1

PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file - Votre référence

Our file - Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

**Analysis of Interference Excision Algorithms
For Direct Sequence Spread Spectrum Signals**

By

Harvey Hum, B.Eng.

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

Master of Engineering

Ottawa-Carleton Institute of Electrical Engineering
Faculty of Engineering
Department of Electronics
Carleton University
June 1994

© Harvey Hum, 1994

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

THE AUTHOR HAS GRANTED AN
IRREVOCABLE NON-EXCLUSIVE
LICENCE ALLOWING THE NATIONAL
LIBRARY OF CANADA TO
REPRODUCE, LOAN, DISTRIBUTE OR
SELL COPIES OF HIS/HER THESIS BY
ANY MEANS AND IN ANY FORM OR
FORMAT, MAKING THIS THESIS
AVAILABLE TO INTERESTED
PERSONS.

L'AUTEUR A ACCORDE UNE LICENCE
IRREVOCABLE ET NON EXCLUSIVE
PERMETTANT A LA BIBLIOTHEQUE
NATIONALE DU CANADA DE
REPRODUIRE, PRETER, DISTRIBUER
OU VENDRE DES COPIES DE SA
THESE DE QUELQUE MANIERE ET
SOUS QUELQUE FORME QUE CE SOIT
POUR METTRE DES EXEMPLAIRES DE
CETTE THESE A LA DISPOSITION DES
PERSONNE INTERESSEES.

THE AUTHOR RETAINS OWNERSHIP
OF THE COPYRIGHT IN HIS/HER
THESIS. NEITHER THE THESIS NOR
SUBSTANTIAL EXTRACTS FROM IT
MAY BE PRINTED OR OTHERWISE
REPRODUCED WITHOUT HIS/HER
PERMISSION.

L'AUTEUR CONSERVE LA PROPRIETE
DU DROIT D'AUTEUR QUI PROTEGE
SA THESE. NI LA THESE NI DES
EXTRAITS SUBSTANTIELS DE CELLE-
CI NE DOIVENT ETRE IMPRIMES OU
AUTREMENT REPRODUITS SANS SON
AUTORISATION.

ISBN 0-315-98579-8

Canada

Name

Harvey Hum

Dissertation Abstracts International is arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation. Enter the corresponding four-digit code in the spaces provided.

Electronics Engineering

SUBJECT TERM

0544

U·M·I

SUBJECT CODE

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS

Architecture	0729
Art History	0377
Cinema	0900
Dance	0378
Fine Arts	0357
Information Science	0723
Journalism	0391
Library Science	0399
Mass Communications	0708
Music	0413
Speech Communication	0459
Theater	0465

EDUCATION

General	0515
Administration	0514
Adult and Continuing	0516
Agricultural	0517
Art	0273
Bilingual and Multicultural	0282
Business	0688
Community College	0275
Curriculum and Instruction	0727
Early Childhood	0518
Elementary	0524
Finance	0277
Guidance and Counseling	0519
Health	0680
Higher	0745
History of	0520
Home Economics	0278
Industrial	0521
Language and Literature	0279
Mathematics	0280
Music	0522
Philosophy of	0998
Physical	0523

Psychology	0525
Reading	0535
Religious	0527
Sciences	0714
Secondary	0533
Social Sciences	0534
Sociology of	0340
Special	0529
Teacher Training	0530
Technology	0710
Tests and Measurements	0288
Vocational	0747

LANGUAGE, LITERATURE AND LINGUISTICS

Language	
General	0679
Ancient	0289
Linguistics	0290
Modern	0291
Literature	
General	0401
Classical	0294
Comparative	0295
Medieval	0297
Modern	0298
African	0316
American	0591
Asian	0305
Canadian (English)	0352
Canadian (French)	0355
English	0593
Germanic	0311
Latin American	0312
Middle Eastern	0315
Romance	0313
Slavic and East European	0314

PHILOSOPHY, RELIGION AND

THEOLOGY

Philosophy	0422
Religion	
General	0318
Biblical Studies	0321
Clergy	0319
History of	0320
Philosophy of	0322
Theology	0469

SOCIAL SCIENCES

American Studies	0323
Anthropology	
Archaeology	0324
Cultural	0326
Physical	0327
Business Administration	
General	0310
Accounting	0272
Banking	0770
Management	0454
Marketing	0338
Canadian Studies	0385
Economics	
General	0501
Agricultural	0503
Commerce-Business	0505
Finance	0508
History	0509
Labor	0510
Theory	0511
Folklore	0358
Geography	0366
Gerontology	0351
History	
General	0578

Ancient	0579
Medieval	0581
Modern	0582
Black	0328
African	0331
Asia, Australia and Ocean	0332
Canadian	0334
European	0335
Latin American	0336
Middle Eastern	0333
United States	0337
History of Science	0585
Law	0398
Political Science	
General	0615
International Law and	
Relations	0616
Public Administration	0617
Recreation	0814
Social Work	0452
Sociology	
General	0626
Criminology and Penology	0627
Demography	0938
Ethnic and Racial Studies	0631
Individual and Family	
Studies	0628
Industrial and Labor	
Relations	0629
Public and Social Welfare	0630
Social Structure and	
Development	0700
Theory and Methods	0344
Transportation	0709
Urban and Regional Planning	0999
Women's Studies	0453

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES

Agriculture	
General	0473
Agronomy	0285
Animal Culture and	
Nutrition	0475
Animal Pathology	0476
Food Science and	
Technology	0359
Forestry and Wildlife	0478
Plant Culture	0479
Plant Pathology	0480
Plant Physiology	0817
Range Management	0777
Wood Technology	0746
Biology	
General	0306
Anatomy	0287
Biostatistics	0308
Botany	0309
Cell	0379
Ecology	0329
Entomology	0353
Genetics	0369
Limnology	0793
Microbiology	0410
Molecular	0307
Neuroscience	0317
Oceanography	0416
Physiology	0433
Radiation	0821
Veterinary Science	0778
Zoology	0472
Biophysics	
General	0786
Medical	0760

EARTH SCIENCES

Biogeochemistry	0425
Geochemistry	0996

Geodesy	0370
Geology	0372
Geophysics	0373
Hydrology	0388
Mineralogy	0411
Paleobotany	0345
Paleoecology	0426
Paleontology	0418
Paleozoology	0985
Palynology	0427
Physical Geography	0368
Physical Oceanography	0415

HEALTH AND ENVIRONMENTAL SCIENCES

Environmental Sciences	0768
Health Sciences	
General	0566
Audiology	0300
Chemotherapy	0992
Dentistry	0567
Education	0350
Hospital Management	0769
Human Development	0758
Immunology	0982
Medicine and Surgery	0564
Mental Health	0347
Nursing	0569
Nutrition	0570
Obstetrics and Gynecology	0380
Occupational Health and	
Therapy	0354
Ophthalmology	0381
Pathology	0571
Pharmacology	0419
Pharmacy	0572
Physical Therapy	0382
Public Health	0573
Radiology	0574
Recreation	0575

Speech Pathology	0460
Toxicology	0383
Home Economics	0386

PHYSICAL SCIENCES

Pure Sciences

Chemistry	
General	0485
Agricultural	0749
Analytical	0486
Biochemistry	0487
Inorganic	0488
Nuclear	0738
Organic	0490
Pharmaceutical	0491
Physical	0494
Polymer	0495
Radiation	0754
Mathematics	0405
Physics	
General	0605
Acoustics	0986
Astronomy and	
Astrophysics	0606
Atmospheric Science	0608
Atomic	0748
Electronics and Electricity	0607
Elementary Particles and	
High Energy	0798
Fluid and Plasma	0759
Molecular	0609
Nuclear	0610
Optics	0752
Radiation	0756
Solid State	0611
Statistics	0463

Applied Sciences

Applied Mechanics	0346
Computer Science	0984

Engineering	
General	0537
Aerospace	0538
Agricultural	0539
Automotive	0540
Biomedical	0541
Chemical	0542
Civil	0543
Electronics and Electrical	0544
Heat and Thermodynamics	0348
Hydraulic	0545
Industrial	0546
Marine	0547
Materials Science	0794
Mechanical	0548
Metallurgy	0743
Mining	0551
Nuclear	0552
Packaging	0549
Petroleum	0765
Sanitary and Municipal	0554
System Science	0790
Geotechnology	0428
Operations Research	0796
Plastics Technology	0795
Textile Technology	0994

PSYCHOLOGY

General	0621
Behavioral	0384
Clinical	0622
Developmental	0620
Experimental	0623
Industrial	0624
Personality	0625
Physiological	0989
Psychobiology	0349
Psychometrics	0632
Social	0451

**The undersigned hereby recommend to
the Faculty of Graduate Studies and Research
acceptance of the thesis,**

**ANALYSIS OF INTERFERENCE EXCISION ALGORITHMS
FOR DIRECT SEQUENCE SPREAD SPECTRUM SIGNALS**

**submitted by
Harvey Hum, B.Eng.,**

**in partial fulfilment of the requirements
for the degree of Master of Engineering.**

A handwritten signature in cursive script, appearing to read "J. Wright", positioned above a horizontal line.

Chair, Department of Electronics

A handwritten signature in cursive script, appearing to read "J. Wright", positioned above a horizontal line.

Thesis Supervisor

Carleton University

June 1994

Abstract

The effectiveness of four interference excision algorithms for direct sequence spread spectrum signals has been investigated. These algorithms, the Fast-Fourier-Transform (FFT), the Least-Mean-Squares (LMS), the Recursive-Least-Squares (RLS), and Wiener-Hopf, use either spectral analysis or linear prediction to design transversal filters to suppress a narrowband continuous tone. The performance of each algorithm was investigated first by simulation and later experimentally using the Intel iWarp parallel process system on which the algorithms were implemented. The results of the analyses have shown that all four algorithms had effectively suppressed the interference tone and provided an adequate performance level under a hostile environment.

Acknowledgements

I would like to thank Professor J.S. Wight for his guidance and supervision throughout this thesis.

I would like to thank Dr. Raymond Charbonneau and the staff of the former Communications Electronic Warfare Section, Defence Research Establishment Ottawa for providing me the opportunity to perform this thesis at such a prestigious facility and with the use of their resources. .

I would like to express my appreciation to Dr. Brian Kozminchuk at DREO for his wisdom, insight and guidance of this thesis at DREO.

I would also like to thank Rene Lamontagne and Andrew Mudry for their help both mentally, physically, and spiritually during times of crisis.

Last but not least, I would like to express my gratitude to my friends and family who helped and supported me throughout the course of this thesis.

Table of Contents

Abstract	ii
Acknowledgements	iii
List of Figures	vii
List of Tables	xiii
Chapter 1 Introduction	
1.1 Motivation	1
1.2 Thesis Objectives	2
1.3 Thesis Organization	3
Chapter 2 Background Theory	
2.1 Introduction	4
2.2 Spread Spectrum Systems	4
2.2.1 <i>Introduction</i>	4
2.2.2 <i>Direct-Sequence Spread Spectrum</i>	5
2.2.3 <i>Rejection of Intentional Jamming</i>	11
2.3 Wiener Filter Theory	13
2.3.1 <i>Introduction</i>	13
2.3.2 <i>Implementation of the Wiener Filter Using Adaptive Filters</i>	14
2.3.3 <i>Error Performance Surface</i>	16
2.3.4 <i>The Normal Equation</i>	18
2.4 Linear Prediction	20
2.4.1 <i>Introduction</i>	20
2.4.2 <i>Forward Linear Prediction</i>	21
2.4.3 <i>Normal Equation Under Forward Linear Prediction</i>	23
2.4.4 <i>Backward Linear Prediction</i>	24
2.4.5 <i>Normal Equation Under Backward Linear Prediction</i>	26
2.5 Summary	27

Chapter 3 Interference Excision Algorithms

3.1 Introduction	28
3.2 Fast Fourier Transform	29
3.2.1 <i>Introduction</i>	29
3.2.2 <i>FFT Algorithm</i>	29
3.3 Least-Mean-Squares	31
3.3.1 <i>Introduction</i>	31
3.3.2 <i>Derivation of the LMS Algorithm</i>	31
3.3.3 <i>Convergence of the LMS Algorithm</i>	33
3.3.4 <i>LMS Algorithm Based on Linear Prediction</i>	35
3.4 Recursive-Least-Squares	36
3.4.1 <i>Introduction</i>	36
3.4.2 <i>RLS algorithm</i>	39
3.4.3 <i>RLS Algorithm Based on Linear Prediction</i>	42
3.4.4 <i>Fast Transversal Filters Algorithm</i>	44
3.5 Wiener-Hopf	50
3.5.1 <i>Introduction</i>	50
3.5.2 <i>Wiener-Hopf Algorithm</i>	50
3.5.3 <i>Summary</i>	53

Chapter 4 Simulation Results

4.1 Introduction	55
4.2 DS-SS Communications Model	55
4.3 Simulation Results	57
4.3.1 <i>Simulation Parameters</i>	57
4.3.2 <i>FFT Algorithm</i>	67
4.3.3 <i>LMS Algorithm</i>	85
4.3.4 <i>RLS Algorithm</i>	95
4.3.5 <i>Wiener-Hopf Algorithm</i>	102
4.4 Summary	107

Chapter 5 Experimental Communication System

5.1 Introduction	109
5.2 System Overview	109
5.3 System Components	112
5.3.1 <i>Spread Spectrum Generator</i>	112
5.3.2 <i>Carrier Frequency Generator</i>	114
5.3.3 <i>Interference Generator</i>	116
5.3.4 <i>Noise Generator</i>	116
5.3.5 <i>Steinbrecher Receiver</i>	116
5.3.6 <i>iWarp DSP Boards</i>	117
5.3.7 <i>Sun SparcStation</i>	118
5.4 System Limitations	118
5.5 Summary	120

Chapter 6 Experimental Results

6.1 Introduction	121
6.2 Experimental Parameters	121
6.3 FFT Algorithm	123
6.3.1 <i>System Configuration</i>	123
6.3.2 <i>Results</i>	132
6.4 LMS Algorithm	142
6.4.1 <i>System Configuration</i>	142
6.4.2 <i>Results</i>	143
6.5 RLS Algorithm	150
6.5.1 <i>System Configuration</i>	150
6.5.2 <i>Results</i>	151
6.6 Summary	157

Chapter 7 Conclusions

7.1 Summary	160
7.2 Future Work	162

Appendix A Matrix Inversion Lemma	163
--	-----

References	164
-----------------------------	-----

List of Figures

Figure 2.1 Model of a DS/BPSK System.	7
Figure 2.2 DS/BPSK spreading and despreading signals.	8
Figure 2.3 Power spectral density of (a) BPSK modulated carrier, (b) spreading code, and (c) data and spreading code modulated carrier.	10
Figure 2.4 Receiver power spectral densities with tone jamming.	12
Figure 2.5 (a) Adaptive transversal filter. (b) Adaptive transversal filter with desired response and error signals.	15
Figure 2.6 Two dimensional performance surface with optimum weight vector at W^* . . .	19
Figure 2.7 (a) Forward linear predictor; (b) prediction-error filter; (c) relationship between the predictor and the prediction-error filter.	22
Figure 2.8 (a) Backward linear predictor; (b) backward prediction-error filter; (c) backward prediction-error filter in terms of the forward prediction-error tap weights.	25
Figure 3.1 Adaptive linear predictor using LMS algorithm.	37
Figure 3.2 Simplified flow graph of the RLS algorithm as a linear predictor.	45
Figure 3.3 Fast transversal filters to realize RLS solution.	47
Figure 4.1 Baseband PN spread spectrum communications model.	56
Figure 4.2 A sample of the data bits $D(t)$ and its corresponding power spectral estimation.	50
Figure 4.3 A sample of the pseudo-noise sequence $PN(t)$ used to spread the signal and its corresponding power spectral estimation.	61
Figure 4.4 A sample of the spread spectrum modulated signal $s(t)$ and its power spectral estimation at the output of the Matched filter sampled at the chip rate of 1 Hz.	62
Figure 4.5 A sample of the noise $n(t)$ and its corresponding power spectral estimation.	63
Figure 4.6 A sample of the interference signal $i(t)$ and its corresponding power spectral estimation.	64
Figure 4.7 A sample of the received signal r_k at the output of the Matched filter with noise and interference added to the spread signal and its corresponding power spectral estimation. For this case, $P_I/P_S = 20$ dB and $E_b/N_o = 12$ dB.	65

