $P$ over the VCG. This is the basis of the VCG algorithm with gradient reuse (VCGR). A variation of this idea was proposed by Hush and Salas [163] for reducing the computational complexity of backpropagation weight updates in neural networks where they showed that the convergence rate speed-up or slow-down is a related to the reuse rate. It is also possible to reuse the weight updates several times per sample iteration $n$, i.e. $P<1$, however for the application described here, we only update the weights once per sample with a gradient calculation every $P$ samples. The trade-off is that the convergence rate will become poorer in correlated environments as $P$ increases. However, it provides a basis for trading computational complexity for performance in the same way as the gradient window size $n_w$. The algorithm is the same as the VCG except for the following changes which are indicated with an asterisk in bold type:
VCG with Gradient Reuse (VCGR)

**Initialization:** $w_0(0) = 0, \beta_k(0) = 0$.

***count=0:
For each iteration $n$, do steps 1, 2 and 3.

**Step 1:**

1a) Shift in new data into vector $x(n)$

1b) Starting with an initial weight vector $w_0(n)$, compute the initial error:

$$e(n) = w_0^T(n) x(n) - d(n) \quad (6.30)$$

***count=count+1

***if count=P, continue, else goto Step 3)
Perform rest of Step 1) and Step 2) here

**Step 3.**

*** If count=1,

$$\Delta w_k(n) = w_k(n) - w_o(n) \quad (6.31)$$

$$w_o(n + 1) = w_k(n) \quad (6.32)$$

*** else

$$w_o(n + 1) = w_o(n) + \Delta w_k(n) \quad (6.33)$$

Replace $w_o(n+1)$ by $w_k(n)$, and go back to Step 1.
6.2.6 Complexity

In the regular CG algorithm, the number of multiplications required in Step 1) is $3m^2$ per gradient calculation or $6m^2$ total. In step 2), the number of multiplications per sample is $m \left( 6m^2 + 6m \right)$ per sample for an overall total of $6m^3 + 12m^2$. In the VCG, the number of multiplications per sample in Step 1) is $2mn_w + 1$. Step 2b is done $R$ times resulting in a complexity of $n_w R (2m + 6)$ multiplications per sample where $R \leq n_w$. Step 2d is done $n_w - 1$ times for a complexity of $(n_w - 1) (2mn_w + 3m)$ multiplications per sample. Summing all of these contributions, the overall complexity of the VCG is equal to:

$$
(2mn_w + 1) + n_w R (2m + 6) + (n_w - 1) (2mn_w + 3m)
$$

(6.34)

multiplications per sample. For $R=1$ the VCG will default to the FCG algorithm and for $n_w=1$, it reverts to the NLMS algorithm. The standard RLS algorithm has complexity of $(2m^2 + 4m)$. The VCG has a slight increase in computational complexity over the FCG algorithm due to the conjugate direction reuse rate $R$. However, if fewer than $R$ successive steps of 2b) are needed before the minimum is reached, this estimate of complexity would represent an upper bound. It is possible to limit the value of $R$ to some value smaller than $n_w$ to provide a limited complexity increase. Simulation results will show that by using a restricted $R$, it is possible to obtain the same performance with the VCG as with the FCG algorithm, even though the latter requires a larger window size to obtain this performance and is therefore more complex.

The VCGR algorithm only performs gradient calculations every $P$ samples, and this reduces the complexity to:

$$
\frac{2mn_w + n_w R (2m + 6) + (n_w - 1) (2mn_w + 3m)}{(m + 1) + \frac{P}{P}}
$$

(6.35)
multiplications per sample for \( R \geq 2 \). For \( P=1 \), the simplified algorithm reverts to the VCG.

Table 6.2 gives comparative complexities of the CG, FCG, VCG, VCGR and RLS algorithms for \( m=50, n_w=5, R=2 \) and \( P=3 \).

**TABLE 6.2 Comparison of algorithm complexity.**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Mult./sample for ( m=50, n_w=5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG</td>
<td>( 6m^3 + 12m^2 )</td>
</tr>
<tr>
<td>FCG</td>
<td>((2mn_w + 1) + n_w(2m + 6) + (n_w - 1)(2mn_w + 3m))</td>
</tr>
<tr>
<td>VCG</td>
<td>((2mn_w + 1) + n_wR(2m + 6) + (n_w - 1)(2mn_w + 3m))</td>
</tr>
<tr>
<td>VCGR</td>
<td>(\frac{2mn_w + n_wR(2m + 6) + (n_w - 1)(2mn_w + 3m)}{P})</td>
</tr>
<tr>
<td>RLS</td>
<td>(2m^2 + 4m)</td>
</tr>
</tbody>
</table>

\[ x(n) = 0.9x(n-1) + 0.2v(n) \]

![System identification model](FIGURE 6.7 System identification model. An uncorrelated noise source with variance \( \sigma_N^2 \) is added to the adaptive filter output \( y(n) \) to produce an SNR of 50 dB.)

**6.2.7 Computer Simulations**

In this section, we apply the VCG algorithm to the problem of system identification as illustrated in Figure 6.7.
The unknown system is modelled by an impulse 50 taps long which is obtained from an exponentially decaying set of random values between ±1. This is representative of an LREM obtained in a highly damped conference room or in automobiles where both fast convergence and tracking are required. The input to the system is a coloured noise sequence obtained from a single pole autoregressive process described by equation (6.17). This signal is then filtered by the unknown system and finally, a small amplitude uncorrelated white gaussian noise signal is then added to the system output to produce a desired signal to noise ratio of 50 dB. In order to demonstrate the tracking capabilities of the VCG, the unknown system impulse response is changed halfway through the data sequence by multiplying all coefficients by -1.0. This change in the transfer function will cause a temporary increase in the mean square error as the algorithms try to readjust the weights to the new optimum weight vector and gives some measure of the tracking performance of a training algorithm. Subsequently, the NMSE convergence curves for the RLS, NLMS, FCG, MVSS and VCG are plotted for comparison. The NMSE curves are obtained by averaging the error and desired signals over 100 independent runs and then smoothing according to the following formula,

\[
NMSE(n) = 10\log\left( \frac{\sum_{r=0}^{50} [\overline{e}_r(n-r)]^2}{\sum_{r=0}^{50} [\overline{d}_r(n-r)]^2} \right) dB
\]

(6.36)

where \( \overline{e}_r(n) \) and \( \overline{d}_r(n) \) represent the averaged error and desired signals at time \( n \) averaged over 100 independent trials and \( r \) represents the window values over which these averages are then smoothed, in this case equal to 50. A summary of the parameters used in the simulations is listed in Table 6.3.
### Table 6.3 List of parameters used for simulations #1, #2 and #3.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>#Taps m</th>
<th>( \alpha )</th>
<th>( \lambda )</th>
<th>( \mu_{max} )</th>
<th>( \mu_{min} )</th>
<th>( \xi )</th>
<th>( \Gamma )</th>
<th>( \gamma )</th>
<th>( n_w )</th>
<th>SNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLMS</td>
<td>50</td>
<td>0.5</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50 dB</td>
</tr>
<tr>
<td>RLS</td>
<td>50</td>
<td></td>
<td>0.997</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50 dB</td>
</tr>
<tr>
<td>MVSS</td>
<td>50</td>
<td></td>
<td>1.0</td>
<td>1e-5</td>
<td>0.97</td>
<td>0.99</td>
<td>1e9</td>
<td></td>
<td></td>
<td>50 dB</td>
</tr>
<tr>
<td>FCG</td>
<td>50</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50 dB</td>
</tr>
<tr>
<td>VCG</td>
<td>50</td>
<td></td>
<td></td>
<td>see eqn. (6.23)</td>
<td>( \mu_{max} )(10 )</td>
<td>0.4</td>
<td>0.4</td>
<td>1e2</td>
<td>5</td>
<td>50 dB</td>
</tr>
</tbody>
</table>

The \( \xi \) parameter for the MVSS is large since the data sequences used are 15 bit integer which have been normalized with respect to 32768. The window sizes for the FCG and VCG algorithm are both set to 5 which provides a good performance/complexity trade-off. The minimum step size for the line search portion of the VCG algorithm was chosen as \( \mu_{min} = \frac{\mu_{max}}{10} \). The signal to noise ratio of the desired signal \( d \) is set to 50 dB.