Figure 4.8	A sample of the transmitted data D_k and the corresponding decoded data $\hat{D}_k$ with no interference suppression algorithm, $P_I/P_S = 20$ dB and $E_b/N_o = 12$ dB.	66
Figure 4.9	Power spectral estimation of received signal r_k based on parameters listed in Table 4.2. The lower graph is a magnified version of the upper graph to illustrate the small contribution of the SS signal and noise.	71
Figure 4.10	Interpolated power spectral estimation of Fig.4.9 from 16 points to 15 points.	72
Figure 4.11	Square root of the power spectral estimation of Fig.4.10.	73
Figure 4.12	Inverse square root of the power spectral estimation of Fig.4.10.	74
Figure 4.13	The magnitude and phase of the transfer function H_k of the interference suppression filter.	75
Figure 4.14	The normalized impulse response or tap vector w_k of the linear suppression filter H_k	76
Figure 4.15	The continuous spectrum, $H(e^{j\omega})$ of the interference suppression filter with impulse response w_k for a 15-tap filter.	77
Figure 4.16	The power spectral estimation of the sampled signal before the suppression filter (upper plot) and the power spectral estimation after the suppression filter (bottom plot).	78
Figure 4.17	The transmitted data sequence D_k (64 bits) and the demodulated data $\hat{D}_k$ after the 15-tap suppression filter.	79
Figure 4.18	The continuous spectrum, $H(e^{j\omega})$ of the interference suppression filter with impulse response w_k for a 31-tap filter.	80
Figure 4.19	The transmitted data sequence D_k (64 bits) and the demodulated data $\hat{D}_k$ after the 31-tap suppression filter.	81
Figure 4.20	Bit error rate performance for a single tone interferer with $P_I/P_S = 20$ dB at frequency 0.125 Hz for 7, 15, and 31 tap filters. . .	82
Figure 4.21	The suppression filters for interference located at frequencies 0.111 Hz, 0.125 Hz, 0.2 Hz, 0.25 Hz, 0.35 Hz, and 0.375 Hz.	83
Figure 4.22	Bit error rate performance for a single tone interferer with $P_I/P_S = 20$ dB at frequency 0.111 Hz for 7, 15, and 31 tap filters. . .	84
Figure 4.23	Convergence of the first three weight tap vector from an initial starting point of [0,0,0].	88
Figure 4.24	Plot of the squared error function with $\mu = 0.0001$	89

Figure 4.25	Power spectral estimation of the received sampled data (top graph) and the power spectral estimation of the data with the LMS algorithm (bottom graph) with $\mu = 0.0001$	90
Figure 4.26	Transmitted data, D_k , and the demodulated data, $\hat{D}_k$, once the received data has been passed through the algorithm with $\mu=0.0001$. The bottom graph illustrates the squared error between the transmitted and decoded bits. .	91
Figure 4.27	Bit error rate curve for 7, 15, and 31 tap vectors for the LMS algorithm with $\mu=0.0001$	92
Figure 4.28	Squared error plot as a function of iterations k , and step size μ	93
Figure 4.29	Bit error rate curve for values of $\mu = 0.01, 0.001$, and 0.0001	94
Figure 4.30	Squared error function for RLS algorithm with $\lambda = 0.9999$	96
Figure 4.31	Power spectral estimation of the received sampled signal with interference and noise (top graph) and the power spectral estimation of the signal using the RLS algorithm (bottom graph) with $\lambda = 0.9999$. .	99
Figure 4.32	Transmitted data D_k and the decoded data $\hat{D}_k$ using the RLS algorithm with $\lambda=0.9999$	100
Figure 4.33	Bit error rate curve for 7, 15, and 31 tap vectors for the RLS algorithm with $\lambda=0.9999$	101
Figure 4.34	Power spectral estimation of the first 1024 received data samples (top graph) and the filtered data for this block (bottom graph).	104
Figure 4.35	Transmitted data, D_k , and the decoded data, $\hat{D}_k$, using the Wiener-Hopf algorithm with a 15 tap weight vector for the first 1024 sample data block.	105
Figure 4.36	Bit error rate performance of the Wiener-Hopf algorithm with tap sizes of 7, 15, and 31.	106
Figure 4.37	Bit error rate performances of the four algorithms in comparison to the performance with no suppression algorithm and with the theoretical BER of BPSK signal with AWGN only.	108
Figure 5.1	Experimental system setup.	110
Figure 5.2	Circuit diagram of spread spectrum generator.	113
Figure 5.3	Generated signals of the spread spectrum circuit in Figure 5.2.	115
Figure 5.4	(a) Two-dimensional torus array for a 9-cell iWarp system. (b) iWarp system architecture and I/O capacity.	119

Figure 6.1 (a) Time data of the baseband spread spectrum signal with [4,1] PN and data sequences, chip rate of 156.25 KHz and $P_G=15$. (b) Power spectrum of the signal in (a) modulated on a carrier centered at 15 MHz.	124
Figure 6.2 (a) Time data of AWGN signal. (b) Power spectrum of the AWGN signal.	125
Figure 6.3 (a) Sinusoidal wave to create a continuous narrowband interference signal with a frequency of 15.020 MHz and $P_I/P_S = 20$ dB. (b) Power spectrum of the interference tone.	126
Figure 6.4 Power spectrum of the analog signal containing the spread spectrum, AWGN, and interference signals.	127
Figure 6.5 Power spectrum of the received digitized signal demodulated to baseband and decimated by a factor of 32 such that sampling rate = chip rate = 156.25 KHz.	128
Figure 6.6 Transmitted data bit (upper graph), and decoded data bits (middle graph) with no excision algorithm on the received jammed signal. The squared error for this set of data is shown in the lower graph.	129
Figure 6.7 The cell responsibilities on the iWarp system to perform the FFT algorithm.	130
Figure 6.8 Power spectral estimation of the received digital signal by averaging 64 16-pt FFTs.	134
Figure 6.9 The magnitude (upper plot) and phase (lower plot) of the transfer function of the suppression filter to be designed.	135
Figure 6.10 The normalized impulse response w_k of the linear suppression filter. .	136
Figure 6.11 The continuous spectrum $H(e^{j\omega})$ of the suppression filter with impulse response in Fig.6.10.	137
Figure 6.12 (a) The power spectral estimation of the received digital signal before the FFT algorithm. (b) The power spectral estimation of the filtered data using filter in Fig.6.11.	138
Figure 6.13 (a) The correlation signal between the [4,1] PN sequence and the FFT filtered digital signal. (b) Bit detection of the filtered spread spectrum signal.	139
Figure 6.14 (a) The transmitted bits before spreading with the PN sequence. (b) The decoded data bits of the FFT filtered spread spectrum signal. (c) Squared error between (a) and (b).	140

Figure 6.15	Comparison of the BER curve of the FFT algorithm with BER curves using no algorithm, signal in AWGN only, and the theoretical BER for a BPSK signal in AWGN only. For the FFT algorithm, 16 pt FFTs and 15 taps were used.	141
Figure 6.16	The cell responsibilities on the iWarp system to perform the LMS algorithm.	144
Figure 6.17	(a) Power spectrum of the unfiltered data with interference at 20 KHz and P_I/P_S approximately 20 dB. (b) Power spectrum of the filtered data using the LMS algorithm to suppress the interference. The number of taps was 15 and $\mu = 1E-7$	145
Figure 6.18	The square error function of the LMS filtered data with $\mu = 1E-7$. . .	146
Figure 6.19	(a) The correlation signal between the [4,1] PN sequence and the LMS filtered data. (b) Bit detection of the filtered spread spectrum signal. .	147
Figure 6.20	(a) The transmitted bits before spreading with PN sequence. (b) The decoded data bits of the LMS filtered spread spectrum signal. (c) Squared error between (a) and (b).	148
Figure 6.21	Comparison of the BER curve of the LMS algorithm with BER curves using no algorithm, signal in AWGN only, and the theoretical BER for a BPSK signal in AWGN only. For the LMS algorithm, 15 taps and $\mu = 1E-7$ were used.	149
Figure 6.22	The cell responsibilities on the iWarp system to perform the RLS algorithm.	152
Figure 6.23	(a) Power spectrum of the unfiltered data with interference at 20 KHz and P_I/P_S approximately 20 dB. (b) Power spectrum of the filtered data using the RLS algorithm to suppress the interference. The number of taps was 15 and $\lambda = 0.9999$	153
Figure 6.24	The square error function of the RLS filtered data with $\lambda = 0.9999$. .	154
Figure 6.25	(a) The transmitted bits before spreading with the PN sequence. (b) The decoded data bits of the RLS filtered spread spectrum signal. (c) Squared error between (a) and (b).	155
Figure 6.26	Comparison of the BER curve of the RLS algorithm with BER curves using no algorithm, signal in AWGN only, and the theoretical BER for a BPSK signal in AWGN only. For the RLS algorithm, 15 taps and $\lambda = 0.9999$ were used.	156

Figure 6.27 Comparison of the experimental BER curves for the FFT, LMS and RLS algorithms using 15 taps.	159
--	-----

List of Tables

Table 3.1	Summary of the fast Fourier transform algorithm for interference excision. .	32
Table 3.2	Summary of the LMS algorithm for linear prediction.	38
Table 3.3	Summary of the RLS algorithm.	43
Table 3.4	Initialization of the RLS algorithm.	51
Table 3.5	Summary of the main RLS algorithm.	52
Table 4.1	Parameters to generate data.	59
Table 4.2	Parameters for the FFT algorithm.	70
Table 4.3	Parameters for the LMS algorithm.	87
Table 4.4	Parameters for the RLS algorithm.	97
Table 4.5	Parameters for the Wiener–Hopf algorithm.	103
Table 6.1	Parameters for performing the FFT algorithm on the iWarp system. . . .	131

Chapter 1

Introduction

1.1 Motivation

In the military, the main objective of communication systems is to provide a form of secure and reliable communication between allied forces in a hostile environment. In such an environment the enemy may be attempting to detect and intercept the transmitted signal in order to gain some valuable information, i.e., location of forces or strategic plans, or to hinder communications by using a hostile transmitter to apply an intentional interference in an attempt to jam the transmission. Many commercially used communication methods such as AM, FM, BPSK, FSK, etc., are very susceptible to both detection and interception as well as intentional jamming. One method that is widely used to combat these situations is to use direct-sequence spread spectrum (DS-SS) signals.

One major characteristic of direct-sequence spread spectrum communication systems is the inherent ability to combat narrowband interference or intentional jamming as a result of the processing gain of the system. However, when the power of the interferer is much greater than that of the spread spectrum system or there are bandwidth restrictions, the processing gain may not be adequate to effectively suppress the interferer. This will result in a severe degradation in system performance as seen by a sudden increase in bit error rate (BER). To enhance the system performance, at the expense of introducing some distortion onto the signal, the interference can be filtered out using various techniques. This process of filtering is referred to as interference excision.

Several analog and digital excision techniques have been presented in the past [1], [2], and [3]. However many of these techniques are computationally complex and/or requires some knowledge of the interferer, i.e., power and frequency. In the military environment, such a priori knowledge of the interference may not be readily available. Also, since fast data processing is essential, these techniques may not be realistic in such an environment.

Thus, the motivation for this thesis is to provide several possible digital interference excision techniques which do not require a priori knowledge of the interferer and is capable of suppressing an intentional interference in a hostile environment with an adequate bit error rate performance.

1.2 Thesis Objectives

The objectives of this thesis are: to examine several possible digital interference excision algorithms, determine their performance with simulations, provide an experimental communication system on which these algorithms may be implemented, and to determine their performance experimentally under an intentional jamming environment.

These algorithms are based on spectral analysis and adaptive linear prediction. Simulation results for each algorithm will be presented and their performance compared. An experimental communication system will be presented with the algorithms to be implemented on a high speed parallel processing board to process real data. This system contains all the necessary components to simulate a real hostile environment, i.e., spread spectrum generator, additive white gaussian noise (AWGN) generator, interference

generator, and a wideband receiver. The experimental results will then be presented and compared with the simulations. Conclusions will then be drawn from the results and future work that will be needed will be outlined.

1.3 Thesis Organization

This thesis is divided up into seven chapters with their contents summarized below.

Chapter 2 discusses the relevant background theory that is required in order to understand these algorithms. This chapter will briefly examine spread spectrum modulation with an emphasis on direct-sequence spread spectrum signals, the Wiener filter theory, and linear prediction theory as they apply to interference excision algorithms.

Chapter 3 will discuss in detail the four algorithms that will be examined in this thesis. These include the Fast Fourier Transform (FFT), the Least-Mean-Square (LMS), the Recursive-Least-Square (RLS), and the Wiener-Hopf algorithms.

Chapter 4 outlines a theoretical communication system under jamming conditions and simulation results for each algorithm will be given. A comparison of the performances between the different algorithms will also be discussed.

Chapter 5 will detail an experimental communication system which can implement the excision algorithms on a high speed DSP board. Chapter 6 reports the results of the experiments using the system described in Chapter 5 with real data for each algorithm except the Wiener-Hopf.

Finally, Chapter 7 gives a brief summary and conclusions along with recommendations for future work.

Chapter 2

Background Theory

2.1 Introduction

This chapter discusses the background information that is necessary in order to fully understand the interference excision algorithms. A brief discussion on spread spectrum systems, with an emphasis on direct-sequence spread spectrum signals, and its effectiveness in combatting intentional jamming will be given. The Wiener filter, which is optimal in the mean square error sense, will also be discussed as they apply to adaptive transversal filters. Finally, the basis for the LMS, RLS, and Wiener-Hopf algorithms, i.e. adaptive linear prediction, will be examined.

2.2 Spread Spectrum Systems

2.2.1 Introduction

In many real-world stationary AWGN environments, standard digital modulation/demodulation techniques such as frequency-shift-keying (FSK), phase-shift-keying (PSK), and quadrature amplitude modulation (QAM) etc., are relatively efficient in transmitting information from one location to another.

However, in a military environment, it is not uncommon for a communication system to be subjected to intentional jamming through the use of a continuous interference tone. In such a situation, the channel can no longer be modelled as a stationary AWGN channel and the communication techniques mentioned above are no longer dependable. A modulation technique which has demonstrated an inherent anti-jamming capability,

and is widely used throughout the military and more recently in civilian applications, is spread spectrum. Spread spectrum modulation

is a means of transmission in which the signal occupies a bandwidth in excess of the minimum necessary to send the information; the band spread is accomplished by means of a code which is independent of the data, and a synchronized reception with the code at the receiver is used for despreading and subsequent data recovery. [4]

In addition to the anti-jamming capabilities, spread spectrum signals have other advantages which are very attractive to the military world. These include:

- (1) low probability of intercept (LPI),
- (2) interference rejection including multipath rejection, and
- (3) multiple user random access communications.

There are many types of spread spectrum modulation, including

- (1) direct-sequence (DS),
- (2) frequency-hopped (FH),
- (3) time-hopped (TH),
- (4) chirp, and
- (5) hybrid.

In this thesis, however, only direct-sequence modulation will be studied and discussed.

Anyone interested in the other types of modulation should refer to [5], [6], and/or [7].

2.2.2 Direct-Sequence Spread Spectrum

Direct-sequence spread spectrum systems involves modulating the digital information signal with a wideband spreading signal. From the Fourier transform theory, multiplica-

tion of two unrelated signals in the time domain will result in the convolution of the two spectra in the frequency domain. Therefore, the result of multiplying a narrowband signal (data), with a wideband signal (PN sequence), is a new signal whose spectral envelope is similar to the wideband signal's. In effect, the narrowband power of the signal becomes spread out over a wider frequency band, thus allowing the transmitted signal to assume a noise-like appearance.

This wideband signal, more commonly known as a pseudo-noise (PN) random sequence, is usually some form of digital modulation: binary phase-shift-keying (BPSK), quadrature phase-shift-keying (QPSK), or minimum-shift-keying (MSK). Since the focus of this thesis is on the interference excision algorithms and not spread spectrum communications, the type of modulation that will be used in this thesis is BPSK since it is the easiest type to implement and decode.

Figure 2.1 illustrates a simplified model for a DS/BPSK spread spectrum communication system and Fig.2.2 details the workings of this model. The transmitted data $d(t)$ is shown in Fig.2.2(a) with a data rate of $1/T_b$. This data is then modulated using BPSK to achieve a constant-envelope signal $s_d(t)$ with power P as shown in Fig.2.2(b). DS/BPSK spreading is achieved by multiplying the signal $s_d(t)$, with the spreading code $c(t)$ of chip rate $1/T_c$ shown in Fig.2.2(c). The resulting spread signal $s_t(t)$ in Fig.2.2(d) can then be transmitted via a distortionless path with delay T_d assuming no jamming signal, $j(t)$, is present.

To demodulate the received signal, the spreading code must first be removed, i.e., the received signal must be despread. This is achieved by remodulating the received signal

Figure 2.1 Model of a DS/BPSK System.

Figure 2.2 DS/BPSK spreading and despreading signals.[6]

with the spreading code synchronized to take the transmission delay into account. Once code synchronization has been achieved, the output of the despreader becomes $s_d(t)$ with additive noise and a conventional coherent phase demodulator can be used to demodulate the signal. Figures 2.2(f) and (g) illustrate the results with both unsynchronized and synchronized despreading, respectively.

To comprehend how this method of spreading is useful in military communications, a look at the power spectrum is required. Figure 2.3(a) shows the power spectral density of data that has been modulated using BPSK. The constant-envelope carrier is assumed to have power P , frequency f_o , and data phase modulation $\theta_d(t)$, i.e.,

$$s_d(t) = \sqrt{2P} \cos(2\pi f_o t + \theta_d(t)). \quad (2.1)$$

The signal has a bandwidth (peak to null) equal to the data rate (i.e., $1/T_b$) and an overall spectrum with the distinct $\text{sinc}^2(x)$ form. In normal communication systems, this signal would be transmitted and then demodulated at the receiver. However, this signal can easily be detected with the use of spectral analysis and can easily be intercepted or jammed. This is where direct-sequence spread spectrum modulation becomes useful.

Figure 2.3(b) represents the power spectral density of a PN sequence. Note that it has the same features as the modulated carrier except that it has a larger bandwidth and a much lower PSD since the data rate is much slower than the chip rate (i.e., $T_c \ll T_b$). A useful characteristic of a DS-SS system is the processing gain (PG) which can be defined as the ratio of the chip rate to data rate (i.e., T_b/T_c). Once the BPSK signal has been spread, the bandwidth of the BPSK signal has now been increased by a factor of this

Figure 2.3 Power spectral density of (a) BPSK modulated carrier, (b) spreading code, and (c) data and spreading code modulated carrier.

processing gain while the power spectral density of the spread signal has decreased by this same factor, as illustrated in Fig.2.3(c). This signal can now be transmitted with a relatively flat power spectrum thus making it difficult for the enemy to locate the signal frequency to intercept and/or jam the signal.

One of the major advantages of spread spectrum is its ability to reject intentional jamming or interference. This will now be discussed.

2.2.3 Rejection of Intentional Jamming

When an interference tone is applied to the DS-SS signal, its power spectral density is more highly peaked than that of the transmitted signal. At the receiver, the signal must first be despread. While this is occurring, the spectrum of the interference tone is being spread. If the interference energy is spread over a bandwidth that is much larger than the data bandwidth, most of the energy will be rejected by the data filter.

Assume that the interference is a continuous single tone with power J . With BPSK used for both data and spreading modulation, the obvious frequency to place the interference is at or near the center of the transmission bandwidth. Figure 2.4 illustrates the anti-jamming properties. In Fig.2.4(a) the spread signal is being jammed at the center of the bandwidth with a jammer power of J . Note the wideband and narrowband characteristics of the spread data and jamming signal respectively. Figure 2.4(b) shows the PSD at the output of the despreader in the receiver. In this case, the spread signal has now been returned to its original form while the interference tone has become spread. The despreading operation in spread spectrum receivers is followed by a filtering operation to limit the bandwidth at the input to the data demodulator to approximately the data

Figure 2.4 Receiver power spectral densities with tone jamming.[6]

bandwidth, as illustrated in Fig.2.4(c). Nearly all of the signal power passes through this filter while a large percentage of the spread jammer power is filtered out, Fig.2.4(d). By using DS-SS, the jamming power can be effectively reduced by a factor of the inverse of the processing gain.

However, in certain situations when the power of the interferer is much greater than that of the spread spectrum system, or the ratio of the wideband spread signal to the narrowband interferer is not large enough, the interference will not be effectively suppressed by the spread spectrum system. As a result, further interference suppression is required, which is the focus of this thesis.

2.3 Wiener Filter Theory

2.3.1 Introduction

In signal analysis, one of the most important operations is linear filtering, which involves the extraction of information at time t by using a linear function of data that has been measured up to and including t .

The criterion for designing a linear filter is to minimize the mean-squared value of an error signal, which is defined as the difference between a desired output and the actual filter output when its input contains noisy data. For stationary inputs, when certain statistical parameters such as the mean and correlation functions of the signal and unwanted additive noise are known, the desired linear filter is often referred to as the Wiener filter which is optimal in the mean-squared sense.

Note however that the Wiener filter cannot be used when the signal and/or noise is nonstationary. In this case, the optimal filter must assume a time-varying form.

2.3.2 Implementation of the Wiener Filter Using Adaptive Filters

As mentioned in the previous section, the Wiener filter requires a priori statistical information of the data to be processed, in particular the autocorrelation function. However, in certain cases this information may not be available which results in the design of a less than optimal filter or the inability to design one at all. To overcome this problem, an adaptive filter using recursive algorithms can be used which estimates the autocorrelation function. This algorithm begins at some predetermined set of initial conditions, and after a few iterations, will converge to and fluctuate about the optimal Wiener solution.

For the theoretical Wiener filter, a tapped-delay-line or transversal filter is used as the basis for implementing the adaptive filter. For this type of filter, the finite impulse response is defined by a set of tap weights of which the mean-squared error is a second order function. This error has the form of a multidimensional paraboloid with a uniquely defined minimum point and is referred to as the error performance surface. The tap weights corresponding to the minimum point of the surface is the optimum Wiener solution.

The adaptive filter illustrated in Fig.2.5(a) consists of three basic operations: storage, multiplication, and addition. The storage is represented by $N-1$ delays, resulting in the input signal vector $[x_k \ x_{k-1} \ x_{k-2} \ \dots \ x_{k-N}]$. The number of delay elements in the filter represents the order of the transverse filter. The elements of this vector are then

(a)

(b)

Figure 2.5 (a) Adaptive transversal filter. (b) Adaptive transversal filter with desired response and error signals.

multiplied with a corresponding set of adjustable weights, $[w_{0k} \ w_{1k} \ w_{2k} \ \dots \ w_{Nk}]$ and the results summed together to produce a single output y_k .

Assuming that the desired response is known, the problem is to choose values for the weights, $[w_{0k} \ w_{1k} \ w_{2k} \ \dots \ w_{Nk}]$, such that the difference between the desired response at time k , and the filter output at time k is minimized, i.e.,

$$\epsilon_k = d_k - y_k. \quad (2.2)$$

In the case of the Wiener filter, the filter is optimized in the minimum mean-squared sense,

$$L(\mathbf{W}) = E[\epsilon_k \epsilon_k^*] \quad (2.3)$$

where $\mathbf{W}$ is the tap-weight vector, and $L(\mathbf{W})$ is the mean-squared error, which is real and positive.

2.3.3 Error Performance Surface

To simplify the mathematics for the error performance surface, vector notations will be used for the tap weights and input signals. The N tap-weights may then be expressed as

$$\mathbf{W}^T = [w_{0k} \ w_{1k} \ w_{2k} \ \dots \ w_{Nk}] \quad (2.4)$$

where $\mathbf{T}$ represents the matrix transpose. The N signal inputs at time k is

$$\mathbf{X}_k^T = [x_k \ x_{k-1} \ x_{k-2} \ \dots \ x_{k-N}]. \quad (2.5)$$

The output of the filter can now be expressed as

$$y_k = \mathbf{X}_k^T \mathbf{W} = \mathbf{W}^T \mathbf{X}_k. \quad (2.6)$$

Substituting Eq.(2.6) into Eq.(2.2),

$$\epsilon_k = d_k - \mathbf{X}_k^T \mathbf{W} = d_k - \mathbf{W}^T \mathbf{X}_k \quad (2.7)$$

the mean-squared error can now be written as

$$L(\mathbf{W}) = E \left[\left(d_k - \mathbf{W}^T \mathbf{X}_k \right) \left(d_k^* - \mathbf{X}_k^T \mathbf{W} \right) \right] \quad (2.8)$$

where d_k^* is the complex conjugate of the desired signal at time k . Expanding the terms in the bracket,

$$L(\mathbf{W}) = E[d_k d_k^*] - \mathbf{W}^T E[\mathbf{X}_k d_k^*] - E[d_k \mathbf{X}_k^T] \mathbf{W} + \mathbf{W}^T E[\mathbf{X}_k \mathbf{X}_k^T] \mathbf{W}. \quad (2.9)$$

Assuming that d_k has zero mean, the first term on the right hand side of Eq.(2.9) is the variance of the desired response,

$$\sigma_d^2 = E[d_k d_k^*] \quad (2.10)$$

The second term on the right is the set of cross correlations between the tap-input vector and the desired response. This cross-correlation can be denoted by

$$\mathbf{P} = E[\mathbf{X}_k d_k] = E[x_k d_k \quad x_{k-1} d_k \quad x_{k-2} d_k \quad \dots \quad x_{k-N} d_k]^T \quad (2.11)$$

and using a property of the transpose,

$$\mathbf{P}^T = E[d_k \mathbf{X}_k^T]. \quad (2.12)$$

The final term in the Eq.(2.9) is equal to the correlation matrix of the tap-input vector $\mathbf{X}_k$ and can be denoted by

$$\mathbf{R} = E[\mathbf{X}_k \mathbf{X}_k^T]. \quad (2.13)$$

Note that for stationary process, this correlation matrix is similar to the autocorrelation function of the input. Equation (2.9) can now be expressed as

$$L(\mathbf{W}) = \sigma_d^2 - \mathbf{W}^T \mathbf{P} - \mathbf{P}^T \mathbf{W} + \mathbf{W}^T \mathbf{R} \mathbf{W} \quad (2.14)$$

This equation is known as the error performance surface and represents the mean-squared error. To minimize this error, the tap-weight vector must be adjusted to its optimum value. Figure 2.6 shows a typical two-dimensional mean-squared error function. The vertical axis represents the mean-squared error and the horizontal axes the two weights, w_1 , and w_2 . The optimal weight vector or point of minimum mean-squared error is located at $\mathbf{W}^*$.

2.3.4 The Normal Equation

To determine the optimum tap-weight vector $\mathbf{W}^*$, Eq. (2.14) can be differentiated with respect to the tap-weight vector and the result set to the null vector. To differentiate vectors, the gradient operator, ∇ , where

$$\nabla = \frac{d}{d\mathbf{W}} = \left[\frac{\partial}{\partial w_1} \quad \frac{\partial}{\partial w_2} \quad \cdots \quad \frac{\partial}{\partial w_N} \right]^T \quad (2.15)$$

is used. Using this operator, the gradient of the mean-squared-error performance surface is

$$\nabla[L(\mathbf{W})] = 2\mathbf{R}\mathbf{W} - 2\mathbf{P}. \quad (2.16)$$

To obtain the minimum mean-squared error,

$$\nabla[L(\mathbf{W})] = \mathbf{0} = 2\mathbf{R}\mathbf{W}^* - 2\mathbf{P} \quad (2.17)$$

resulting in

$$\mathbf{R}\mathbf{W}^* = \mathbf{P}. \quad (2.18)$$

Figure 2.6 Two dimensional performance surface with optimum weight vector at $\mathbf{W}^*$. [8]

Equation (2.18) is the discrete form of the Wiener-Hopf equation and is known as the normal equation. Assuming that $\mathbf{R}$ is non-singular, the optimal weight vector is

$$\mathbf{W}^* = \mathbf{R}^{-1}\mathbf{P}. \quad (2.19)$$

Therefore, in order to determine the optimum tap-weight vector, two quantities, the correlation matrix $\mathbf{R}$ of the tap-input vector and the cross-correlation vector $\mathbf{P}$ between the tap-input and the desired response must be known.

However, in practical situations these quantities are unknown and thus must be estimated. Linear prediction techniques can be used to perform this estimation and is discussed next.

2.4 Linear Prediction

2.4.1 Introduction

In time series analysis, one of the most often used algorithm is linear prediction. Linear prediction is used to predict a future value of a stationary discrete-time stochastic process by analyzing a set of past sample values of the process. In other words, to predict the value x_k , the previous N data, $x_{k-1}, x_{k-2}, \dots, x_{k-N}$, must be retained and processed in some form as to provide a good estimate of x_k . When the estimate is formed as a linear combination of the previous N samples, i.e.,

$$\hat{x}_k = \sum_{n=1}^N d_n x_{k-n} \quad (2.20)$$

where $d_1, d_2, \dots, d_N$ are constant coefficients, the prediction is called linear.

The method described above is referred to as forward prediction since it predicts the next value. However, it can also be used to predict a value in the past. For example, the

samples $x_k, x_{k-1}, x_{k-2}, \dots, x_{k-N+1}$, can be used to predict the value of x_{k-N} . This is known as backward prediction. In this section, both types of prediction will be discussed.

2.4.2 Forward Linear Prediction

As mentioned in the previous section, forward linear prediction involves the linear combination of N previous data to predict the data x_k as shown in Fig.2.7(a). Here the tap-inputs are from a stationary stochastic process of mean zero and are optimized in the mean-squared sense according to the Wiener filter theory. For forward prediction, the desired response d_k is now x_k , the value to be estimated.

The forward prediction error represents the difference between the actual value of x_k and the predicted value $\hat{x}_k$ or

$$\epsilon_{N,k}^f = x_k - \hat{x}_k \quad (2.21)$$

where N refers to the predictor order.

Equation (2.21) can be rewritten as

$$\epsilon_{N,k}^f = x_k - \sum_{n=1}^N w_n^* x_{k-n} \quad (2.22)$$

where $w_1^*, w_2^*, \dots, w_N^*$ are the forward tap weights optimized in the mean-squared sense. Letting

$$a_{N,n} = \begin{cases} 1, & n=0 \\ -w_n^*, & n=1,2,\dots,N \end{cases} \quad (2.23)$$

the forward prediction error becomes

$$\epsilon_{N,k}^f = \sum_{n=0}^N a_{N,n} x_{k-n} \quad (2.24)$$

Figure 2.7 (a) Forward linear predictor; (b) prediction-error filter; (c) relationship between the predictor and the prediction-error filter.[9]

Figure 2.7(b) shows the forward prediction error filter based on this equation, and the relationship between the forward linear prediction filter and the forward prediction error filter is shown in Fig.2.7(c).

2.4.3 Normal Equation Under Forward Linear Prediction

To minimize the mean-squared error, the optimum tap-weight vector can be solved using the normal equation, Eq.(2.18). As before, the $N \times N$, correlation matrix of the tap inputs must be known. In this case the input vector is $[x_{k-1} \ x_{k-2} \ \cdots \ x_{k-N}]^T$ resulting in

$$\mathbf{R} = E[\mathbf{X}_{k-1} \mathbf{X}_{k-1}^T] \quad (2.25)$$

Since the input is stationary, the correlation matrix of the process is time invariant and Toeplitz in form. Thus

$$\begin{aligned} \mathbf{R} &= [\mathbf{X}_{k-1} \mathbf{X}_{k-1}^T] = E[\mathbf{X}_k \mathbf{X}_k^T] \\ &= \begin{bmatrix} r_0 & r_1 & \cdots & r_{N-1} \\ r_{-1} & r_0 & \cdots & r_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ r_{-N+1} & r_{-N+2} & \cdots & r_0 \end{bmatrix} \end{aligned} \quad (2.26)$$

where r_k is the autocorrelation function of the input process for lag k .

Another statistic that must be known is the cross correlation vector between the

tap-inputs and the desired response, x_k .

$$\begin{aligned} \mathbf{P} = E[\mathbf{X}_{k-1} x_k] &= \begin{bmatrix} E[x_{k-1} x_k] \\ E[x_{k-2} x_k] \\ \vdots \\ E[x_{k-N} x_k] \end{bmatrix} \\ &= \begin{bmatrix} r_{-1} \\ r_{-2} \\ \vdots \\ r_{-N} \end{bmatrix} \end{aligned} \quad (2.27)$$

The optimum tap-weights can now be solved using Eq.(2.19). Note that only the autocorrelation function of the input is required to solve this equation.

2.4.4 Backward Linear Prediction

For the backward linear predictor, data $x_k, x_{k-1}, x_{k-2}, \dots, x_{k-N+1}$, are linearly combined to predict data x_{k-N} i.e.,

$$\hat{x}_{k-N} = \sum_{n=1}^N g_n^* x_{k-n+1} \quad (2.28)$$

where $g_1^*, g_2^*, \dots, g_N^*$ are the backward tap weights optimized in the mean-squared sense.

The model for this type of predictor is shown in Fig.2.8(a).