**Simulation #1, Correlated Input:** Figure 6.8 shows the results when the input is coloured by the first order autoregressive process described by equation (6.17). During the first part of the training, the RLS converges quickly owing to its insensitivity to eigenvalue spread. The FCG and VCG algorithms also converge quickly but the VCG is faster than the FCG algorithm. Both the MVSS and NLMS have poor convergence characteristics due to the correlated input data. At iteration 1000, the unknown system is changed and the RLS algorithm has problems tracking due to the forgetting factor \( \lambda \) being close to 1 and only manages to obtain a lower error than the NLMS and MVSS algorithms by iteration 1500. The VCG and FCG algorithms convergence rates after iteration 1000 are almost identical to the initial convergence rate. *The VCG obtains the best convergence rate of all the above algorithms.*
**FIGURE 6.8** Simulation #1 results. Correlated noise input with a sudden change in the unknown system transfer function at iteration 1000.

*Simulation #2, Comparison of FCG and VCG Algorithms Using Different $n_w$: The conditions for this simulation are the same as in simulation #1 (correlated input) with parameters as listed in Table 6.3. The results in Figure 6.9 show the performance of the VCG with a *limited* conjugate direction reuse rate $R$, as compared to the FCG algorithm using $n_w=5$ and $n_w=8$. In this experiment, the NMSE curves were obtained using the parameters listed in Table 6.3 which also indicates the relative complexity.*

**TABLE 6.4** Parameter and complexity comparison for FCG ($n_w=5$ and 8) and VCG algorithm ($n_w=5$, $R=2$).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>#Taps $m$</th>
<th>$\alpha$</th>
<th>$\zeta$</th>
<th>$\Gamma$</th>
<th>$\gamma$</th>
<th>$n_w$</th>
<th>$R$</th>
<th>SNR</th>
<th>Complexity (mults/iter)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCG</td>
<td>50</td>
<td>0.4</td>
<td>0.4</td>
<td>le2</td>
<td>5</td>
<td>2</td>
<td>50 dB</td>
<td>4161</td>
<td></td>
</tr>
<tr>
<td>FCG</td>
<td>50</td>
<td>0.5</td>
<td></td>
<td></td>
<td>5</td>
<td>50 dB</td>
<td>3631</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FCG</td>
<td>50</td>
<td>0.5</td>
<td></td>
<td></td>
<td>8</td>
<td>50 dB</td>
<td>8299</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FIGURE 6.9 Simulation #2 results. Comparison of FCG and VCGR using a limited gradient reuse rate. Correlated noise input with a sudden change in the unknown system transfer function at iteration 1000.

The gradient averaging window \( n_w \) in the FCG algorithm had to be increased to 8 in order to obtain the same tracking convergence performance as the VCG algorithm with \( R=2 \) and \( n_w=5 \). As a result, the complexity of the FCG (\( n_w=8 \)) is approximately 100% greater than the VCG (\( n_w=5 \), \( R=2 \)) for similar performance results using correlated input signals.

**Simulation #3, VCGR (Simplified VCG) performance:** The conditions for this simulation are the same as in Simulation #1 with parameters as listed in Table 6.3. The results in Figure 6.10 show the performance of VCG and VCGR, as compared to the NLMS and FCG algorithms. The convergence curves illustrate that the VCG outperforms all other algorithms. The convergence of the VCGR algorithm (\( n_w=5 \), \( R=P=5 \)) outperform the NLMS algorithm 300 samples after the transfer function change even with a reduced gradient update rate.
A comparison of the relative complexities is shown in Table 6.3.

**TABLE 6.5** Comparative complexity using NLMS, FCG, VCG and VCGR ($n_w=R=P=5$).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>#Taps $m$</th>
<th>$\alpha$</th>
<th>$\xi$</th>
<th>$\Gamma$</th>
<th>$\gamma$</th>
<th>$n_w$</th>
<th>$R$</th>
<th>$P$</th>
<th>SNR</th>
<th>Complexity (mults/iter)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLMS</td>
<td>50</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50 dB</td>
<td>102</td>
</tr>
<tr>
<td>FCG</td>
<td>50</td>
<td>0.5</td>
<td></td>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td>50 dB</td>
<td>3631</td>
</tr>
<tr>
<td>VCG</td>
<td>50</td>
<td>0.4</td>
<td>0.4</td>
<td>1e2</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td></td>
<td>50 dB</td>
<td>5751</td>
</tr>
<tr>
<td>VCGR</td>
<td>50</td>
<td>0.4</td>
<td>0.4</td>
<td>1e2</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td></td>
<td>50 dB</td>
<td>1968</td>
</tr>
<tr>
<td>VCGR</td>
<td>50</td>
<td>0.4</td>
<td>0.4</td>
<td>1e2</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td></td>
<td>50 dB</td>
<td>1201</td>
</tr>
</tbody>
</table>

The complexity of the VCGR ($n_w=5$, $P=5$) for this case is approximately 33% of the FCG. There are some transients during the first few iterations since the initial weight change estimates (which are reused $P$ times) will be inaccurate during this period. The transients are higher for increased $P$ both during initial convergence and when the transfer function is changed at iteration 1000 but will die out as the algorithm converges. The results indicate that depending on the value of $P$, the convergence rate can be tailored to be fast or slow. Increasing the value of $P$ reduces the convergence rate (and complexity) such that it falls somewhere between the FCG and NLMS algorithms.
FIGURE 6.10 Simulation #3 results. Simplified VCG performance results (VCG). Correlated noise input with a sudden change in the unknown system transfer function at iteration 1000.

6.2.8 Application to Acoustic Echo Cancellation

In this section, the VCG algorithm is applied to a real speech signal as recorded in Conference room #1 using HFT #1 at an average volume of 70 dB SPL. A 20 second speech excitation signal\(^1\) was created by concatenating seven repeated speech segments illustrated in Figure 6.11 to generate the complete 20 second segment shown in Figure 6.12. *Room nonstationarities are introduced during the fifth burst (11-14 seconds) by waving a hand quickly approximately 1 foot above the HFT.* It is possible therefore to observe the characteristics of the algorithm with speech and room nonstationarity. A comparison of the NLMS, MVSS, accelerated SFTF, FCG and VCG algorithms

---

\(^1\) The speech signal is obtained from Nortel as a result of analysis of "typical" conversations on telecommunication networks. The speech signal is recorded at 8kHz.
is shown in Figure 6.13. The number of taps used is 1000, which is sufficient to ensure that the TIP/TP ratio is not a limiting factor. The parameters used are listed in Table 6.6.

**TABLE 6.6 Algorithm parameters**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLMS</td>
<td>$N=1000$, $\alpha=1.0$, halting parameter=0.2</td>
</tr>
<tr>
<td>MVSS</td>
<td>$N=1000$, $\xi=0.97$, $\Gamma=0.99$, $\gamma=10^9$, $\mu_{\text{max}}=1.0$, $\mu_{\text{min}}=10^{-5}$</td>
</tr>
<tr>
<td>Accelerated SFTF</td>
<td>$N=1000$, $\lambda=0.9998$, acceleration factor=0.90, soft initialization constant=50, halting parameter=0.2</td>
</tr>
<tr>
<td>VCG</td>
<td>$N=1000$, $\xi=0.4$, $\Gamma=0.4$, $\gamma=10^2$, $n_{w}=3$, halting parameter=0.2</td>
</tr>
<tr>
<td>FCG</td>
<td>$N=1000$, $\alpha=1.0$, $n_{w}=3$</td>
</tr>
</tbody>
</table>

A close-in view of the *initial convergence* of the algorithms is shown in Figure 6.14. The VCG algorithm attains the fastest convergence and achieves 20 dB of ERLE at 0.5 seconds. The FCG algorithm closely follows the VCG convergence but has approximately 1.2 dB poorer ERLE performance.