The desired response in this case is x_{k-N} and the backward prediction error becomes

$$\epsilon_{N,k}^b = x_{k-N} - \hat{x}_{k-N} \quad (2.29)$$

To determine the backward prediction-error filter, Eq.(2.29) can be rewritten as

$$\epsilon_{N,k}^b = x_{k-N} - \sum_{n=1}^N g_n^* x_{k-n+1} \quad (2.30)$$

and letting

$$c_{N,k} = \begin{cases} -g_{n+1}^*, & n = 0, 1, 2, \dots, N-1 \\ 1, & n = N \end{cases} \quad (2.31)$$

Figure 2.8 (a) Backward linear predictor; (b) backward prediction-error filter; (c) backward prediction-error filter in terms of the forward prediction-error tap weights.[9]

Eq.(2.30) becomes

$$\epsilon_{N,k}^b = \sum_{n=0}^N c_{N,n} x_{k-n} \quad (2.32)$$

The model for this backward prediction-error filter is shown in Fig.2.8(b).

2.4.5 Normal Equation Under Backward Linear Prediction

As with the forward predictor, to solve the normal equation, two statistics that must be known are: the $N \times N$ correlation matrix of the tap inputs and the cross-correlation vector between the desired response and the tap inputs.

Since the tap inputs are $[x_k \ x_{k-1} \ \dots \ x_{k-N+1}]$, the correlation matrix is

$$\begin{aligned} \mathbf{R} &= E[\mathbf{X}_k \mathbf{X}_k^T] \\ &= \begin{bmatrix} r_0 & r_1 & \dots & r_{N-1} \\ r_{-1} & r_0 & \dots & r_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ r_{-N+1} & r_{-N+2} & \dots & r_0 \end{bmatrix} \end{aligned} \quad (2.33)$$

The cross-correlation vector becomes

$$\begin{aligned} \mathbf{P} &= E[\mathbf{X}_k x_{k-N}] = \begin{bmatrix} E[x_k x_{k-N}] \\ E[x_{k-1} x_{k-N}] \\ \vdots \\ E[x_{k-N+1} x_{k-N}] \end{bmatrix} \\ &= \begin{bmatrix} r_N \\ r_{N-1} \\ \vdots \\ r_1 \end{bmatrix} \end{aligned} \quad (2.34)$$

Equation (2.19) can now be used to solve for $\mathbf{G}$ where $\mathbf{G}$ replaces $\mathbf{W}$ in this equation.

2.5 Summary

In this chapter, the relevant background theory necessary to comprehend the interference excision algorithms were discussed. The use of direct-sequence spread spectrum signals allows a highly correlated interference signal to be removed by using the Wiener filter theory and linear prediction to estimate an a priori statistical information, namely the autocorrelation function, to design a filter which is optimal in the mean-squared sense.

Chapter 3

Interference Excision Algorithms

3.1 Introduction

Direct-sequence spread spectrum modulation is used to reduce the effects of narrow-band interference or intentional jamming. This is achieved by spreading the interference across the frequency band of the pseudo-noise signal thereby reducing it to a flat spectrum and thus lower the interference level. However, in certain situations this technique may not be sufficient enough to suppress the interference to an acceptable level. One method to improve the performance of a DS-SS communication system corrupted by narrowband interference is to filter the interference before despreading.

In this thesis, four algorithms will be looked at for estimating and suppressing the narrowband interference: the fast Fourier Transform (FFT), the Least-Mean-Square (LMS), the Recursive-Least-Square (RLS), and the Wiener-Hopf algorithms. These algorithms can be divided into two categories: nonparametric and parametric.

Nonparametric algorithms like the FFT require no a priori knowledge of the narrowband interference. In the case of the FFT algorithm, since the spectrum of a PN signal is relatively flat across the frequency band, narrowband interference can easily be recognized. By performing spectral analysis on the received signal, the interference can be estimated and a transversal filter designed.

Parametric algorithms, such as LMS, RLS, and Wiener-Hopf, are based on linear prediction for estimating the narrowband interference which can then be removed from

the received signal. Since the PN data is wideband, each signal sample or chip can be viewed as being independent of other samples. Thus, the signal sequence may be considered to be white noise. However, the interference samples, since the interference is a narrowband signal, are highly correlated with past and future interference samples. Therefore, when the prediction filter is designed and implemented, it is the interference signal that is estimated. In order to suppress the interference, this estimate is then subtracted from the received sampled signal.

In this chapter, a brief description of these four excision algorithms will be given.

3.2 Fast Fourier Transform

3.2.1 Introduction

The Fast Fourier Transform (FFT) is a nonparametric algorithm since no a priori statistical knowledge of the interference is required and is used to model the interference as a sum of weighted sinusoids. This algorithm, as described in [10] and [11], uses the fact that the power density spectrum of the PN sequence is relatively flat while the spectrum of the narrowband interference is highly peaked and can thus be easily recognized and estimated by using spectral analysis.

The FFT algorithm is of practical importance due to the ease of use and the existence of high speed DSP boards for calculating the FFT. The FFT algorithm will now be discussed.

3.2.2 FFT Algorithm

The first step of the FFT algorithm is to estimate the power spectral density for a

given block of received data. This can be accomplished by using the Welch method of power spectral estimation [12]. Once the power spectral density of the received signal has been calculated, the relative location of the highly peaked interference can easily be distinguished and a suppression filter designed at that location.

Since it is desired to have a filter with notches or zeroes at the frequency of the interference, a transversal filter is used. A simple method for designing the filter in the discrete time domain is to select its discrete Fourier transform to be the reciprocal of the square root of the input power spectral density at equally spaced frequencies. The discrete Fourier transform of the N tap coefficient vector or $\mathbf{W}_N$ is defined as

$$H_k = \sum_{n=0}^{N-1} w_n e^{-j\frac{2\pi}{N}nk}, \quad k = 0, 1, \dots, N-1. \quad (3.1)$$

In terms of the PSD of the received signal, H_k is given by

$$H_k = \frac{1}{\sqrt{P(\frac{k}{N}R_s)}} e^{-j\frac{2\pi}{N}(\frac{N-1}{2})k} \quad (3.2)$$

where $P(f)$, $0 \leq f \leq R_s$, denotes the estimate of the PSD and R_s is the sampling rate.

Note that $P(f)$ is periodic with period R_s and the folding frequency is $R_s/2$.

Since it is desirable to have a linear phase filter with an integer number of delays, the impulse response or tap vector $\mathbf{W}_N$ must be real, symmetric, and contain an odd number of taps. i.e.,

$$w_n = w_{N-1-n}, \quad n = 0, 1, \dots, N-1 \quad (3.3)$$

where N is an odd number

However, the Welch method using the FFT produces an even number of data, resulting in a rational number of delays, $(N-1)/2$. As a result, the data must be interpolated at evenly

spaced intervals to produce an odd number of taps to obtain an integer number of delays. Once the interpolated PSD for this block of data has been determined, Eq.(3.2) can be used and the tap co-efficients computed by taking the inverse discrete Fourier transform.

Once the tap coefficients have been calculated the transversal filter can be designed and the block of data filtered. This filtered data can then be applied to the PN correlator and then to the bit detector to obtain the transmitted data.

These seven steps, summarized in Table 3.1, form the basis for the FFT algorithm for interference suppression which can be done with relative ease and speed.

3.3 Least-Mean-Squares

3.3.1 Introduction

The least-mean-square (LMS) algorithm is important because of its simplicity and ease of computation since it does not require measurements of correlation matrix or any complex matrix operations such as matrix inversion.

The LMS algorithm is based on the method of steepest descent, and uses adaptive linear prediction to build a transversal filter to estimate the interference signal which can then be subtracted from the received signal. In this section, the LMS algorithm will be discussed with emphasis on linear prediction.

3.3.2 Derivation of the LMS Algorithm

From the steepest decent algorithm [9], the updated value of the tap-weight vector at time $k+1$ is computed by using the recursive relation

$$\mathbf{W}_{k+1} = \mathbf{W}_k - \frac{1}{2}\mu \left(\nabla E \left[e_{N,k}^2 \right] \right) \quad (3.7)$$

Step	Action
1	obtain a block of sampled data, r_n .
2	use the Welch algorithm to estimate the power spectrum for pre-defined sub-block size M , overlap size S , and window function.
3	<p>since the previous step produces</p> $P\left(\frac{k}{M}R_s\right), \quad k = 0, 1, \dots, (M - 1), \quad M \text{ even}$ <p>an even number of data, this set of data must be interpolated at a new set of frequencies to achieve an odd number of data. i.e.,</p> $P\left(\frac{k}{N}R_s\right), \quad k = 0, 1, \dots, (N - 1), \quad N \text{ odd}$ <p>A linear interpolation algorithm is used.</p>
4	<p>calculate the function</p> $\frac{1}{\sqrt{P\left(\frac{k}{N}R_s\right)}}, \quad k = 0, 1, \dots, (n - 1)$
5	using equation (3.2), calculate the DFT, H_k .
6	using the inverse DFT, calculate the filter coefficient vector W_k of the suppression filter.
7	apply the block of data to the filter.

Table 3.1 Summary of the fast Fourier transform algorithm for interference excision.

where $\mathbf{W}_{k+1}$ is the updated tap-weight vector, μ is the gain constant that regulates the speed and stability of adaptation, and ∇ is the gradient operator defined in Eq.(2.15). Recall from Eq.(2.16) that minimizing the mean-squared error $L(\mathbf{W})$ resulted in

$$\nabla E[\epsilon_{N,k}^f \epsilon_{N,k}^f] = 2\mathbf{R}\mathbf{W} - 2\mathbf{P}. \quad (3.8)$$

Substituting Eq.(3.8) into Eq.(3.7),

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu(\mathbf{P} - \mathbf{R}\mathbf{W}_k). \quad (3.9)$$

Practical implementation of this algorithm requires the use of the instantaneous estimates of $\mathbf{R}$ and $\mathbf{P}$ which are based on sample values of the tap-input vector and desired response. i.e.,

$$\hat{\mathbf{R}} = \mathbf{X}_k \mathbf{X}_k^T \quad (3.10)$$

$$\hat{\mathbf{P}} = \mathbf{X}_k d_k.$$

Substituting Eq.(3.10) into Eq.(3.9), collecting terms and using Eq.(2.7),

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu \mathbf{X}_k [d_k - \mathbf{X}_k^T \mathbf{W}_k] \quad (3.11)$$

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu \mathbf{X}_k \epsilon_k$$

Equation (3.11) is known as the least-mean-square (LMS) algorithm. The second term in this equation represents the correction that is applied to the current estimate $\mathbf{W}_k$ of the tap-weight vector.

3.3.3 Convergence of the LMS Algorithm

With the weight vector changing at each iteration, the convergence to the optimum weight vector $\mathbf{W}^*$ must be considered. Let $\Delta \mathbf{W}_k$ represent the weight error vector as defined by

$$\Delta \mathbf{W}_k = \mathbf{W}_k - \mathbf{W}^*. \quad (3.12)$$

Substituting this definition into the LMS algorithm, Eq.(3.11),

$$\begin{aligned}\Delta \mathbf{W}_{k+1} + \mathbf{W}^* &= \Delta \mathbf{W}_k + \mathbf{W}^* + \mu \mathbf{X}_k \left[d_k - \mathbf{X}_k^T (\Delta \mathbf{W}_k + \mathbf{W}^*) \right] \\ \Delta \mathbf{W}_{k+1} &= \Delta \mathbf{W}_k + \mu \mathbf{X}_k d_k - \mu \mathbf{X}_k \mathbf{X}_k^T \Delta \mathbf{W}_k - \mu \mathbf{X}_k \mathbf{X}_k^T \mathbf{W}^* \\ &= \left[\mathbf{I} - \mu \mathbf{X}_k \mathbf{X}_k^T \right] \Delta \mathbf{W}_k + \mu \left[\mathbf{X}_k d_k - \mathbf{X}_k \mathbf{X}_k^T \mathbf{W}^* \right].\end{aligned}\quad (3.13)$$

Since the weight vector $\mathbf{W}_k$ is a function of past input vectors only, i.e., $\mathbf{X}_{k-1}$, $\mathbf{W}_k$ is independent of $\mathbf{X}_k$. As a result, $\Delta \mathbf{W}_k$ is independent of $\mathbf{X}_k$. Taking the expected value of Eq.(3.13),

$$\begin{aligned}E[\Delta \mathbf{W}_{k+1}] &= E \left[\left(\mathbf{I} - \mu \mathbf{X}_k \mathbf{X}_k^T \right) \Delta \mathbf{W}_k \right] + \mu E \left[\mathbf{X}_k d_k - \mathbf{X}_k \mathbf{X}_k^T \mathbf{W}^* \right] \\ &= \left(\mathbf{I} - \mu E \left[\mathbf{X}_k \mathbf{X}_k^T \right] \right) E[\Delta \mathbf{W}_k] + \mu \left(E[\mathbf{X}_k d_k] - E \left[\mathbf{X}_k \mathbf{X}_k^T \right] \mathbf{W}^* \right) \\ &= (\mathbf{I} - \mu \mathbf{R}) E[\Delta \mathbf{W}_k] + \mu (\mathbf{P} - \mathbf{R} \mathbf{W}^*).\end{aligned}\quad (3.14)$$

The last term in the equation, $\mathbf{R} \mathbf{W}^*$ is equal to $\mathbf{P}$ from the normal equation, Eq.(2.18).

Therefore,

$$E[\Delta \mathbf{W}_{k+1}] = (\mathbf{I} - \mu \mathbf{R}) E[\Delta \mathbf{W}_k]. \quad (3.15)$$

This equation has the same form of the weight error vector for the steepest-descent algorithm [9] for which the weight vector converges to the optimum solution when the following condition is satisfied:

$$0 < \mu < \frac{2}{\lambda_{max}} \quad (3.16)$$

where λ_{max} is the largest eigenvalue of the correlation matrix $\mathbf{R}$. It can be assumed that this condition must also be met for the LMS algorithm to converge to the optimum weight vector. For transversal filters, this bound on μ was shown to be

$$0 < \mu < \frac{2}{\text{total input power}} \quad (3.17)$$

where

$$\text{total input power} = \text{number of taps} \times r_0 \quad (3.18)$$

The LMS algorithm will now be slightly modified to perform as an adaptive linear predictor filter for interference excision.

3.3.4 LMS Algorithm Based on Linear Prediction

A linear predictor is used to predict the next correlated data from a set of N previously estimated correlated data. Since the bandwidth of a direct-sequence spread spectrum signal is relatively wide, each signal sample may be considered independent of other samples and can therefore be viewed as AWGN. The narrowband interference, on the other hand is highly correlated with past and future samples. With this in mind, a linear prediction filter may be used to filter out the AWGN, (i.e., the spread signal) from the interference signal. Once the interference signal has been modelled, it may then be removed from the received signal. In this section, the LMS algorithm will be adapted so as to perform as a linear prediction filter.

From Chapter 2, the linear prediction filter uses the N previous data samples beginning at sample x_{k-1} and the desired signal $d_k = x_k$ to update the weight vector to the optimal Wiener solution. Converting Eq.(3.11),

$$\epsilon_k = x_k - \mathbf{W}_k^T \mathbf{X}_{k-1}, \quad (3.19)$$

and

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu \mathbf{X}_{k-1} \epsilon_k \quad (3.20)$$

which is the equation for the LMS algorithm under adaptive linear prediction for interference suppression. Equation (3.19) represents the spread signal, noise and residual interference because the second term, which is the prediction of the interference sample, is removed from the received sample. Figure 3.1 illustrates the block diagram for the LMS algorithm under linear prediction. The LMS algorithm for linear prediction is summarized in Table (3.2).

3.4 Recursive-Least-Squares

3.4.1 Introduction

Another useful algorithm which uses adaptive linear prediction is the recursive least squares or RLS algorithm. This algorithm uses information contained in all the input data to produce a different data-adaptive filter for each set of input data. By using this algorithm, the rate of convergence of the tap weight vector to the optimum Wiener solution is increased by an order of magnitude over the LMS algorithm. However, a disadvantage is the increase in complexity and computational power.

By using the shifting property of serialized data, the complexity and computational power of the RLS algorithm can be dramatically reduced. One implementation of the RLS algorithm is called the fast transversal filters or FTF, which uses a combination of up to four transversal filters to realize the true RLS solution.

In this section, the RLS algorithm and FTF implementation, as applied to interference suppression will be examined.

Figure 3.1 Adaptive linear predictor using LMS algorithm.