The results in Figure 6.15 show a close-in view of the *tracking performance* of the algorithms during the speech burst where a nonstationarity in the room response occurs. The average ERLE values calculated between 11.75 and 13.5 seconds are listed in Table 6.7.

**TABLE 6.7 Average ERLE calculated between 11.75-13.5 seconds.**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average ERLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLMS</td>
<td>14.14 dB</td>
</tr>
<tr>
<td>SFTF</td>
<td>16.13 dB</td>
</tr>
<tr>
<td>VCG</td>
<td>15.33 dB</td>
</tr>
<tr>
<td>FCG</td>
<td>16.56 dB</td>
</tr>
<tr>
<td>MVSS</td>
<td>15.28 dB</td>
</tr>
</tbody>
</table>

On average, the FCG algorithm obtains the best tracking performance, however it is only 2.4 dB better than the NLMS algorithm. The VCG algorithm obtains a 1.19 dB improvement over the NLMS algorithm.
FIGURE 6.11 Time series plot of the speech excitation signal.

FIGURE 6.12 Time series plot of the complete excitation signal.
FIGURE 6.13 Results for different algorithms using speech excitation. A nonstationarity occurs between 11.5 and 14 seconds.

FIGURE 6.14 Close up view illustrating initial convergence performance of algorithms.
6.2.9 Discussion

The VCG algorithm has reduced complexity as compared to the regular CG and is slightly more complex than the FCG algorithm depending on the number of iterations performed during the one dimensional line search. In computer simulations, the VCG algorithm has been shown to have better convergence and tracking properties than the FCG in correlated nonstationary environments and can achieve the same performance as the FCG with reduced complexity. When applied to real speech signals, the VCG algorithm is found to have superior convergence compared to the accelerated SFTF algorithm, however, it exhibits poorer tracking ability than the FCG algorithm but is still better than the NLMS algorithm by approximately 1.2 dB. In terms of AEC application, it is found that the VCG and FCG algorithms can offer improvements in initial convergence and tracking ability with real speech inputs compared to standard algorithms like NLMS and SFTF.
6.3 Summary

This chapter has presented two new methods to enhance the convergence rate of adaptive filters. The first training method, presented in Section 6.1, is based on extending the linear fast conjugate gradient method to the nonlinear domain, specifically neural based adaptive filters. This new training method enhances the convergence rate as compared to the conventional backpropagation algorithm by using a gradient averaging window and simplified conjugate gradient computations. Computer simulations show that the convergence rate can be ‘tailored’ depending on the size of the gradient window. The proposed algorithm was then applied to the nonlinear TDNN-FIR structure and trained with experimentally obtained data, consisting of both noise and speech signals which were recorded at high volumes. The experimental results show that the proposed algorithm, when applied to TDNN based adaptive filters, can obtain better ERLE performance than the linear FIR counterpart trained using the accelerated SFTF algorithm with real speech signals. Up to 5 dB improvement in ERLE is obtained at 95 dB SPL. Also, the initial convergence is comparable to that obtained by the accelerated SFTF algorithm.

The second training method, presented in Section 6.2, is intended for, but by no means restricted to linear adaptive filters. The method combines the FCG algorithm with a one dimensional line search, based on the MVSS algorithm. The MVSS algorithm is used to provide a dynamic step size to replace the optimum step size as calculated in the conventional CG algorithm, and hence can be considered as an alternate to other line search methods like Armijo’s rule. A simplified version of the proposed algorithm is also presented which utilizes the concept of gradient reuse to reduce the computational complexity, depending on the reuse rate. Experimental results using real speech signals show that the VCG algorithm is capable of improved initial convergence rate as compared to the SFTF algorithm and has improved tracking ability as compared to the NLMS algorithm.
Chapter 7

Conclusions

7.1 Summary of this Research

The objective of this thesis was the investigation of nonlinear adaptive filter structures and algorithms for identifying cascaded linear and nonlinear systems with specific application to compensating for nonlinear loudspeaker effects in acoustic echo cancellers (AEC's) placed inside handsfree telephones (HFT's). We concentrated our investigations on neural based filters as computationally attractive alternatives to third order Volterra filters and developed several architectures to identify cascaded linear and nonlinear systems such as those encountered in the handsfree telephone domain. We determined that HFT enclosure resonances and vibrations can be a more serious limitation than loudspeaker nonlinearity to the achievement of a high steady-state Echo Return Loss Enhancement (ERLE). By controlling mechanical vibrations and resonances through appropriate design, neural based filters can provide substantial improvements in ERLE. Finally, we concentrated on developing a nonlinear training algorithm that was efficient, in that a perfor-
mance/complexity trade-off could be selected. When applied to real world HFT’s using noise and speech training signals, significant improvements in converged ERLE could be obtained.

We asked ourselves four questions in the Chapter 1:

1. *What sort of limitations do typical nonlinear loudspeakers present to achieving high ERLE values in typical HFT’s?*

2. *What kind of filters are best suited for nonlinear AEC applications and how can we arrive at that conclusion?*

3. *How can an efficient nonlinear structure and training algorithm be designed that is not overly complicated yet provide reasonable improvements in performance?*

4. *Can the new structures/algorithms be successfully applied in real-world applications?*

A review of nonlinear loudspeaker dynamics and performance in Chapter 2 indicates the following:

- The major cause of distortion in loudspeakers is due to nonuniform flux density and two point suspension nonlinearity which become predominant at low frequencies and high volumes. Two point suspension nonlinearity is present also at extremely low volumes.

- The Volterra filter has found some success in modelling low frequency (woofer) loudspeakers for high quality studio applications. A 3rd order filter is usually necessary to reduce nonlinear distortion to target values of -30 dB.

- No applications in the literature could be found dealing with mid-frequency, low quality loudspeakers that are typically used in HFT’s.

In Chapter 4, we looked at the acoustic echo cancellation problem and determined the following:
7.1 Summary of this Research

- Although high values of ERLE are often stipulated in specifications\(^1\), there is little information on how nonlinear loudspeaker performance impacts on the practical achievement of these values.

- Transducer nonlinearities limit the achievable ERLE at high and extremely low volumes. Transducer quality also plays a role in determining the achievable ERLE value.

- Vibration and key rattling is a limitation. It is shown in Chapter 5 that this limitation can prevent nonlinear algorithms from achieving their full potential if not controlled adequately.

*Our first question is now answered.* However, achieving 40 dB of echo cancellation even in the linear range of the loudspeaker appears to be possible only when *separated* loudspeaker and microphone components are used in anechoic conditions. When placed inside a typical HFT enclosure, vibrations, rattling and other effects will serve to limit the achievable ERLE to below 35 dB.

In Chapter 4 a subsection on the application of infinite impulse response (IIR) structures for AEC’s arrived at the following conclusions:

- Experimental results performed on both separate transducers in an anechoic chamber and HFT’s in furnished conference rooms suggest that IIR structures are not well suited to modelling a typical loudspeaker-room-enclosure-microphone (LREM) even though *some* literature suggests otherwise.

- A Hankel norm approximation error bound for IIR filters is shown to have similar characteristics to the Total Impulse Power to Tail Power (TIP/TP) ratio at low filter orders.

---

1. Typical specifications are at least 30 dB of cancellation in 0.1 seconds.
This partially answers our second question, namely, feedforward structures should be considered as more promising candidates for AEC applications than recursive structures.

In Chapter 5, Volterra and neural filters were examined as possible candidates for a nonlinear AEC. The following conclusions can be summarized:

- Simulation results show that the Volterra filter can obtain a higher modelling accuracy than a neural based filter. However, experimental results show that when presented with real-world signals, neural filters can achieve equal or better results compared to the Volterra filter.

- For AEC applications where the model order is quite large Volterra filters have a much higher complexity than neural based filters for equivalent performance results.