<i>Initialize :</i>	$\mathbf{W}_0 = \mathbf{0}$
<i>get tap – input vector at time $k - 1$:</i>	$\mathbf{X}_{k-1}$
<i>get the desired response :</i>	x_k
<i>compute :</i>	$\epsilon_k^* = x_k - \mathbf{W}_k^T \mathbf{X}_{k-1}$
	$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu \mathbf{X}_{k-1} \epsilon_k$
<i>where :</i>	$0 < \mu < \frac{2}{Nr_0}$

* ϵ_k represents the received signal with the interference filtered out.

Table 3.2 Summary of the LMS algorithm for linear prediction.

3.4.2 RLS algorithm

Since the RLS algorithm uses information in new data samples to update the estimates from previous data samples, the index of performance that is used is the sum weighted square error at the current time k , i.e.,

$$\xi_k = \sum_{i=1}^k \beta_{k,i} e_i^2 \quad (3.22)$$

where e_i is defined as

$$\begin{aligned} e_i &= d_i - \hat{d}_i \\ &= d_i - \mathbf{W}_k^T \mathbf{X}_i \end{aligned} \quad (3.23)$$

with the tap weights of the transversal filter fixed during the time $1 \leq i \leq k$. The term $\beta_{k,i}$ is known as the weighting factor and is used to give more weight to the information from present samples than the samples in the distant past since the statistics of the data can vary in a nonstationary environment. The weighting factor must have the following property:

$$0 < \beta_{k,i} \leq 1, \quad i = 1, 2, \dots, k. \quad (3.24)$$

As such, the most commonly used weighting factor is the exponential weighting factor defined as

$$\beta_{k,i} = \lambda^{k-i}, \quad i = 1, 2, \dots, k \quad (3.25)$$

where λ is a positive constant close to, but less than 1. Equation (3.22) becomes

$$\xi_k = \sum_{i=1}^k \lambda^{k-i} e_i^2 \quad (3.26)$$

This equation is similar to the mean-squared error discussed in the Wiener filter section in Chapter 2, (Eq.2.3) and so the minimum error can be found by using the normal equation, restated here as

$$\mathbf{R}_k \mathbf{W}_k = \mathbf{P}_k \quad (3.27)$$

where

$$\mathbf{R}_k = \sum_{i=1}^k \lambda^{k-i} \mathbf{X}_i \mathbf{X}_i^T \quad (3.28)$$

and

$$\mathbf{P}_k = \sum_{i=1}^k \lambda^{k-i} \mathbf{X}_i d_i. \quad (3.29)$$

The next step is to remove the summation from above the equations. To achieve this, the equations may be rewritten as

$$\mathbf{R}_k = \lambda \sum_{i=1}^{k-1} \lambda^{(k-1)-i} \mathbf{X}_i \mathbf{X}_i^T + \mathbf{X}_k \mathbf{X}_k^T \quad (3.30)$$

$$\mathbf{P}_k = \lambda \sum_{i=1}^{k-1} \lambda^{(k-1)-i} \mathbf{X}_i d_i + \mathbf{X}_k d_k. \quad (3.31)$$

The summation terms are just $\mathbf{R}_{k-1}$ and $\mathbf{P}_{k-1}$ respectively. Therefore, the equation may be written as the following recursive equations:

$$\mathbf{R}_k = \lambda \mathbf{R}_{k-1} + \mathbf{X}_k \mathbf{X}_k^T \quad (3.32)$$

and

$$\mathbf{P}_k = \lambda \mathbf{P}_{k-1} + \mathbf{X}_k d_k. \quad (3.33)$$

In order to calculate the optimal tap weight vector, the correlation matrix at time k , R_k must be inverted. This can be achieved by using the matrix inversion lemma¹ which results in

$$R_k^{-1} = \lambda^{-1} R_{k-1}^{-1} - \frac{\lambda^{-2} R_{k-1}^{-1} X_k X_k^T R_{k-1}^{-1}}{1 + \lambda^{-1} X_k R_{k-1}^{-1} X_k} \quad (3.34)$$

Letting

$$\Phi_k = R_k^{-1} \quad (3.35)$$

and

$$k_k = \frac{\lambda^{-1} \Phi_{k-1} X_k}{1 + \lambda^{-1} X_k^T \Phi_{k-1} X_k} \quad (3.36)$$

Eq.(3.34) becomes

$$\Phi_k = \lambda^{-1} \Phi_{k-1} - \lambda^{-1} k_k X_k^T \Phi_{k-1}. \quad (3.37)$$

Using the above equation, the optimal weight vector can now be solved by

$$\begin{aligned} W_k &= R_k^{-1} P_k \\ &= \Phi_k P_k \\ &= \lambda \Phi_k P_{k-1} + \Phi_k X_k d_k. \end{aligned} \quad (3.38)$$

Substituting Eq.(3.37) into the first term on the right side of Eq.(3.38),

$$\begin{aligned} W_k &= \Phi_{k-1} P_{k-1} - k_k X_k^T \Phi_{k-1} P_{k-1} + \Phi_k X_k d_k \\ &= R_{k-1}^{-1} P_{k-1} - k_k X_k^T R_{k-1}^{-1} P_{k-1} + \Phi_k X_k d_k \\ &= W_{k-1} - k_k X_k^T W_{k-1} + \Phi_k X_k d_k \end{aligned} \quad (3.39)$$

¹ See Appendix A.

By rearranging Eq.(3.36), it can be shown that

$$\mathbf{k}_k = \Phi_k^T \mathbf{X}_k \quad (3.40)$$

and by substituting this into the previous equation

$$\begin{aligned} \mathbf{W}_k &= \mathbf{W}_{k-1} - \mathbf{k}_k \mathbf{X}_k^T \mathbf{W}_{k-1} + \mathbf{k}_k d_k \\ &= \mathbf{W}_{k-1} + \mathbf{k}_k \epsilon_k^{fp} \end{aligned} \quad (3.41)$$

where

$$\epsilon_k^{fp} = d_k - \mathbf{X}_k^T \mathbf{W}_{k-1} \quad (3.42)$$

and is referred to as the a priori estimation error since it is based on the old least squares estimate of the tap weight vector.

Equations (3.36), (3.42), (3.41), and (3.37) constitute the RLS algorithm and is summarized in Table 3.3.

3.4.3 RLS Algorithm Based on Linear Prediction

For interference suppression, adaptive linear prediction will be used in the same manner as the LMS algorithm. Therefore, the RLS algorithm must be modified slightly to work as a predictor. It was shown in Chapter 2 that for linear prediction, the desired sample d_k becomes x_k , and the algorithm is based on the previous N data samples, i.e., $\mathbf{X}_{k-1}$ where $\mathbf{X}_{k-1} = [x_{k-1} \ x_{k-2} \dots x_{k-N}]^T$ and N is the filter order. Implementing this into the RLS algorithm, the new set of equations become:

Initialize : $\Phi_0 = \delta^{-1} \mathbf{I}$, $\delta = \text{small positive constant}$

$$\mathbf{W}_0 = \mathbf{0}$$

Compute : $\mathbf{k}_k = \frac{\lambda^{-1} \Phi_{k-1} \mathbf{X}_k}{1 + \lambda^{-1} \mathbf{X}_k^T \Phi_{k-1} \mathbf{X}_k}$

$$\epsilon_k^{fp} = d_k - \mathbf{X}_k^T \mathbf{W}_{k-1}$$

$$\mathbf{W}_k = \mathbf{W}_{k-1} + \mathbf{k}_k \epsilon_k^{fp}$$

$$\Phi_k = \lambda^{-1} \Phi_{k-1} - \lambda^{-1} \mathbf{k}_k \mathbf{X}_k^T \Phi_{k-1}$$

Table 3.3 Summary of the RLS algorithm.

Initialize Variables

$$\Phi_0 = \delta^{-1} \mathbf{I}, \quad \delta = \text{a small positive constant}$$

$$\mathbf{W}_0 = \mathbf{0}$$

Compute

$$\mathbf{k}_k = \frac{\lambda^{-1} \Phi_{k-1} \mathbf{X}_{k-1}}{1 + \lambda^{-1} \mathbf{X}_{k-1}^T \Phi_{k-1} \mathbf{X}_{k-1}} \quad (3.44)$$

$$\epsilon_k^{fp} = x_k - \mathbf{X}_{k-1}^T \mathbf{W}_{k-1}$$

$$\mathbf{W}_k = \mathbf{W}_{k-1} + \mathbf{k}_k \epsilon_k^{fp}$$

$$\Phi_k = \lambda^{-1} \Phi_{k-1} - \lambda^{-1} \mathbf{k}_k \mathbf{X}_{k-1}^T \Phi_{k-1}$$

where ϵ_k^{fp} represents the forward a priori prediction error and is the received signal with the interference filtered out. Figure 3.2 illustrates a simplified signal flow graph of the RLS algorithm under linear prediction.

3.4.4 Fast Transversal Filters Algorithm

Although the RLS algorithm converges quite rapidly to the optimal solution, this is achieved at the expense of processing power and speed. Specifically, N^2 arithmetic operations per iteration, where N is the number of taps, are required for the RLS algorithm while the LMS algorithm only requires N arithmetic operations per iteration.

An algorithm which uses the shifting property of serialized input data has been previously developed which reduces the number of computations similar to the LMS algorithm, while still realizing the true RLS solution. This algorithm is called the fast transversal filters algorithm or FTF. For interference suppression, this algorithm uses three transversal filters with similar inputs to perform individual parts of the algorithm as

Figure 3.2 Simplified flow graph of the RLS algorithm as a linear predictor.

illustrated in Fig.3.3. Each filter performs the following two processes: filtering, where an error signal is produced as a result of a new data input, and adapting, where the tap-weight vector of that filter is updated.

Filter I is used to perform adaptive forward linear prediction on the input data to produce the following outputs: the forward a priori prediction error ϵ_k^{fp} , the forward a posteriori prediction error ϵ_k^f , and the sum of weighted forward prediction-error squares ξ_k^f . Filter II is used to perform adaptive backward linear prediction on the same input data to produce a corresponding set of outputs: the backward a priori prediction error ϵ_k^{bp} , the backward a posteriori prediction error ϵ_k^b , and the sum of weighted backward prediction-error squares ξ_k^b . Filter III is used to define and update the gain vector $k_{N,k-1}$ to produce the estimation error γ_k .

The FTF algorithm will now be given. The derivation of this algorithm is given in [9].

For adaptive forward linear prediction, the forward a priori prediction error may be denoted by

$$\epsilon_k^{fp} = x_k - \mathbf{W}_{k-1}^T \mathbf{X}_{N,k-1} \quad (3.45)$$

where N is the predictor order. By letting

$$\begin{aligned} \mathbf{a}_{N,k-1} &= \begin{bmatrix} 1 \\ -\mathbf{W}_{k-1} \end{bmatrix}, \quad \text{and} \\ \mathbf{X}_{N+1,k} &= \begin{bmatrix} x_k \\ \mathbf{X}_{N,k-1} \end{bmatrix} \end{aligned} \quad (3.46)$$

where $\mathbf{a}_{N,k-1}$ represent the tap-weight vector of the prediction error filter, Fig.2.7(b), the forward a priori prediction error can be rewritten as

$$\epsilon_{N,k}^{fp} = \mathbf{a}_{N,k-1}^T \mathbf{X}_{N+1,k}. \quad (3.47)$$

Figure 3.3 Fast transversal filters to realize RLS solution.

The forward a posteriori prediction error may be computed by multiplying the forward a priori prediction error by a delayed estimation error, or

$$\epsilon_{N,k}^f = \epsilon_{N,k}^{fp} \gamma_{N,k-1} \quad (3.48)$$

and the minimum value of the sum of weighted forward prediction-error squares updated by the following recursive equation

$$\xi_{N,k}^f = \lambda \xi_{N,k-1}^f + \epsilon_{N,k}^{fp} \epsilon_{N,k}^f. \quad (3.49)$$

The a posteriori estimation error is then updated by

$$\gamma_{N+1,k} = \lambda \frac{\xi_{N,k-1}^f}{\xi_{N,k}^f} \gamma_{N,k-1} \quad (3.50)$$

and the gain vector by

$$\mathbf{k}_{N+1,k} = \begin{bmatrix} 0 \\ \mathbf{k}_{N,k-1} \end{bmatrix} + \lambda^{-1} \frac{\epsilon_{N,k}^{fp}}{\xi_{N,k-1}^f} \mathbf{a}_{N,k-1}. \quad (3.51)$$

The tap-weight vector of the forward prediction error filter is now updated by

$$\mathbf{a}_{N,k} = \mathbf{a}_{N,k-1} - \epsilon_{N,k}^f \begin{bmatrix} 0 \\ \mathbf{k}_{N,k-1} \end{bmatrix}. \quad (3.52)$$

Up to this point of the algorithm, both the forward prediction-error filter and the gain vector filter has been updated. What is now required is to update the backward prediction-error filter. To begin with, let $k_{N+1,k}$ denote the last element of the gain vector. The backward a priori prediction error can be calculated by

$$\epsilon_{N,k}^{bp} = \lambda \xi_{N,k-1}^b k_{N+1,k}. \quad (3.53)$$

By using Eq.(3.50), the filter order has increased by one. To correct this, the current value of the estimation error is now calculated.

$$\gamma_{N,k} = \left[1 - \epsilon_{N,k}^{bp} \gamma_{N+1,k} k_{N+1,k} \right]^{-1} \gamma_{N+1,k} \quad (3.54)$$

This equation can now be used to calculate the backward a posteriori prediction error

$$\epsilon_{N,k}^b = \gamma_{N,k} \epsilon_{N,k}^{bp} \quad (3.55)$$

and the minimum value of the sum of weighted backward a posteriori prediction-error squares updated by the following recursive equation

$$\xi_{N,k}^b = \xi_{N,k-1}^b + \epsilon_{N,k}^{bp} \epsilon_{N,k}^b. \quad (3.56)$$

The N order gain vector can now be calculated by

$$\begin{bmatrix} k_{N,k} \\ 0 \end{bmatrix} = k_{N+1,k} - k_{N+1,k} c_{N,k-1} \quad (3.57)$$

where the tap-weight vector of the backward prediction-error filter, $c_{N,k}$ is given by

$$c_{N,k} = c_{N,k-1} - \epsilon_{N,k}^b \begin{bmatrix} k_{N,k} \\ 0 \end{bmatrix}. \quad (3.58)$$

Equations (3.47) – (3.58) constitutes the FTF algorithm which realizes the RLS solution at a more optimal computational power. Note that the forward a priori prediction error $\epsilon_{N,k}^{fp}$ represents the filtered data.

When the FTF algorithm is implemented on a digital computer or DSP board with a limited precision, it was previously determined by the algorithm's developers that finite-precision errors will occur causing a certain positive variable to become negative. This

variable, $rv = 1 - \epsilon_{N,k}^{bp} \gamma_{N+1,k} k_{N+1,k}$, is called the rescue variable. When this variable becomes negative, the FTF algorithm must be restarted.

For the rest of the thesis, the FTF algorithm will be known as the RLS algorithm for interference excision. The initialization and main routine for this algorithm is summarized in Tables 3.4 and 3.5 respectively.

3.5 Wiener-Hopf

3.5.1 Introduction

The Wiener-Hopf algorithm for interference suppression is based on the discrete form of the Wiener-Hopf equation, more commonly known as the normal equation. By using the normal equation under linear prediction for a block of data, the optimal weight vector can be obtained for this set of data and the interference suppressed data can be obtained.