- A tapped delay line neural network (TDNN) filter can achieve better performance than a linear finite impulse response (FIR) filter only when the nonlinear distortion is greater than several percent of the primary signal power. Given that some improvement can be made with a neural filter, a low number of hidden nodes is sufficient to provide several dB of ERLE improvement.

- In the undermodelled case, the TDNN provides a significant improvement in ERLE compared to the linear FIR which in the linear region is fundamentally limited by the TIP/TP ratio.

- The TDNN was found to have poorer performance at low distortion levels compared to the linear FIR filter. A linear region in the activation function was found to improve the modelling accuracy at both low and high distortion levels compared to networks using hyperbolic tangent function activation functions.

We are now in a position to completely answer the second question, namely that simple, feedforward neural based filters can achieve the best performance complexity trade-off compared to Volterra algorithms and recursive IIR structures.
Since simplicity of design is of utmost importance in AEC design, several new neural filters were subsequently proposed, the most successful being the two stage neural filter. The two stage neural filter has the following features:

- The neural filter is composed of a TDNN in parallel with an FIR filter. The TDNN portion models the first part of the LREM where most of the signal energy is contained. The FIR portion models the remaining echo tail.

- A low order TDNN with a single hidden layer and one or two hidden nodes is sufficient to model nonlinearities typically encountered in the AEC domain. Experimental results show that the converged steady-state ERLE of a real world HFT can be improved by up to 11 dB when trained with a filtered noise signal.

The variable activation function was proposed which adapts to be highly linear when trained with signals which have little nonlinear distortion. A training algorithm was developed.

As an alternative to weakened nonlinear systems which consist of fixed nonlinearities sandwiched between linear adaptive filters, the synaptic FIR multilayer perceptron (MLP) with variable activation function was proposed (VA-FIR MLP), along with an appropriate backpropagation (BP) based training method.

- Computer simulations show that the VA-FIR MLP is well suited to identifying low order nonlinear systems of the sandwiched linear-nonlinear-linear type. For application to nonlinear AEC, experimental results show that it is not as effective as the two stage neural filter described earlier.
7.1 Summary of this Research

We can now answer the first part of Question 3). By mixing neural networks and linear FIR structures, a two stage neural filter has been constructed which maintains the benefits of both filter types and keeps the overall architectural complexity low.

In order to study the performance of the structures proposed in Chapter 5, a nonlinear fast conjugate gradient (NFCG) backpropagation algorithm is developed in Chapter 6. The NFCG algorithm has the following features:

- The NFCG is based on the standard conjugate gradient (CG) algorithm. It is developed by extending the fast conjugate gradient (FCG) algorithm to the nonlinear case, specifically neural networks.

- There is a complexity/performance trade-off which is determined by the size of the gradient averaging window.

We can now answer the last part of Question 3). A simple algorithm based on the CG method can be used to train the nonlinear network and provide a complexity/performance trade-off.

When applied to the two stage neural filter of Chapter 5 using real speech inputs at an average volume of 95 dB SPL, the NFCG is capable of improving the ERLE by approximately 5 dB as compared to an FIR trained with the accelerated Stabilized Fast Transversal Filter (SFTF) algorithm.

A linear variation on the FCG algorithm is also developed which uses a dynamic step size calculation using the modified variable step size (MVSS) algorithm. Experimental results using speech signals in a conference room environment show that the variable step size CG (VCG) algorithm achieves the fastest convergence rate of several algorithms tested, including the FCG and accelerated SFTF algorithm.
Finally, Question 4) is answered. The proposed architectures and algorithms have been successfully applied to real world signal processing applications.

7.2 Summary of Contributions

The investigation, development and subsequent simulation and experimental performance results presented in this thesis provide several important contributions to the field of neural networks, acoustic echo cancellation, and nonlinear system identification. These contributions are briefly summarized here:

1. Vibration and rattling within the HFT handset can present a physical limitation to achievable ERLE which may or may not be more severe than nonlinear loudspeaker distortion. It must be controlled in order to allow nonlinear speech and echo cancellation algorithms/structures to work effectively.

2. A novel two stage neural filter has been developed. It has a simple architecture that requires only a small number of nonlinear nodes. It is capable of providing up to 11 dB ERLE improvement at high volumes using training signals consisting of noise and up to 5 dB improvement when using real speech signals.

3. The development of a unique variable activation function and derivation of the update equations for incorporation into a standard TDNN or synaptic FIR MLP has been presented. The update equations are based on the temporal backpropagation algorithm with modifications to allow for a window of accumulated gradients.

4. A new fast conjugate gradient algorithm for neural networks has been developed and applied to the TDNN and two stage neural filter. The update equations are based on the fast conjugate gradient algorithm which has been modified for application to neural networks. A linear FCG
algorithm that incorporates MVSS line search and gradient reuse to obtain complexity reductions is also presented which obtains substantial convergence rate improvements with real speech signals.

5. Publication of several refereed conference and journal papers which report on the research results. See [52],[127],[128],[164],[165],[166],[167],[168],[169],[170],[171].

7.3 Suggestions for Future Research

During the course of this research, several issues arose which merit further research. These are summarized below:

• Combining frequency domain and nonlinear methods. The GMDF [138] algorithm is a successful AEC algorithm that might benefit from the addition of nonlinear signal processing of the form described in this thesis. The GMDF algorithm is capable of achieving high levels of converged ERLE with speech inputs\(^1\), however it would appear that further improvements may not be possible without the inclusion of some sort of nonlinear signal processing.

• Investigation of methods to model and reduce vibrations and resonances within a handsfree terminal, including proper selection, orientation and mounting of both the loudspeaker and microphone elements. This would improve the achievable steady state ERLE and perceived speech quality.

• Experimental determination of the typical parameters of small inexpensive loudspeakers typically used in HFT's using a laser displacement system and determination of how closely these fit the theoretical models. The nonlinear modelling presented in Chapter 2 assumed that the

---

\(^1\) Recent measurements performed by J.P. Lariviere [172] have produced real-world ERLE values of 35-40 dB using speech inputs recorded at mid volumes in the 70 dB SPL range.
inductance \( L(x) = L_0 \) and the higher order terms \( L_1 x + L_2 x^2 \) were negligible. This may be an invalid assumption for these types of loudspeakers.

- Application of the conjugate gradient training algorithms in Chapter 6 to the structures which contain variable activation functions.

### 7.4 Conclusion

In conclusion, this thesis has presented the key highlights in our study on the use of nonlinear signal processing techniques to the field of acoustic echo control. The research results have given positive, definitive answers to the four questions posed previously.

In an ideal world, all problems would be isolated from one another, and the solution procedures would involve direct methods, based on the principle of divide and conquer. However, in the real world, every piece of matter is immutably connected with every other piece of matter in some way, either directly or vicariously, and what may appear to be an obvious solution to a problem at first glance is often cursory. We are then presented with the opportunity to solve a small part of a much larger puzzle. Indeed, the puzzle is limitless.
References


Conference Room #1
Minto 3033

FIGURE A.1 Conference room #1 (Minto 3033).

Characteristics:

- rectangular: 12'W x 18'L x 10'H
- reflective walls and floor.
- two tables, 6 padded chairs arranged around one table.
- ventilation fan on
Conference Room #2
Minto 2014

FIGURE A.2 Conference room #2 (Minto 2014)

Characteristics:

- rectangular conference room: 19'W x 38' L x 10'H
- furnished, padded chairs, paintings, one small sound absorbing tile.
- floor is carpeted.
- ventilation fan on.
Standard Audio Baffle

(Anechoic Recordings)

FIGURE A.3 Standard baffle used to test stand-alone loudspeakers. (a) Dimensions of plywood section (b) Detail of plexiglass submount.