3.5.2 Wiener-Hopf Algorithm

It was shown in Chapter 2 that to minimize the mean-squared-error, the optimum tap weight vector can be solved using the normal equation. Under forward linear prediction, this equation becomes

$$\mathbf{W}^* = \mathbf{R}^{-1} \mathbf{P}, \quad \text{where}$$

$$\begin{aligned} \mathbf{R} &= E \left[\mathbf{X}_{k-1} \mathbf{X}_{k-1}^T \right] \\ &= \begin{bmatrix} r_0 & r_1 & \cdots & r_{N-1} \\ r_{-1} & r_0 & \cdots & r_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ r_{-N+1} & r_{-N+2} & \cdots & r_0 \end{bmatrix} \end{aligned} \quad (3.61)$$

for $k = 1$,

$$\mathbf{a}_{0,1} = \mathbf{c}_{0,1} = 1$$

$$\mathbf{k}_{0,1} = [] \text{ (zero dimension)}$$

$$\gamma_{0,1} = 1$$

$$\xi_{0,1}^f = |x_1|^2$$

for $2 \leq k \leq N + 1$,

$$\epsilon_{k-2,k}^{fp} = \mathbf{a}_{k-2,k-1}^T \mathbf{X}_{k-1,k}$$

$$\mathbf{a}_{k-1,k} = \begin{bmatrix} \mathbf{a}_{k-2,k-1} \\ \frac{-\epsilon_{k-2,k}^{fp}}{x_1} \end{bmatrix}$$

$$\epsilon_{k-2,k}^f = \gamma_{k-2,k-1} \epsilon_{k-2,k}^{fp}$$

$$\xi_{k-1,k}^f = \lambda \xi_{k-2,k-1}^f$$

$$\xi_{k-2,k}^f = \xi_{k-1,k}^f + \epsilon_{k-2,k}^{fp} \epsilon_{k-2,k}^f$$

$$\gamma_{k-1,k} = \frac{\xi_{k-1,k}^f}{\xi_{k-2,k}^f} \gamma_{k-2,k-1}$$

$$\mathbf{k}_{k-1,k} = \begin{bmatrix} 0 \\ \mathbf{k}_{k-2,k-1} \end{bmatrix} + \frac{\epsilon_{k-2,k}^{fp}}{\xi_{k-1,k}^f} \mathbf{a}_{k-2,k-1}$$

$$\mathbf{c}_{k-1,k} = \begin{bmatrix} -x_1 \gamma_{k-1,k} \mathbf{k}_{k-1,k} \\ 1 \end{bmatrix}, \text{ compute only when } k = N + 1$$

$$\xi_{k-1,k}^b = \gamma_{k-1,k} |x_1|^2, \text{ compute only when } k = N + 1$$

Table 3.4 Initialization of the RLS algorithm.

$$\begin{aligned}
\epsilon_{N,k}^{fp} &= \mathbf{a}_{N,k-1}^T \mathbf{X}_{N+1,k} \\
\epsilon_{N,k}^f &= \gamma_{N,k-1} \epsilon_{N,k}^{fp} \\
\xi_{N,k}^f &= \lambda \xi_{N,k-1}^f + \epsilon_{N,k}^{fp} \epsilon_{N,k}^f \\
\gamma_{N+1,k} &= \lambda \frac{\xi_{N,k-1}^f}{\xi_{N,k}^f} \gamma_{N,k-1} \\
\mathbf{k}_{N+1,k} &= \begin{bmatrix} 0 \\ \mathbf{k}_{N,k-1} \end{bmatrix} + \frac{\epsilon_{N,k}^{fp}}{\lambda \xi_{N,k-1}^f} \mathbf{a}_{N,k-1} \\
\mathbf{a}_{N,k} &= \mathbf{a}_{N,k-1} - \epsilon_{N,k}^f \begin{bmatrix} 0 \\ \mathbf{k}_{N,k-1} \end{bmatrix} \\
\epsilon_{N,k}^{bp} &= \lambda \xi_{N,k-1}^b k_{N+1,k} \\
\gamma_{N,k} &= \left[1 - \epsilon_{N,k}^{bp} \gamma_{N+1,k} k_{N+1,k} \right]^{-1} \gamma_{N+1,k} \\
rv^* &= \left[1 - \epsilon_{N,k}^{bp} \gamma_{N+1,k} k_{N+1,k} \right] \\
\epsilon_{N,k}^b &= \gamma_{N,k} \epsilon_{N,k}^{bf} \\
\xi_{N,k}^b &= \lambda \xi_{N,k-1}^b + \epsilon_{N,k}^{bf} \epsilon_{N,k}^b \\
\begin{bmatrix} \mathbf{k}_{N,k} \\ 0 \end{bmatrix} &= \mathbf{k}_{N+1,k} - k_{N+1,k} \mathbf{c}_{N,k-1} \\
\mathbf{c}_{N,k} &= \mathbf{c}_{N,k-1} - \epsilon_{N,k}^b \begin{bmatrix} \mathbf{k}_{N,k} \\ 0 \end{bmatrix}
\end{aligned}$$

* If rv is negative, the algorithm must be restarted.

Table 3.5 Summary of the main RLS algorithm.

and

$$\begin{aligned} \mathbf{P} &= E[\mathbf{X}_{k-1}\mathbf{X}_k] \\ &= \begin{bmatrix} r_{-1} \\ r_{-2} \\ \vdots \\ r_{-N} \end{bmatrix} \end{aligned} \quad (3.62)$$

where N is the number of taps.

To apply this equation to interference suppression, an M block of data is used. Once the optimum weight vector for this block has been calculated, the data can be filtered producing a new set of data which represents an estimate of the interference. This new set of data can then be subtracted from the original block of data to suppress the interference.

This algorithm is computationally expensive since inversion of an N -by- N matrix is required. Therefore, it may not be too practical in a real-time environment. However, it does allow a basis to compare the other algorithms with in terms of minimizing the mean-squared-error since this represents the optimal solution.

3.5.3 Summary

In this chapter, four algorithms for estimating and suppressing a stationary narrow-band interference were presented. The Fast Fourier Transform or FFT is a nonparametric algorithm which require no a priori knowledge of the interference signal. This algorithm uses spectral analysis to determine the relative location of the interference tone and then designs a transversal filter at this location.

The other three algorithms, the Least-Mean-Square, the Recursive-Least-Square, and the Wiener-Hopf, are considered parametric algorithms which relies on linear prediction to estimate the narrowband interference. Once this signal has been modelled, it can then be removed from the received signal.

In the next chapter, simulations to determine the effectiveness of these algorithms will be performed. These simulations will involve the transmission of a BPSK spread spectrum signal in a hostile environment with these algorithms implemented at the receiver.

Chapter 4

Simulation Results

4.1 Introduction

In order to verify the validity of these algorithms, simulations were generated using MATLABTM and C language. In this chapter, the communication model that was used for the simulations will be described, and the simulation results presented. The last section will compare the different algorithms and discuss the advantages and disadvantages of each.

4.2 DS-SS Communications Model

Figure 4.1 illustrates the baseband direct-sequence BPSK spread spectrum communication system that will be used for simulation of the interference excision algorithms.

The data sequence $D(t-kT_b)$ consists of rectangular pulses of amplitude ± 1 and period T_b . This data sequence is spread by a pseudo-random noise sequence $PN(t-nT_c)$, which is also rectangular in nature with an amplitude of ± 1 and period T_c , where $T_c \ll T_b$. This spread signal $s(t)$ is transmitted over a channel which consists of AWGN with two-sided power spectral density (PSD) equal to $N_o/2$ and a narrowband interference tone $i(t)$.

At the receiver of the communication model, the received signal $y(t)$ is applied to a matched filter producing sampled data at the chip rate. i.e.,

$$r_k = s_k + n_k + i_k \quad (4.1)$$

where s_k is the signal amplitude, n_k is AWGN of variance $N_o/2$, and i_k is a sample of the interference. The N past samples are then applied to the excision algorithm to produce the

Figure 4.1 Baseband PN spread spectrum communications model.[10]

tap-weight vectors for the transversal filter. With the majority of the interference samples suppressed from the received samples, the samples are applied to the PN correlator, assuming perfect tracking and synchronization, to despread the signal. This signal is then applied to a bit detector to retrieve the data.

4.3 Simulation Results

4.3.1 Simulation Parameters

In order to make a fair comparison between the different interference excision algorithms, the same set of data will be generated for each algorithm. Table 4.1 lists the parameters that will be used to generate the sampled data at the receiver.

Figure 4.2 illustrates a sample (64 bits) of the BPSK data that is to be transmitted along with the corresponding power spectral density. Note that the PSD has the form of the sinc function with a bandwidth of approximately $2/32$ Hz or twice the bit rate T_b . Figure 4.3 illustrates a sample (256 chips) of the BPSK pseudo-noise spreading sequence and its corresponding PSD. For these simulations, an infinite PN sequence period is used. Note how the power spectrum has a noise like appearance. This is the result of sampling the signal at the chip rate. If the signal was sampled at a higher rate, the power spectrum would be similar to Fig.4.2 with a bandwidth of twice the chip rate and the null points occur at multiples of the chip rate. Figure 4.4 shows a sample (8 bits) of the spread spectrum signal to be transmitted along with its PSD. The spread signal has converted the data signal to a faster rate and whose PSD is relatively flat and wide, similar to the PSD of the PN sequence. This signal is then transmitted to the receiver to be despread and decoded.

In the communication model that is used, the transmitted signal is subjected to a narrowband interference signal and the ever present additive white Gaussian noise. The additive noise was generated using a random number generator with a Gaussian distribution and variance of 1, so that $E_c/N_o = -3.05$ dB, or $E_b/N_o = 12$ dB for this example. A sample of this noise $n(t)$ and its PSD is illustrated in Fig.4.5. The interference is a single tone located at a frequency of $f_i = 2/16$ Hz and has an interference power to spread signal power of 20 dB. This interference signal and its power spectral density are shown in Fig.4.6.

The sampled data at the output of the Matched Filter consists of the spread signal, noise, and interference. This data r_k and its PSD is illustrated in Fig.4.7. From the PSD plot, the interference signal is highly visible in comparison with the spread spectrum modulated signal. This data, sampled at the chip rate, is then applied to the interference excision algorithms discussed in the previous section. The excised signal is then applied to the PN correlator to despread the signal, assuming perfect synchronization, and then applied to a bit detector.

The effect of this interference tone is illustrated in Fig.4.8. When the received transmission is despread and then decoded, a high rate of bit errors have occurred. In the next few sections, each excision algorithm will be used on simulated data generated using this model to suppress this tone. Several different conditions will be tried and the results illustrated.

Parameter	Value
Data Amplitude, $D(t)$	± 1
PN Amplitude, $PN(t)$	± 1
Bit Rate, R_b	1/32 Hz
Chip Rate R_c	1 Hz
Sampling Rate, f_s	1 Hz

Table 4.1 Parameters to generate data.

Figure 4.2 A sample of the data bits $D(t)$ and its corresponding power spectral estimation.

Figure 4.3 A sample of the pseudo-noise sequence $PN(t)$ used to spread the signal and its corresponding power spectral estimation.

Figure 4.4 A sample of the spread spectrum modulated signal $s(t)$ and its power spectral estimation at the output of the Matched filter sampled at the chip rate of 1 Hz.

Figure 4.5 A sample of the noise $n(t)$ and its corresponding power spectral estimation.

Figure 4.6 A sample of the interference signal $i(t)$ and its corresponding power spectral estimation.

Figure 4.7 A sample of the received signal r_k at the output of the Matched filter with noise and interference added to the spread signal and its corresponding power spectral estimation. For this case, $P_I/P_S = 20$ dB and $E_b/N_0 = 12$ dB.

Figure 4.8 A sample of the transmitted data D_k and the corresponding decoded data $\hat{D}_k$ with no interference suppression algorithm, $P_I/P_S = 20$ dB and $E_b/N_o = 12$ dB.

4.3.2 FFT Algorithm

In this section, the FFT algorithm as described in Section 3.2.2 and presented in Table 3.1 will be used to suppress the interference for the above signal. For this algorithm, the parameters used are given in Table 4.2 with each step of the algorithm to be illustrated.

Figure 4.9 shows the PSD of the received samples r_k calculated by averaging 64 blocks of 16-point FFTs with no overlap and using a rectangular window. The conditions of this test situation are as follows:

$$\begin{aligned} E_b/N_o &= 12 \text{ dB} \\ P_I/P_S &= 20 \text{ dB.} \end{aligned} \tag{4.2}$$

The top graph illustrates what little effect the addition of signal and noise (bottom graph) has on the interference due to the high interference to signal power.

The next step in the algorithm is to interpolate the power spectrum from M-even points to N-odd points. For this simulation, interpolation from 16-points to 15-points is required. This was done by applying the power spectrum to a linear interpolation algorithm. The result is shown in Fig.4.10. Figures 4.11 and 4.12 illustrates the square root and inverse square root of this interpolated power spectrum respectively. Note the low spectral lines near the interference frequency, i.e., 0.125 Hz.

Equation 3.2 can now be used to calculate the magnitude and phase of H_k , the sampled transfer function of the transversal filter. The result is shown in Fig.4.13. From this figure it can be seen that the phase is linear, which is desired in order to prevent any phase distortion on the signal.

The tap weight vectors or impulse response w_k can now be calculated by taking the inverse discrete Fourier transform of the transfer function H_k . The normalized impulse response is used to keep the filter gain to unity. The result of the tap weight vectors is illustrated in Fig.4.14.

Figure 4.15 illustrates the magnitude of the continuous transfer function of the suppression filter $H(e^{j\omega})$ using the normalized set of tap weights. Note the notch of the filter is located at the interference frequency, i.e., 0.125 Hz, and thus a suppression of interference is expected, as shown in the lower part of Fig.4.16. The attenuation for this simulation is approximately 20 dB.

Finally, Fig.4.17 illustrates the result of the excision algorithm. The upper plot represents 64 bits of the transmitted data while the middle plot represents the demodulated data after the received data has passed through the suppression filter. Notice that there are no errors for this set of bits.

The effects of changing algorithm parameters, i.e. number of taps, and interference frequency will be studied next. In order to increase the number of taps, the FFT size must be increased as well. For this simulation, a 31-tap and 32-point FFT will be used. Figure 4.18 illustrates the magnitude of the continuous transfer function $|H(e^{j\omega})|$ for this set of parameters. Note that it has the same form as the 15-tap filter illustrated in Fig.4.15. Figure 4.19 shows the transmitted and decoded set of data. Notice again that this set of data (64 bits) is decoded properly. Figure 4.20 illustrates the BER curve for three number of taps, 7, 15, and 31 with the interference located at 0.125 Hz. From this graph, it can be seen that there is hardly any difference in terms of performance while

the number of computations required increases as the number of filter taps increases. In order to determine a suitable number of taps, another factor that must be looked at is the interference frequency.

Although, the three tap sizes chosen above gave similar performance results, this similarity can be easily explained by the choice of the interference frequency. Since 0.125 Hz falls on a spectral frequency for an FFT size of 8, 16, and 32 points, the algorithm produces a similar notch filter for each, resulting in similar BER results. Figure 4.21 illustrates the resulting continuous transfer function of the excision filter, $H(e^{j\omega})$, using a 15-tap filter, for three frequencies located at a spectrally related frequency (0.125 Hz, 0.250 Hz, and 0.375 Hz), and three frequencies not located at a spectrally related frequency (0.111 Hz, 0.2 Hz, and 0.350 Hz). Therefore, when the interference lies on a spectrally related frequency, the suppression filter is more defined and can filter out the interference more efficiently. Figure 4.22 illustrates the BER curve for the three different number of tap filters when the interference does not fall on a spectrally related frequency. In this case, the 7-tap filter has little effect in suppressing the interference, while the 15 and 31-tap filters still maintain an adequate and similar performance levels.

From these simulations, it was shown that a 15-tap filter was adequate in both performance and computational complexity to suppress the interference tone. Later in this chapter, this algorithm will be compared with the other algorithms in terms of speed and performance. The next two chapters will outline an experimental setup and the results of implementing the FFT algorithm with a 15-tap filter on this system.

Parameter	Value
Block Size	1024
Number of Taps	15
Number of FFTs to Average	64
Size of the FFT	16
Overlap	0
Window Type	Rectangular

Table 4.2 Parameters for the FFT algorithm.

Figure 4.9 Power spectral estimation of received signal r_k based on parameters listed in Table 4.2. The lower graph is a magnified version of the upper graph to illustrate the small contribution of the SS signal and noise.

Figure 4.10 Interpolated power spectral estimation of Fig.4.9 from 16 points to 15 points.

Figure 4.11 Square root of the power spectral estimation of Fig.4.10.

Figure 4.12 Inverse square root of the power spectral estimation of Fig.4 10.

Figure 4.13 The magnitude and phase of the transfer function H_k of the interference suppression filter.

Figure 4.14 The normalized impulse response or tap vector w_k of the linear suppression filter H_k .

Figure 4.15 The continuous spectrum, $H(e^{j\omega})$ of the interference suppression filter with impulse response w_k for a 15-tap filter.

Figure 4.16 The power spectral estimation of the sampled signal before the suppression filter (upper plot) and the power spectral estimation after the suppression filter (bottom plot).

Figure 4.17 The transmitted data sequence D_k (64 bits) and the denodulated data $\hat{D}_k$ after the 15-tap suppression filter.

Figure 4.18 The continuous spectrum, $H(e^{j\omega})$ of the interference suppression filter with impulse response w_k for a 31-tap filter.

2 of/de 2

PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

PRECISIONSM RESOLUTION TARGETS

Figure 4.19 The transmitted data sequence D_k (64 bits) and the demodulated data $\hat{D}_k$ after the 31-tap suppression filter.

Figure 4.20 Bit error rate performance for a single tone interferer with $P_I/P_S = 20\text{dB}$ at frequency 0.125 Hz for 7, 15, and 31 tap filters.

Figure 4.21 The suppression filters for interference located at frequencies 0.111 Hz, 0.125 Hz, 0.2 Hz, 0.25 Hz, 0.35 Hz, and 0.375 Hz.

Figure 4.22 Bit error rate performance for a single tone interferer with $P_I/P_S = 20\text{dB}$ at frequency 0.111 Hz for 7, 15, and 31 tap filters.

4.3.3 LMS Algorithm

In this section, the least-mean-squares algorithm will be used to estimate the filter parameter of the excision filter. The same parameters given in Table 4.1 were used to generate the received signal with the interference. Using Eq.(3.17), the value of μ was calculated to be between 0 and 0.0013, i.e., $0 < \mu < 0.0013$. For this simulation, μ was chosen to be 0.0001. The parameters used for the LMS algorithm is given are Table 4.3.

Figure 4.23 illustrates the convergence of the tap weight vector from an initial value of 0 to the optimal solution as given by the Wiener solution. The three graphs (top left, top right, and bottom left) show the first three weights as a function of iteration number while the bottom right graph shows a 3-D plot of the first three weights to illustrate how rapid these weights converge.

Figure 4.24 illustrates the squared error function. This error represents the difference between the received sampled signal r_k , which consists of the SS signal, noise and interference, and the predicted interference tone. Again one can see the rapid speed of convergence and relatively accurate results.

The suppression of the interference tone is shown next in Fig.4.25. The top graph illustrates the power spectrum of the received signal with the interference while the bottom graph illustrates the power spectrum of the filtered data. Most of the interference has been suppressed, approximately 40 dB of attenuation, from the received data. The slight distortion caused by this algorithm should be easily removed when despreading is done. Figure 4.26 illustrates the despread and decoded result of the filtered data with

this set of data (64 bits) containing no errors.

The bit error rate curve for the LMS algorithm is shown in Fig.4.27. In this graph, three different taps sizes, 7, 15 and 31, with $\mu=0.0001$ were used. From this plot, the LMS algorithm appears to be a relatively efficient interference suppression algorithm with little dependence on the tap size. Next, the parameter μ will be looked at more closely.

Figure 4.28 illustrates the effect of changing the value of μ . As this parameter is increased, the squared error decreases more rapidly. However, when this value exceeds the bounds, approximately 0.0013 in this case, the square error will never decrease resulting in severe system degradation. Finally, Fig.4.29 illustrates the bit error rate curve for three different values of μ . The one outside the bound, $\mu=0.01$, has no effect in suppressing the interference while the one near the bound, $\mu=0.001$, has some success. The best results occur within the boundary.

This algorithm has shown to be very effective in suppressing the intentional jammer in simulations. This result will be compared with the other algorithms later in this chapter and verified experimentally in the later chapters.

Parameter	Value
Data Size	2048
Number of Taps	15
μ	0.0001
Initial Condition	$\mathbf{W}_0 = \mathbf{0}$

Table 4.3 Parameters for the LMS algorithm.

Figure 4.23 Convergence of the first three weight tap vector from an initial starting point of $[0,0,0]$.

Figure 4.24 Plot of the squared error function with $\mu = 0.0001$.

Figure 4.25 Power spectral estimation of the received sampled data (top graph) and the power spectral estimation of the data with the LMS algorithm (bottom graph) with $\mu = 0.0001$.

Figure 4.27 Bit error rate curve for 7, 15, and 31 tap vectors for the LMS algorithm with $\mu=0.0001$.

Figure 4.28 Squared error plot as a function of iterations k , and step size μ .

Figure 4.29 Bit error rate curve for values of $\mu = 0.01$, 0.001 , and 0.0001 .

4.3.4 RLS Algorithm

In this section, the recursive-least-squares algorithm will be used to suppress the same received signal with interference and noise as in the previous two algorithms. The parameters used for this simulation are given in Table 4.4.

Figure 4.30 illustrates the squared error function for the RLS algorithm. From this plot, the squared error converges quite rapidly to the optimal solution. Comparing this result with the squared error function of the LMS algorithm, in Fig.4.24, one can easily see that this algorithm converges much more quickly with approximately the same order of computations required for each. Another advantage that will be seen later on is the BER of this algorithm in comparison to the others.

Figure 4.31 shows the result of this algorithm in terms of the power spectrum. The top graph illustrates the PSD received data which contains the interference while the bottom graph illustrates the PSD of the filtered data. In this graph, the interference has been mostly suppressed with approximately 40 dB of attenuation. As with the case of the LMS algorithm, the slight distortion on the signal will be removed when the despreading process begins. Figure 4.32 shows the transmitted data (top graph) and the decoded suppressed data (middle graph) with no errors for this set of data.

The bit error rate curve for three different tap sizes, 7, 15, and 31 with $\lambda=0.9999$ is shown in Fig.4.33. As with the case for the LMS algorithm, this algorithm appears to be very efficient in suppressing narrowband interference regardless of the number of taps.

From these simulations, the recursive-least-squares algorithm appears to be a very

efficient algorithm to suppress narrowband interference. To verify these results, an experimental system will be built to test this algorithm further and the results presented in the later chapters.

Parameter	Value
Data Size	2048
Number of Taps	15
λ	0.9999

Table 4.4 Parameters for the RLS algorithm.

Figure 4.30 Squared error function for RLS algorithm with $\lambda = 0.9999$.

Figure 4.31 Power spectral estimation of the received sampled signal with interference and noise (top graph) and the power spectral estimation of the signal using the RLS algorithm (bottom graph) with $\lambda = 0.9999$.

Figure 4.32 Transmitted data D_k and the decoded data $\hat{D}_k$ using the RLS algorithm with $\lambda=0.9999$.

Figure 4.33 Bit error rate curve for 7, 15, and 31 tap vectors for the RLS algorithm with $\lambda=0.9999$.

4.3.5 Wiener-Hopf Algorithm

In this section, the Wiener-Hopf algorithm will be looked at next with the parameters listed in Table 4.5. For this simulation, the algorithm will be done on blocks of 1024 data samples. For this case, since 2048 data samples are generated, two sets of weights will be calculated.

By using Eq.(3.61), this algorithm is very straightforward but is very computationally complex since matrix inversion is required. Figure 4.34 illustrates the power spectrum before and after the algorithm for the first block of data. The decoded data for this first block is shown in Fig.4.35 with no errors in this case. Finally, the BER curves for three different tap sizes, 7, 15, and 31, are illustrated in Fig.4.36. This algorithm appears to perform well regardless of the number of taps used, however the number of computations involved increases with increasing filter size.

Although, this algorithm is very capable of suppressing narrowband interference, requires a large number of computations in order to compute the autocorrelation functions and to invert this matrix. As a result, this algorithm will not be tested experimentally, but will be used to compare the other algorithms in the next section.

Parameter	Value
Data Size	2048
Number of Taps	15
Block Size	1024

Table 4.5 Parameters for the Wiener-Hopf algorithm.

Figure 4.34 Power spectral estimation of the first 1024 received data samples (top graph) and the filtered data for this block (bottom graph).

Figure 4.35 Transmitted data, D_k , and the decoded data, $\hat{D}_k$, using the Wiener-Hopf algorithm with a 15 tap weight vector for the first 1024 sample data block.

Figure 4.36 Bit error rate performance of the Wiener-Hopf algorithm with tap sizes of 7, 15, and 31.

4.4 Summary

From the simulation results, each algorithm appears to be capable of suppressing the narrowband interference from the spread spectrum signal. Figure 4.37 compares these algorithms in terms of the bit error rate. The parameters listed in Tables 4.2–4.5 were used to generate the curves for the FFT, LMS, RLS, and Wiener-Hopf algorithms respectively.

Although the Wiener-Hopf algorithm approaches the optimal set of weights for a stationary interferer, this comes at the expense of computational power and speed, since the autocorrelation matrix must be first calculated and then inverted. As a result, this algorithm, although the most efficient as seen by the BER curve, will not be implemented experimentally but will be used to compare the other algorithms.

From Fig.4.37, the second most efficient algorithm is the RLS. This algorithm easily converges to the optimal solution much more rapidly than the LMS algorithm with approximately the same order of magnitude of computations required. However, one drawback is that this algorithm must be restarted periodically in account for the computational precision error. The LMS algorithm, on the other hand, converges to the optimal solution with only one initialization required and is much easier to implement.

The FFT algorithm is a frequency dependent algorithm which does not provide the same results as the other algorithms. However, with the availability of high speed DSP and specialized FFT boards, this algorithm provides a fast and inexpensive method to suppressing interference.

Figure 4.37 Bit error rate performances of the four algorithms in comparison to the performance with no suppression algorithm and with the theoretical BER of BPSK signal with AWGN only.

Chapter 5

Experimental Communication System

5.1 Introduction

In the previous chapter, the theoretical analyses of the four algorithms have illustrated their respective effectiveness in combatting intentional narrowband interference. However, to further verify their capabilities, these algorithms must be tested experimentally. Although it was not possible to actually test these algorithms in a real hostile environment, a hardware experimental communication system was implemented and built in a laboratory. This system consisted of all the necessary components to subject the transmitted signal under the same conditions, and with such a controlled environment, various performance tests on each algorithm were carried out.

This chapter will discuss the experimental communication system which can implement these algorithms to suppress a real narrowband continuous interferer. Each component of the system and any experimental considerations and/or difficulties encountered will be discussed.

5.2 System Overview

To simulate an intentional jamming environment, several components are necessary. These components include: a spread spectrum generator, a carrier frequency synthesizer, intentional narrowband continuous jammer, AWGN generator, receiver, and high speed DSP boards. Figure 5.1 illustrates the system setup of these components.

Figure 5.1 Experimental system setup.

Although a spread spectrum generator was readily available for use, this generator was far too complex and would require a vast amount of computational power and time to track, synchronize and despread the signal on the 9-cell parallel processing iWarp boards. Instead, a simple spread spectrum generator was designed and built using standard TTL logic chips. This generator produces [4,1] sequences for both the PN spreading code and data with exactly one complete PN sequence for each data bit (i.e., $code\ length/pg = 1$), which allows for easier correlation and bit detection algorithms on the iWarp boards.

The receiver that was used is the Steinbrecher Model DT-14088 C-1 Digital Tuner. This receiver is a wideband HF/LVHF double-conversion superheterodyne receiver which accepts information contained within a tunable 2.1 MHz passband over an input frequency range of 1.5 to 88.0 MHz and translates this information to a baseband frequency range of 2.7 to 4.8 MHz. This information is then converted to digital form at a sampling rate of 5 MHz.

Since this tuner only accepts input with a frequency range between 1.5 to 88.0 MHz, the spread spectrum signal must be mixed up between this range. The HP-3325A synthesizer/function generator was used to generate the carrier frequency at 15 MHz with an amplitude of +10dBm. The spread spectrum signal and the carrier signal were mixed together using a Mini-Circuits model ZP-10514 frequency mixer.

In order to simulate a real RF hostile environment, the spread spectrum signal must be subjected to both narrowband intentional jamming as well as AWGN. To generate the interference, an HP-3335A synthesizer/function generator was used which can produce a continuous single tone interference at any frequency on the main lobe of the spread

spectrum signal and any jammer power to signal power ratio (P_I/P_S). The AWGN was generated using the NOD 5000 series noise generator which has a frequency range from 10 Hz to 2 GHz. This generator was connected to a model KAY-5150 attenuator so that BER curves could be determined. The interference, noise and spread spectrum signal were combined together using the Mini-Circuits model ZSC 3-1 three-input combiner. The output of the combiner was then fed into the input of the receiver.

To perform the interference excision algorithms, a 9-cell parallel processing DSP board called the iWarp was installed in a Sun 4/370 sparcstation. This board receives the digital data from the Steinbrecher through the iWaim, an I/O interface board between the Steinbrecher output and one cell on the iWarp board. The iWarp board was then programmed to perform the three algorithms, the FFT, LMS, and RLS, as well as signal despreading and bit detection. The output data bits were then saved in a file encoded as 30-bit long integers, with the Sun workstation used to generate graphical analyses.

The next section will describe each component of this communication system in more detail.

5.3 System Components

5.3.1 Spread Spectrum Generator

The spread spectrum generator produces a [4,1] 15 length sequence for both the PN and data. The processing gain for this circuit was 15, thereby giving exactly one complete PN sequence per data bit, which reduced the complexity required to perform bit detection. Figure 5.2 illustrates the circuit diagram for this generator which will now be discussed.

Figure 5.2 Circuit diagram of spread spectrum generator.

In order to keep the frequency drifts between the receiver input and output synchronized, the 5 MHz clock used for the sampling clock of the receiver was also used to clock the spread spectrum generator. This 5 MHz clock could then be divided to generate the chip and data clock all in synch with the receiver.

The chip clock was generated by using two divide-by-16 counters (chip clock generator) to divide the 5 MHz clock by 32, thereby obtaining a chip rate of 156.25 KHz. This chip rate was then fed into a divide-by-15 circuit, (data clock generator), which results in a data rate of 10.417 KHz or a processing gain of 15. These two signals were then used to clock two identical sequence generators. These sequence generators consisted of 4 SR flip-flops with the 4th and 1st bits XORed together to provide the feedback. The reset switch was used to feed initial ones to begin the sequence and also keep the start of the sequence constant. The output of the two sequence generators were then XORed together to produce the spread spectrum signal with +5v and 0v as the references. This signal was then passed through a line driver to produce a signal with ± 15 volts as the transmitting signal. Figure 5.3 illustrates the generated signals.

5.3.2 Carrier Frequency Generator

The carrier frequency generator was used to mix the baseband spread spectrum signal up to a frequency within the range of the Steinbrecher receiver input. For this purpose, the HP-3325A synthesizer/function generator was used. This generator can produce a sine wave with a minimum frequency of 1 μ Hz and a maximum frequency of 20 MHz. The output amplitude can range from 1 mV to 10 V peak-to-peak. For the experimental system, the carrier frequency was set at 15 MHz and a power level of +10 dBm, enough

Figure 5.3 Generated signals of the spread spectrum circuit in Figure 5.2.

to drive the mixer, was used. In order to keep the frequency drifts of the receiver and carrier synchronized, a 10 MHz reference signal was used to clock both the receiver and function generator.

5.3.3 Interference Generator

To generate a continuous narrowband interference, the HP-3335A synthesizer/level generator was used. This generator is capable of generating a single tone over the range of 200 Hz to 80 MHz with an output resolution of 0.001 Hz and an output power level over the range of +13.01 dBm to -86.98 dBm. For this system, the interference generator was set such that the interference fell along the main lobe of the transmitted signal with a interference to signal power of approximately 20 dB.

5.3.4 Noise Generator

To simulate an RF channel, the NOD-5107 AWGN generator was used. This generator is capable of producing random instantaneous peak signals covering a range from 10 Hz to 2 GHz with an output power level of +10 dBm. To obtain different SNR results, the noise generator was connected to a KAY-5150 attenuator which is capable of attenuating signals from 0 dB to 80 dB in steps of one dB.

5.3.5 Steinbrecher Receiver

The Steinbrecher Model DT-14088 C-1 Digital tuner is a wideband HF/LVHF double-conversion superheterodyne receiver. It accepts an analog input consisting of sinusoidal frequency components in the range of 1.5 to 88.0 MHz and, using two mixing processes, converts the signal to baseband (2.7 to 4.8 MHz). By adjusting the first local oscillator, one can center a 2.1 MHz passband segment of the input signal anywhere between 2.7

to 4.8 MHz. The Steinbrecher converts this information to digital form with 14 bit resolution at a sampling rate of 5 MHz.

5.3.6 iWarp DSP Boards

iWarp is a parallel computer system built by Intel and capable of supporting high-performance computational algorithms including matrix operations and signal processing. It can support two forms of communications between cells : memory-based and systolic-based. Memory-based communications uses memory block transfers between cells thereby separating the communication and computation components thus allowing for multitasking operations. In systolic communications, each cell must retrieve each data immediately as it arrives thus creating a long data stream through the array. This type of communication allows for fine-grain communication, reduced access and size of local memory and increased instruction-level parallelism.