Notes:

- The loudspeaker is placed in standard baffle to remove effects of vibrations and resonances within the HFT enclosure.
- Used during anechoic recordings.
- The large size approximates an "infinite baffle" characteristic.
- Dimensions are offset to prevent vibrational modes within the baffle.
## Appendix B

### Equipment Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Model</th>
<th>Rating</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPK#1</td>
<td>CF050-A00604</td>
<td>50Ω, 2W</td>
<td>2&quot; dia. round, large magnet.</td>
</tr>
<tr>
<td>SPK#2</td>
<td>LS06C050</td>
<td>50Ω, 0.5W</td>
<td>2.25&quot; x 3&quot; oval. Medium size magnet.</td>
</tr>
<tr>
<td>SPK#3</td>
<td>AD4061/W8</td>
<td>8Ω, 10W</td>
<td>4&quot; dia. round. Large magnet, high quality.</td>
</tr>
<tr>
<td>MIC#1</td>
<td>Archer 270-090</td>
<td>4.5VDC thru ext. 1kΩ S/N &gt;40 dB, Sens.=6.5 dB</td>
<td>Electret Microphone (low quality).</td>
</tr>
<tr>
<td>MIC#2</td>
<td>Audio Technica AT831b and power module</td>
<td>Cardiod sen. -44 dBm 200Ω.</td>
<td>”High quality” microphone element.</td>
</tr>
<tr>
<td>HFT#1</td>
<td>Northern Telecom / NT8B04AA</td>
<td>SPK#2</td>
<td>$75 price range</td>
</tr>
<tr>
<td>HFT#2</td>
<td>Mitel/Superset 430</td>
<td>60Ω, 0.5W, 2.5&quot; round spkr.</td>
<td>Office set functions</td>
</tr>
<tr>
<td>HFT #3</td>
<td>Mitel/Superset 410</td>
<td>60Ω, 0.5W, 2.5&quot; round spkr.</td>
<td>Office set functions</td>
</tr>
<tr>
<td>HFT#4</td>
<td>Telemax / CP268A</td>
<td>8Ω, 0.25W, 2&quot; round spkr.</td>
<td>$30 price range</td>
</tr>
<tr>
<td>HFT#5</td>
<td>Panasonic / KX-T2315</td>
<td>32Ω, 0.4W, 2.5&quot; round spkr.</td>
<td>$120 price range</td>
</tr>
<tr>
<td>HFT#6</td>
<td>Northern Telecomm / Vista 350</td>
<td>50Ω, 0.5W, 2&quot; round spkr.</td>
<td>Electret mic. inside a rubber grommet. Loudspeaker not screwed to enclosure.</td>
</tr>
<tr>
<td>SPL Meter</td>
<td>Realistic 33-2050</td>
<td>50-125 dB SPL</td>
<td>0dB=0.0002 µbar.</td>
</tr>
<tr>
<td>DAT #1</td>
<td>TEAC DA-P20</td>
<td>Portable</td>
<td></td>
</tr>
<tr>
<td>DAT #2</td>
<td>Sony TC-D7</td>
<td>Portable</td>
<td></td>
</tr>
<tr>
<td>Noise Generator</td>
<td>General Radio Company 1390-B</td>
<td>0.0005-5 Vrms range and 20 kHz/500kHz/5Mhz selectable BW.</td>
<td>Noise diode, amplified.</td>
</tr>
<tr>
<td>Audio Amplifier</td>
<td>Samson Servo-150</td>
<td>75W rms/channel. 2 channels.</td>
<td>Studio quality.</td>
</tr>
</tbody>
</table>
Appendix C  Circuit Schematics

A number of circuits were required in order to make the appropriate measurements. The schematics are shown in Sections C.1 through C.7.

C.1 Primary Conditioning Amplifier

AD524: Instrumentation Amplifier.
C.2 Reference Conditioning Amplifier

AD524: Instrumentation Amplifier.
C.3 Switched Capacitor Filter

U1: 74HC04
XTAL: MP-1-4.096 MHz

Lowpass Gain = 1.0  Lowpass Frequency: DC-3400 Hz
Bandpass Gain = 1.0  Bandpass Frequency: 200-3400 Hz
C.4 Resistive Attenuator Circuits

Output impedance approximates microphone loading.

XLR type socket
C.5 General Purpose Amplifiers


**Appendix D**

**Algorithms**

### D.1 The LMS Algorithm

**Initialization.**

\[ w(n) = 0 \]  \hspace{1cm} (D.1)

**Algorithm.**

\[ e(n) = y(n) - w(n)^T \cdot x(n) \]  \hspace{1cm} (D.2)

\[ w(n+1) = w(n) + \mu \cdot e(n) \cdot x(n) \]  \hspace{1cm} (D.3)

**Stability.** \( \mu \) is the step size which must be chosen to ensure stability:

\[ 0 < \mu < \frac{2}{\sum_{i=0}^{M-1} \lambda_i} \]  \hspace{1cm} (D.4)

where \( \lambda_i \) are the eigenvalues of the input correlation matrix.

### D.2 The Normalized LMS Algorithm

**Initialization.**

\[ w(n) = 0 \]  \hspace{1cm} (D.5)

**Algorithm.**

\[ e(n) = y(n) - w(n)^T \cdot x(n) \]  \hspace{1cm} (D.6)

\[ w(n+1) = w(n) + \frac{\alpha \cdot e(n) \cdot x(n)}{\varepsilon + \|x(n)\|^2} \]  \hspace{1cm} (D.7)

where

- \( \alpha \) is the normalized step size constant.
- \( \varepsilon \) is a small positive constant used to place a lower bound on the input signal power.

**Stability.** \( 0 < \alpha < 2 \).
D.3 The Modified Variable Step Size Algorithm

The MVSS is a robust variable step size algorithm for LMS type filters that estimates the autocorrelation of the error signal to determine when the minimum of the performance surface is reached. When the error correlation is large, dynamic step size control adjusts the step size to be large thus speeding up the convergence. When the error correlation is small, as is the case in high noise environments, or when the minimum is reached, the step size is reduced correspondingly.

_Initialization._

\[ w(n) = 0 \]  \hspace{1cm} (D.8)

_Algorithm._

\[ e(n) = y(n) - w(n)^T \cdot x(n) \]  \hspace{1cm} (D.9)

\[ w(n + 1) = w(n) + \mu(n) \cdot e(n) \cdot x(n) \]  \hspace{1cm} (D.10)

\[ \mu(n + 1) = \begin{cases} 
\mu_{max} & ; \mu(n + 1) \geq \mu_{max} \\
\mu_{min} & ; \mu(n + 1) \leq \mu_{min} \\
\zeta \mu(n) + \gamma \rho^2(n) & ; \mu_{min} < \mu(n + 1) < \mu_{max}
\end{cases} \]  \hspace{1cm} (D.11)

where,

\[ \rho(n) = \Gamma \rho(n - 1) + (1 - \Gamma) e(n) e(n - 1) \]  \hspace{1cm} (D.12)

\[ \begin{cases} 
0 < \zeta < 1 \\
\Gamma < 1 \\
\gamma > 0
\end{cases} \]  \hspace{1cm} (D.13)

The parameter \( \gamma \) controls the convergence time as well as the final misadjustment. The parameter \( \zeta \) controls the averaging of the step size update and \( \Gamma \) controls the averaging time constant of the filtered error update. The parameter \( \rho \) gives a short time estimate of the error signal autocorrelation. Typical parameter values are \( \zeta = 0.97 \), \( \Gamma = 0.99 \), \( \gamma = 1e-5 \) [56].
D.4 The Exponentially Weighted RLS Algorithm

The exponentially weighted RLS algorithm can be constructed from the *accelerated steepest descent algorithm*.

\[
\varepsilon(n) = y(n) - w^T(n-1)x(n) \\
w(n) = w(n-1) + \mu(n) \varepsilon(n) P(n-1)x(n)
\]  
(D.14)

by setting the variables \(P(n-1)\) and \(\mu(n)\) as follows:

\[
P(n-1) = R^{-1}(n-1) \\
\mu(n) = \frac{1}{1 + q(n)}
\]  
(D.15)

where:

\[
q(n) = \lambda^{-1} x^T(n) P(n-1) x(n)
\]  
(D.16)

\(P(n-1)\) is the estimated inverse of the \(M\) by \(M\) autocorrelation matrix \(R\) at time \(n-1\).

\(\lambda\) is a forgetting factor between 0 and 1.

\(\varepsilon\) is the a priori estimation error based on the weight vector at time \(n-1\).

The term \(q\) is a measure of the input signal power just as \(x^T(n)x(n)\) would be, but with a normalization introduced by \(P(n-1)\). The *matrix inversion lemma* [173] is employed to recursively compute \(P(n)\). It reduces the complexity from \(O(M^3)\) to \(O(M^2)\) by using the previous value \(P(n-1)\) as follows:

\[
P(n) = \lambda^{-1} P(n-1) - \lambda^{-1} k(n) x^T(n) P(n-1)
\]  
(D.17)

where the gain vector \(k(n)\) is described by:
\[ k(n) = \frac{\lambda^{-1} P(n-1) x(n)}{1 + q(n)} \] (D.18)

### D.5 The Accelerated 8N-SFTF Algorithm

**Initialization.**

\[
\alpha(0) = \zeta \cdot \lambda^N
\]

\[
\beta(0) = \zeta
\]

\(\zeta\) is the initial input variance and should be set large enough to prevent start-up divergence. A value between 1 and 200 appears reasonable for filter orders up to 1200 taps.

\[
\gamma_N(0) = 1
\]

\[
x_N = a_N = b_N = k_N = w_N = 0
\]

\[
k_{N+1} = 0
\] (D.19)

**Phase 1: Prediction Part.**

- Compute the forward prediction error \(e_f(n)\).

\[
e_f(n) = x(n) - a_N(n-1)^T \cdot x_N(n-1)
\] (D.20)

- Compute the variance \(\alpha\) of the forward prediction error.

\[
\alpha(n) = \lambda \cdot \alpha(n-1) + \gamma_N(n-1) \cdot e_f(n)^2
\] (D.21)

- Compute the likelihood variable \(\gamma\) of order \(N+1\).

\[
\gamma_{N+1}(n) = \frac{\lambda \cdot \alpha(n-1)}{\alpha(n)} \cdot \gamma_N(n-1)
\] (D.22)

- Compute the first auxiliary variable \(s\) and the dual Kalman gain \(k\) of order \(N+1\).
\[ s_{N+1}(n) = \frac{e_f(n)}{\lambda \cdot \alpha(n-1)} \cdot \begin{bmatrix} 1 \\ -a_N(n-1) \end{bmatrix} \] (D.23)

\[ k_{N+1}(n) = \begin{bmatrix} 0 \\ \hat{k}_N(n-1) \end{bmatrix} - s_{N+1}(n) \] (D.24)

- Update the coefficients of the forward predictor \( a_N \).
  \[ a_N(n) = a_N(n-1) - e_f(n) \cdot \gamma_N(n-1) \cdot k_N(n-1) \] (D.25)

- Compute the backward prediction error \( e_b(n) \).
  \[ e_b(n) = x(n-N) - b_N(n-1)^T \cdot x_N(n) \] (D.26)

- Stabilize the backward prediction error.
  \[ \tilde{e}_b(n) = 2 \cdot e_b(n) + \lambda \cdot \beta(n-1) \cdot k_{N+1}^{N+1}(n) \] (D.27)

- Compute the likelihood variable of order \( N \).
  \[ \gamma_N(n) = \frac{\gamma_{N+1}(n)}{1 + \gamma_{N+1}(n) \cdot \tilde{e}_b(n) \cdot k_{N+1}^{N+1}(n)} \] (D.28)

- Compute the second auxiliary variable \( u \) and the dual Kalman gain \( k \) of order \( N \).
  \[ u_{M+1}(n) = -k_{N+1}^{N+1}(n) \cdot \begin{bmatrix} -b_N(n-1) \\ 1 \end{bmatrix} \] (D.29)

\[ \begin{bmatrix} k_N(n) \\ 0 \end{bmatrix} = k_{N+1}(n) + u_{M+1}(n) \] (D.30)

- Update the coefficients of the backward predictor \( b \).
  \[ b_N(n) = b_N(n-1) - \tilde{e}_b(n) \cdot \gamma_N(n) \cdot k_N(n) \] (D.31)

- Compute the variance \( \beta \) of the backward prediction error.
\[ \beta(n) = \lambda \cdot \beta(n-1) + \gamma_N(n) \cdot \varepsilon_b^2(n) \quad (D.32) \]

**Phase 2: Filtering Part.**

\[ e(n) = y(n) - w_N(n-1) \cdot x_N(n) \quad (D.33) \]

\[ w_N(n) = w_N(n-1) - \eta(n) \cdot e(n) \cdot \gamma_N(n) \cdot k_N(n) \quad (D.34) \]

where

\[ \eta(n) = \frac{1}{1 - \rho \cdot \gamma_N(n)} \quad (D.35) \]

and \( \rho \) is the acceleration factor which varies the effective accelerated forgetting factor \( \lambda_{acc} \) between \( \lambda (\rho=0) \) and \( 0 (\rho=1) \), according to:

\[ \lambda_{acc} = \frac{(1 - \rho) \lambda}{1 - \rho \lambda} \quad (D.36) \]

**Stability.** \( \left( 1 - \frac{1}{2N} \right) < \lambda < 1 \)  (Note: the lower bound should be avoided)

**Conditional Re-initialization.** The prediction part of the algorithm must be re-initialized following periods of poor excitation, which is typical with speech. This is accomplished by resetting all prediction variables whenever \( \gamma_N(n) \) approaches zero without clearing \( w_N(n) \), i.e.,

\[ a_N = b_N = k_N = 0 \quad (D.37) \]

\[ \alpha(n) = \zeta \cdot \lambda^N \quad \beta(0) = \zeta \quad (D.38) \]
D.6 Equation Error LMS-IIR Algorithm

Initialization. \[ w(n) = 0 \] (D.39)

Algorithm.

\[ y(n) = w^T(n)u(n) \] (D.40)
\[ e(n) = d(n) - y(n) \] (D.41)
\[ w(n+1) = w(n) + \mu e(n)u(n) \] (D.42)

where:

\[ w(n) = [a_1(n), a_2(n), \ldots, a_{n_a}(n), b_0(n), b_1(n), \ldots, b_{n_b}(n)]^T \] (D.43)
\[ u(n) = [d(n-1), \ldots, d(n-n_{a}), x(n), x(n-1), \ldots, x(n-n_{b})]^T \] (D.44)
\[ \mu = [\mu^a_1, \ldots, \mu^a_{n_a}, \mu^b_0, \ldots, \mu^b_{n_b}] \] (D.45)

The step sizes \( \mu^a \) and \( \mu^b \) can be fixed or normalized with respect to the input power in the same way as the NLMS algorithm.

D.7 Output Error LMS-IIR Algorithm with Simplified Gradient

The output of an OE-IIR filter is defined by:

\[ y(n) = \sum_{i=1}^{n_a} a_i(n) y(n-i) + \sum_{i=0}^{n_b} b_i(n) x(n-i) \] (D.46)

The parameter update is obtained by minimizing \( J \) with respect to the weight vector \( w \):
\[ \nabla_w (J) = \frac{\partial}{\partial w} (e^2 (n)) = 2e(n) \frac{\partial}{\partial w} [d(n) - y(n)] \]
\[ = -2e(n) \frac{\partial}{\partial w} [y(n)] = -2e(n) \nabla_w (y(n)) \quad (D.47) \]

where the gradient \( \nabla_w (y(n)) \) is the following column vector:
\[ \nabla_w (y(n)) = \left[ \frac{\partial y(n)}{\partial a_1(n)}, \ldots, \frac{\partial y(n)}{\partial a_{n_a}(n)}, \frac{\partial y(n)}{\partial b_0(n)}, \ldots, \frac{\partial y(n)}{\partial b_{n_b}(n)} \right]^T \quad (D.48) \]

Each component \( \nabla_w (y(n)) \) may be obtained by differentiating (D.46) to obtain:
\[ \frac{\partial y(n)}{\partial a_i(n)} = y(n-i) + \sum_{m=1}^{n_a} a_m(n) \frac{\partial y(n-m)}{\partial a_i(n)} = y(n-i) + \sum_{m=1}^{n_a} a_m(n) \frac{\partial y(n-m)}{\partial a_i(n-m)} \quad (D.49) \]
\[ \frac{\partial y(n)}{\partial b_i(n)} = x(n-i) + \sum_{m=1}^{n_a} a_m(n) \frac{\partial y(n-m)}{\partial b_i(n)} = x(n-i) + \sum_{m=1}^{n_a} a_m(n) \frac{\partial y(n-m)}{\partial b_i(n-m)} \quad (D.50) \]

The approximation is valid if the step size is chosen sufficiently small [63]. We may now rewrite (D.48) using the approximations of (D.49) and (D.50) as a purely “autoregressively” filtered regressor:
\[ u_f(n) = u(n) + \sum_{m=1}^{n_a} a_m(n) u_f(n-i) \quad (D.51) \]

\( u_f(n) \) can be viewed as an approximate gradient estimate of the current output \( y(n) \) with respect to the weight vector \( w \). The calculation of \( u_f(n) \) is numerically intensive due to the AR filtering by the \( a_i \) coefficients for each component. A simplified gradient calculation can be made by assuming that the filtered regressor vector can be approximated by taking delayed outputs of the
first filtered component only. As a result, the filtered regressor becomes:

$$u_f(n) = [y_f(n-1), y_f(n-2) \ldots y_f(n-n_a), x_f(n), x_f(n-1), \ldots x_f(n-n_b)]^T$$ (D.52)

Figure D.1 shows how the simplified gradient vector is obtained by filtering the $x(n)$ and $y(n)$ by the all pole section $1 - A(z)$, and then selecting delayed versions of $x(n)$ and $y(n)$.

![Diagram](image)

**FIGURE D.1 The OE simplified gradient evaluation.**

**Stability Monitoring.** One of the drawbacks with IIR algorithms is that the poles may update outside of the unit circle and cause instability. Various tests may be performed to ensure that the updates are stable, but this tends to either add complexity or compromise the performance of the algorithm. If the sum of the magnitude of all the pole coefficients is less than zero, then stability is guaranteed [63] however, it severely limits the values of $a_i(n)$ especially for large $n_a$.

$$\sum_{i=1}^{n_a} |a_i(n)| < 1, \quad \text{for all } n$$ (4.53)

Stability monitoring may be simplified by using parallel or cascaded structures which consist of
1st or 2nd order IIR sections [61] which have a simpler stability check, referred to as the stability triangle [63]. Lattice structures also have simple stability checks (see [60]).

Combining (D.46) through (D.52) yields the adaptive IIR-LMS filter algorithm with the same form as the accelerated steepest descent algorithm of equation (D.14).

---

**OE-IIR Algorithm with Simplified Gradient**

**Initialization.**

\[ w(n) = 0 \]  \hspace{1cm} (D.54)

**Vector Definitions.**

\[ w(n) = [a_1(n), a_2(n) \ldots a_{n_s}(n), b_0(n), b_1(n) \ldots b_{n_p}(n)]^T \]  \hspace{1cm} (D.55)

\[ u(n) = [y(n-1), \ldots, y(n-n_a), x(n), x(n-1), \ldots, x(n-n_b)]^T \]  \hspace{1cm} (D.56)

\[ y_f(n) = y(n) + \sum_{m=1}^{n_s} a_m(n) y_f(n-m) \]  \hspace{1cm} (D.57)

\[ x_f(n) = x(n) + \sum_{m=1}^{n_s} a_m(n) x_f(n-m) \]  \hspace{1cm} (D.58)

\[ u_f(n) = [y_f(n-1), y_f(n-2) \ldots y_f(n-n_a), x_f(n), x_f(n-1), \ldots x_f(n-n_b)]^T \]  \hspace{1cm} (D.59)

**Algorithm.**

\[ e(n) = d(n) - w^T(n) u(n) \]  \hspace{1cm} (D.60)

\[ w(n+1) = w(n) + \mathbf{M} \mathbf{P}(n) u_f(n) e(n) \]  \hspace{1cm} (D.61)
The matrix $P(n)$ is replaced with the identity matrix $I$ in the same manner as $P(n)$ is replaced with the identity matrix in the LMS algorithm, and:

$$M = \text{diag}\left[\mu_1, \ldots, \mu_n, \rho_0, \ldots, \rho_n\right]. \quad (D.62)$$

represents the fixed step sizes. Alternately, if we wish to solve for the coefficients in a least-squares sense, we may replace $P(n)$ with an estimate of the inverse Hessian matrix, updated according to:

$$P^{-1}(n) = \lambda P^{-1}(n-1) + (1 - \lambda) u_f(n-1) u_f^T(n-1) \quad (D.63)$$

and $M$ with a fixed step size $\mu$. Alternately, $P$ may be update using the matrix inversion lemma [173]:

$$P(n) = \frac{1}{\lambda} \left[ P(n-1) - \frac{P(n-1) u_f(n) u_f^T(n) P(n-1)}{\lambda + u_f^T(n) P(n-1) u_f(n)} \right] \quad (D.64)$$

D.8 The LMS Volterra Algorithm

*Mapping and Regressor Construction.* For a $p^{th}$ order polynomial system, construct the $p^{th}$ order weight and input vectors.

$$h_p(n) = [h_p(m_1, m_1, \ldots, m_1), h_p(m_1, m_1, \ldots, m_2), \ldots, h_p(m_p, m_p, \ldots, m_p)]^T \quad (D.65)$$

$$x_p(n) = x_1(n) \otimes x_{p-1}(n) \quad (D.66)$$

where $\otimes$ is the Kronecker product of vectors and it is assumed that the duplicate terms have been removed. Next construct the extended weight and input vectors.
\[ h_e^T(n) = [h_0, h_1^T(n), \ldots, h_p^T] \] (D.67)

\[ x_e^T(n) = [1, x_1^T(n), \ldots, x_p^T] \] (D.68)

**Initialization.**

\[ h_e(n) = 0 \] (D.69)

**Algorithm.**

\[ e(n) = d(n) - h_e^T(n) \cdot x_e(n) \] (D.70)

For each \( p \)th order polynomial section update the weights.

\[ h_p(n + 1) = h_p(n) + \mu_p e(n) x_p(n) \] (D.71)

where \( \mu_p \) is the step size for the \( p \)th power term. Alternately, a normalized step size may be computed for each \( p \)th order power term as:

\[ \hat{\mu}_p = \frac{\alpha}{\varepsilon + \|x_p(n)\|^2} \] (D.72)

**Stability.** Each \( \mu_p \) is chosen to ensure stability according to:

\[ 0 < \mu_p < \frac{2}{\sum_{i=0}^{M-1} \lambda_i} \] (D.73)

where \( \lambda_i \) are the eigenvalues of the \( p \)th extended input correlation matrix.
Appendix E  Statistics of a Nonlinear Function

E.1 Mean and Variance

**Expected Value:** The expected value or *mean* of a continuous random variable $X$ having a probability density function $f(x)$ is given by:

$$E(X) = m = \int_{-\infty}^{\infty} xf(x) \, dx$$  \hspace{1cm} (E.1)

If $X$ is a continuous random variable with probability distribution $f(x)$ and $g(x)$ is any real valued function of $X$, then:

$$E(g(X)) = \int_{-\infty}^{\infty} g(x)f(x) \, dx$$  \hspace{1cm} (E.2)

**Variance:** if $X$ is a random variable with mean $m$, then the *variance* equals:

$$V(X) = E(X^2) - m^2$$  \hspace{1cm} (E.3)

For any random variable $X$ and constants $a$ and $b$:

$$E(aX + b) = aE(X) + b$$  \hspace{1cm} (E.4)

$$V(aX + b) = a^2 V(X)$$  \hspace{1cm} (E.5)

E.2 The Normal and Uniform Distribution

**Normal Distribution:** The normal distribution probability density function is given by:

$$f(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-x^2/2\sigma^2}$$  \hspace{1cm} (E.6)

where $\sigma$ is the standard deviation of the signal, usually set to one.
**Higher Order Moments:** The higher order moments may be obtained from the following equation.

\[
E(X^n) = \begin{cases} 
0 & n = 2k + 1 \\
1 \cdot 3 \ldots (n - 1) \sigma^n & n = 2k
\end{cases} \tag{E.7}
\]

**Variance of The Normal Distribution:** The variance of the normal distribution is obtained by substituting (E.7) into (E.3) to obtain:

\[
V(X) = \sigma^2 \tag{E.8}
\]

**The Uniform Distribution:** The uniform probability density function is given by:

\[
f(x) = \frac{1}{(b - a)} \quad a \leq x \leq b \\
f(x) = 0 \quad \text{elsewhere} \tag{E.9}
\]

**Mean of the Uniform Distribution:** Using the above probability function and equation (E.2) we can find \(E(X)\) as:

\[
E(X) = \frac{b + a}{2} \tag{E.10}
\]

**Higher Order Moments:** Similarly, substituting \(X^2, X^3, X^4, X^5, X^6\) into (E.2), we can get:

\[
E(X^2) = \frac{b^2 + ab + a^2}{3} \tag{E.11}
\]

\[
E(X^3) = \frac{b^3 + b^2a + ba^2 + a^3}{4} \tag{E.12}
\]

\[
E(X^4) = \frac{b^4 + b^3a + b^2a^2 + ba^3 + a^4}{5} \tag{E.13}
\]

\[
E(X^5) = \frac{b^5 + b^4a + b^3a^2 + b^2a^3 + ba^4 + a^5}{6} \tag{E.14}
\]
\[ E(X^6) = \frac{b^6 + ab^5 + a^2b^4 + a^3b^3 + a^4b^2 + a^5b + a^6}{7} \]  
(E.15)

**Variance of The Uniform Distribution:** The variance of a zero mean variable \( X \) with a uniform distribution between \( a \) and \( b \) can be computed by substituting (E.10) and (E.11) into (E.3) to obtain:

\[ V(X) = \frac{(b - a)^2}{12} \]  
(E.16)

### E.3 Mean and Variance of a Nonlinear Function

If \( y \) is a function of a random variable \( X \), i.e.:

\[ y = uX + vX^2 + wX^3 \]  
(E.17)

then the *mean* of the function \( y \) is given by:

\[ E(y) = uE(X) + vE(X^2) + wE(X^3) \]  
(E.18)

and the *variance* of the function \( y \) is given by the relationship:

\[ V(y) = E(y^2) - [E(y)]^2 \]  
(E.19)

where \( E(y^2) \) is given by the following equation:

\[ E(y^2) = u^2E(X^2) + 2uvE(X^3) + (v^2 + 2uw)E(X^4) + 2vwE(X^5) + w^2E(X^6) \]  
(E.20)

### E.4 Nonlinear Examples

**Example 1: Signal-to-Distortion Ratio (SDR) Assuming a Uniform Distribution for \( X \)**

Assume \( X \) has a range between -1 and +1. Calculate the signal to distortion of a variable \( y \) where \( y \) is a function of \( X \), represented by the following equation:
Statistics of a Nonlinear Function

\[ y = \frac{\alpha X + \beta X^2 + \delta X^3}{|\alpha| + |\beta| + |\delta|} \]  
(E.21)

This may be expressed as:

\[ y = s + d \]
\[ = uX + vX^2 + wX^3 \]  
(E.22)

where we have divided \( y \) into signal \( s \) and distortion \( d \) components.

\[ s = uX \]
\[ d = vX^2 + wX^3 \]  
(E.23)

\[
\begin{align*}
u &= \frac{\alpha}{|\alpha| + |\beta| + |\delta|} \\
v &= \frac{\beta}{|\alpha| + |\beta| + |\delta|} \\
w &= \frac{\delta}{|\alpha| + |\beta| + |\delta|}
\end{align*}
\]  
(E.24)

The higher order moments of \( X \) are calculated using equations (E.11) through (E.15).

\[ E(X) = 0 \]
\[ E(X^2) = 1/3 \]
\[ E(X^3) = 0 \]
\[ E(X^4) = 1/5 \]
\[ E(X^5) = 0 \]
\[ E(X^6) = 1/7 \]

The variance of the undistorted signal \( s = uX \) is obtained by setting \( v \) and \( w \) equal to zero in equation (E.17) and applying (E.19):

\[ V(s) = E(s^2) - [E(s)]^2 = u^2 E(X^2) - 0 = \frac{u^2}{3} \]  
(E.25)

The variance of the distortion signal \( d = vX^2 + wX^3 \) is obtained by first computing the moments:
Statistics of a Nonlinear Function

\[ E(d) = vE(X^2) + uE(X^3) = v/3 \]  \hspace{1cm} (E.26)

\[ E(d^2) = v^2E(X^4) + 2vwE(X^5) + w^2E(X^6) \]
\[ = v^2/5 + w^2/7 \]  \hspace{1cm} (E.27)

hence,

\[ V(d) = v^2/5 + w^2/7 - \left[ v/3 \right]^2 = 4v^2/45 + w^2/7 \]  \hspace{1cm} (E.28)

The signal to distortion ratio (SDR) is determined as
\[ 10 \log_{10}(V(s)/V(d)) \]

Example 2: Signal-to-Distortion Ratio (SDR) Assuming a Normal Distribution for \( X \)

Assume \( X \) has a unit standard deviation. The higher order moments are calculated using equation (E.7).

\[
\begin{align*}
E(X) &= 0 \\
E(X^2) &= 1 \\
E(X^3) &= 0 \\
E(X^4) &= 3 \\
E(X^5) &= 0 \\
E(X^6) &= 5
\end{align*}
\]
since \( \sigma = 1 \).

The variance of the undistorted signal \( s = uX \) is obtained by applying (E.19):

\[ V(s) = E(s^2) - [E(s)]^2 = u^2E(X^2) - 0 = u^2 \]  \hspace{1cm} (E.29)

The variance of the distortion signal \( d = vX^2 + wX^3 \) is obtained by first computing the moments:

\[ E(d) = vE(X^2) + uE(X^3) = v \]  \hspace{1cm} (E.30)

\[ E(d^2) = v^2E(X^4) + 2vwE(X^5) + w^2E(X^6) \]
\[ = 3v^2 + 5w^2 \]  \hspace{1cm} (E.31)
hence,

\[ V(d) = 3v^2 + 5w^2 - v^2 = 2v^2 + 5w^2 \]  
(E.32)

\[ (E.33) \]

The SDR is determined as \( 10 \log_{10} (V(s)/V(d)) \).

### E.5 Implications for Adaptive Systems

From the above analysis, the SDR is a nonlinear function of both the input standard deviation (if noise) and the characteristics of the signal. This is illustrated in Figure E.1, which shows the calculated SDR for a cubic system similar to (E.21) where \( \alpha=1, \beta=\delta=0.2 \) are fixed as the standard deviation of the input noise signal is changed. The range of the SDR's is significantly different

![SDR for white noise input and combined quadratic/cubic nonlinearity](image)

**FIGURE E.1** Computed SDR for a cubic order system where the standard deviation of the input signal changes.

between the two signals due to the outliers in the normal distribution. This has important implica-
tions for adaptive systems that deal with inputs that have high peak-to-RMS ratios, for example speech. Essentially, the higher the peak-to RMS ratio encountered in the nonlinear domain, the lower the SDR and the worse the linear algorithms will perform in identifying the unknown system.