The iWarp system consists of three single-board arrays, two containing four cells each, and the other containing an interface cell for the host workstation. The cells are configured as a two-dimensional torus array and connected by communication pathways as shown in Fig.5.4(a). Each iWarp cell consists of an iWarp component, which comprises independent communication and computation agents, and 1 Mb of local memory, except the interface cell which has 2 Mb of local memory. Each cell supports four full duplex I/O channels with a performance of 40 Mb/sec per each channel for a combined input data bandwidth of 160 Mb/sec and an output data bandwidth of 160 Mb/sec per cell. The computational agent is capable of sustaining a 160 Mb/sec data I/O capacity with local

memory and has a peak performance of 20 MFLOPS for 32-bit single precision and 10 MFLOPS for 64-bit double precision computations, as illustrated in Fig.5.4(b).

Programming the iWarp system is done using the iWarp C and assembler compilers as well as the communication libraries. The programming methodology of each cell for the different algorithms will be illustrated in the next chapter.

5.3.7 Sun SparcStation

A Sun 4/370 SparcStation is used to house the three iWarp boards and provide data analysis. With the iWarp storing the resulting data bits to a file, this workstation, using commercial software, decoded the results, calculated the bit error rates and generated graphical outputs.

5.4 System Limitations

After initial programming of the iWarp system, several facts soon became evident. First, the spooling function for performing memory-based communications was not functioning properly. As a result, systolic communication had to be used which required the computational agent to wait until all the data in each block had been transferred before processing the data. This greatly reduced the allowable processing time per cell necessary to keep up with the 5 MHz data rate coming in. Also, with this method of data transfer, the cells could not be used in a feedback configuration since the cells would be too busy reading and sending data to perform any computations efficiently. Consequently, both the LMS and RLS algorithms could only be programmed to run on a single cell, resulting in a much lower maximum data rate.

(a)

(b)

Figure 5.4 (a) Two-dimensional torus array for a 9-cell iWarp system. (b) iWarp system architecture and I/O capacity.[13]

Another problem encountered was that the actual performance of the iWarp system was much less than the stated performance in the literature. With these hardware problems, the iWarp system was not capable of handling a 5 MHz data rate. To solve this problem, decimation of the data to a lower sampling rate was required. This was accomplished by processing every Nth data from the receiver. Since sampling at the chip rate is required, the spread spectrum chip rate had to be lowered as well. This is the reason for the two divide-by-16 counters in the chip clock generator section of Fig.5.2.

5.5 Summary

This chapter described in detail, the experimental communication system that was used to test the excision algorithms under an intentional jamming environment. Each component of this system and the necessary modifications required for it to function properly was discussed. The results of each algorithm will be presented in the next chapter.

Chapter 6

Experimental Results

6.1 Introduction

To determine the performance of the three excision algorithms (FFT, LMS, and RLS), experiments were done on the system described in Chapter 5. These experiments mainly involved the generation of bit error rate curves for each algorithm to compare with the simulations in Chapter 4. The results of these experiments are reported in this chapter.

6.2 Experimental Parameters

In order to make an unbiased comparison between these algorithms, the same input parameters were used. To create an effective jammer in which the inherent interference rejection ability of spread spectrum signals was not enough to suppress the interference alone, the interference to signal power was set to approximately 100 (20 dB). The interference was centered at approximately 20 KHz off the center frequency of the main lobe of the spread spectrum signal, i.e., 15.020 MHz. The SNR is defined as

$$SNR = \frac{E_b}{N_o} \quad (6.1)$$

where E_b is the bit energy and N_o is the spectral noise density. The signal to noise ratio was varied between 0 to 12 dB in steps of one dB to generate the bit error rate curves. With these settings, the input interference created a real jamming situation with real signals similar to the simulations and from which the performance of the algorithms could be determined.

Figure 6.1(a) illustrates the time data of the baseband spread spectrum signal which was transmitted to the receiver. [4,1] PN and data sequences were used with a chip rate of 156.25 KHz and a processing gain of 15. Figure 6.1(b) illustrates the power spectrum of the same spread spectrum signal but modulated on a carrier centered at 15 MHz for input into the receiver. Figure 6.2 shows the time domain and frequency spectrum of the AWGN signal used to simulate an RF channel. The amplitude of this signal was controlled by an attenuator to generate the appropriate SNR to calculate the BER. The continuous narrowband interferer was achieved by using a function synthesizer to generate a sine wave with frequency 15.020 MHz and the power controlled from the synthesizer's front panel. The time domain and frequency spectrum of the interference is illustrated in Fig.6.3. Figure 6.4 shows the power spectrum of the analog receiver's input signal which contains the spread spectrum signal, noise and interference signal.

At the receiver output, the digital data was transferred to the iWarp where the data was first decimated to the chip rate, 156.25 KHz, and mixed to baseband. The data was then processed by the iWarp system to perform the excision algorithms. Figure 6.5 illustrates the power spectrum of the baseband digital signal for a 1024 block of data from the receiver. This interference is easily recognizable at 20 KHz. Finally, Fig.6.6 illustrates the result with no interference excision algorithm performed on this block of data. The upper graph shows the transmitted bits while the middle plot shows the decoded result of the interference. Several errors are visible right away (lower plot) signifying the need for an excision algorithm.

The remaining chapter reports the performance of the three excision algorithms used

to suppress the interference in the above signal.

6.3 FFT Algorithm

6.3.1 System Configuration

To determine the performance of the FFT algorithm, each cell on the iWarp system was programmed to perform individual parts of the algorithm in parallel in order to maintain the 156.25 KHz sampling rate. The task and I/O responsibility of each cell is illustrated in Fig.6.7. Cell 3 contains an interface board which allows the cell to read in every 32nd sample from the Steinbrecher receiver. This cell then outputs alternating N-size blocks of data to cells 2 and 6. These cells are responsible for performing N-point FFTs on the data and then output the results to cell 7. Cell 6 receives the blocks of FFTs and determines the power spectral estimation on K FFT blocks using the Welch method. The cell then performs the remaining parts of the algorithm to determine the tap weight vector for the $N \times K$ size block of data and outputs both the block of data and the tap values to cell 5. Cell 5 performs linear filtering of the data with the tap values and outputs the filtered data to cell 1. Cell 1, which contains the PN sequence, correlates this sequence with the filtered data producing a series of correlation peaks. Cell 0 retrieves these correlation peaks and performs bit detection and estimation. Finally, cell 4 encodes these bits into groups of 30 and outputs them as a long integer to a file where BER computations are done.

The algorithm parameters used for interference excision are listed in Table 6.1. For the experiment, the FFT size N was 16, with the power spectrum estimated on 64 blocks with no overlap. A 15-tap vector is generated for each block of 1024 samples.

Time Data of Baseband SS Signal

(a)

Power Spectrum of SS signal Mixed Up to 15 MHz

(b)

Figure 6.1 (a) Time data of the baseband spread spectrum signal with [4,1] PN and data sequences, chip rate of 156.25 KHz and PG=15. (b) Power spectrum of the signal in (a) modulated on a carrier centered at 15 MHz.

Time Data of Noise Signal

(a)

Power Spectrum of Noise Signal

(b)

Figure 6.2 (a) Time data of AWGN signal. (b) Power spectrum of the AWGN signal.

Time Data of Interference Signal

Power Spectrum of Interference Signal

Figure 6.3 (a) Sinusoidal wave to create a continuous narrowband interference signal with a frequency of 15.020 MHz and $P_I/P_S = 20$ dB. (b) Power spectrum of the interference tone.

Power Spectrum of Receiver Input

Figure 6.4 Power spectrum of the analog signal containing the spread spectrum, AWGN, and interference signals.

Figure 6.5 Power spectrum of the received digitized signal demodulated to baseband and decimated by a factor of 32 such that sampling rate = chip rate = 156.25 KHz.

Figure 6.6 Transmitted data bit (upper graph), and decoded data bits (middle graph) with no excision algorithm on the received jammed signal. The squared error for this set of data is shown in the lower graph.

Figure 6.7 The cell responsibilities on the iWarp system to perform the FFT algorithm.

Parameter	Value
Block Size	1024
Number of Taps	15
Number of FFTs to Average	64
Size of the FFT	16
Overlap	0
Window Type	Rectangular

Table 6.1 Parameters for performing the FFT algorithm on the iWarp system.

6.3.2 Results

The results of the FFT algorithm using the parameters in Table 6.1 are reported in this section. For this experiment the following conditions were used:

$$\begin{aligned} E_b/N_o &= 12 \text{ dB} \\ P_I/P_S &= 20 \text{ dB}. \end{aligned} \tag{6.2}$$

Recall from Chapter 3 that to filter the jammed spread spectrum signal, the power spectral density must be computed. Figure 6.8 illustrates the power spectral estimation for a 1K block of data by averaging 64 blocks of 16-point FFTs. After interpolation of the data from 16-pts to 15-pts, the transfer function $H(k)$ can easily be computed. The result for this 1K block of data is shown in Fig.6.9. The top graph represents the magnitude of the transfer function with the notches easily visible near the interference frequency of 20 KHz. The bottom graph represents the linear phase of the transfer function, which is necessary to prevent any phase distortion.

Taking the inverse FFT of the transfer function results in the impulse response and tap weight vector. Figure 6.10 illustrates the impulse response for the transfer function in Fig.6.9. This tap weight vector has been normalized in order to keep the filter gain to unity. Although the resulting filter, shown in Fig.6.11, appears to provide a small suppression of the interference, approximately 10 dB in Fig.6.12, this suppression should be adequate such that the anti-jamming capabilities of the spread spectrum signal can suppress the rest.

The upper plot of Fig.6.13 shows the resulting correlation between the PN sequence

and the filtered data. This signal contains very noticeable positive and negative correlation peaks spaced 15 points apart, i.e., the processing gain. By sampling at these peaks, bit detection of the spread spectrum signal can be done as shown in the lower graph of Fig.6.13. Figure 6.14 illustrates a segment of the results for the FFT algorithm. For this block of data, the decoded bits shown in the middle graph were exactly what was transmitted in the top graph, verifying that the small suppression is enough for the SS signal to sufficiently suppress the remaining interference.

The BER performance of the FFT algorithm is shown in Fig.6.15. In this graph the bit error rate as a function of bit energy to noise power for several cases is plotted. The effects of the narrowband interference is quite obvious. Without the suppression algorithm the bit error rate was approximately 0.1 regardless of the SNR used, resulting in very poor communications. However, when the FFT algorithm was applied, the performance improved considerably, improving by almost a factor of 1000 for a SNR of 12.

In order to determine the effectiveness of this algorithm, an experiment on this system was performed whereby the spread spectrum signal was subjected to AWGN only. In this situation, no interference excision algorithm was implemented since a normal communication channel was used. The BER curve for this situation was plotted in Fig.6.15 as well. From this plot, a much better performance was achieved than when interference and the excision filter was present. The difference may be attributed to the effects of the residual interference not suppressed by the filter and the distortion introduced onto the filtered data by this algorithm.

Figure 6.8 Power spectral estimation of the received digital signal by averaging 64 16-pt FFTs.

Figure 6.9 The magnitude (upper plot) and phase (lower plot) of the transfer function of the suppression filter to be designed.

Figure 6.10 The normalized impulse response w_k of the linear suppression filter.

Figure 6.11 The continuous spectrum $H(e^{j\omega})$ of the suppression filter with impulse response in Fig.6.10.

Figure 6.12 (a) The power spectral estimation of the received digital signal before the FFT algorithm. (b) The power spectral estimation of the filtered data using filter in Fig.6.11.

Figure 6.13 (a) The correlation signal between the [4,1] PN sequence and the FFT filtered digital signal. (b) Bit detection of the filtered spread spectrum signal.

Figure 6.14 (a) The transmitted bits before spreading with the PN sequence. (b) The decoded data bits of the FFT filtered spread spectrum signal. (c) Squared error between (a) and (b).

Figure 6.15 Comparison of the BER curve of the FFT algorithm with BER curves using no algorithm, signal in AWGN only, and the theoretical BER for a BPSK signal in AWGN only. For the FFT algorithm, 16 pt FFTs and 15 taps were used.

The theoretical BER curve for a BPSK signal in AWGN was illustrated as a measure of the integrity of this experimental communication system. From the graph, the results of the experiment with no interference are consistent with the theoretical results. The slight discrepancies from SNRs of 11 to 12 are due to the limited number of samples that could be used because of memory limitations on the iWarp system. Therefore, the results generated from this system may be considered a true representation of the performance of the FFT algorithm under study.

From this experiment, the performance of the FFT algorithm using real signals was determined. The resulting bit error rate curves illustrated both the destructive nature of the narrowband interference and the effects of the FFT algorithm. This algorithm has shown that it can provide a relatively simple excision method which is capable of suppressing a continuous narrowband interferer and will later be compared with other algorithms using the same communication system.

6.4 LMS Algorithm

6.4.1 System Configuration

To determine the performance of the LMS algorithm, this algorithm was programmed on the iWarp system. Since systolic communications had to be used and the LMS algorithm requires continuous update of the tap vectors after each data sample, it was impractical to run the algorithm on more than one cell. This would require the cells to send and receive updated vector parameters each time a data sample was sent, resulting in a much slower data rate. As a result, the LMS algorithm was programmed only on a single cell as shown in Fig.6.16.

In this configuration, cell 3 retrieves the digital time data from the Steinbrecher receiver and sends the data to cell 7. Cell 7 then performs the LMS algorithm while sending the filtered results to cell 6. The filtered data is then correlated with the PN sequence in the same manner as for the FFT algorithm. The output is then sent to cell 2 for bit detection with cell 1 encoding and outputting the decoded data bits.

Since cell 7 performs the entire LMS algorithm, it spends most of its time performing floating point operations. As a result, it cannot maintain the 156.25 KHz data rate from cell 3. To solve this problem, the sampling rate was decimated by another factor of 2. This results in a sampling rate and chip rate of 78.125 KHz. For this experiment, a 15-tap filter was used with the tap vector initialized to the zero vector. The step size parameter μ was set to 1E-7, well within the limits specified in Eq.(3.17) and the interference and noise parameters similar to the FFT algorithm.

6.4.2 Results

The results of this algorithm are illustrated in the next few pages. The upper plot of Fig.6.17 shows the power spectrum of the jammed signal. The highly peaked narrowband interference is clearly noticeable at 20 KHz against the noise like spread spectrum signal. The lower plot illustrates the power spectrum of the filtered data using the LMS algorithm. For this experiment, the interference has been successfully estimated and suppressed from the received signal with an attenuation of approximately 30 dB. However, the distortion resulting from this algorithm is very noticeable.

Figure 6.18 shows the squared error function of the LMS algorithm for this experiment. From this plot, the rapid convergence to the optimal Wiener solution is evident.

Figure 6.16 The cell responsibilities on the iWarp system to perform the LMS algorithm.

Figure 6.17 (a) Power spectrum of the unfiltered data with interference at 20 KHz and P_i/P_s approximately 20 dB. (b) Power spectrum of the filtered data using the LMS algorithm to suppress the interference. The number of taps was 15 and $\mu = 1E-7$.

Figure 6.18 The square error function of the LMS filtered data with $\mu = 1E-7$.

Figure 6.19 (a) The correlation signal between the [4,1] PN sequence and the LMS filtered data. (b) Bit detection of the filtered spread spectrum signal.

Figure 6.20 (a) The transmitted bits before spreading with PN sequence. (b) The decoded data bits of the LMS filtered spread spectrum signal. (c) Squared error between (a) and (b).

Figure 6.21 Comparison of the BER curve of the LMS algorithm with BER curves using no algorithm, signal in AWGN only, and theoretical BER for a BPSK signal in AWGN only. For the LMS algorithm, 15 taps and $\mu = 1E-7$ were used.

The top graph of Fig.6.19 represents the correlation signal between the LMS filtered data and the PN sequence while the bottom figure represents the bit detection from the correlation signal. In comparison with the correlation signal using the FFT algorithm, Fig.6.13 (a), the LMS algorithm results in more clearly defined correlation peaks, i.e. a less noisy signal. This indicates that the LMS algorithm provides better performance than the FFT algorithm at the same SNR. The squared error between the transmitted and decoded data bits are illustrated in Fig.6.20 with no errors for this block of data.

Finally, Fig.6.21 compares the BER curve for this experiment with the BER curves when no algorithm was used to suppress the interference and when the spread spectrum signal was subjected to AWGN only. Similar to the FFT algorithm, the LMS algorithm greatly improves the performance of a spread spectrum signal with narrowband interference. The difference in performance between the LMS algorithm and the experimental results with AWGN, i.e., no interference and excision filter, is due to the distortion caused by this algorithm. Since only a 15 length sequence was used, there is some correlation in the spread spectrum data. As a result, this signal will also be slightly filtered by the LMS algorithm adding to the distortion on the signal. However, from the results of this experiment, the LMS algorithm does provide a relatively simple and efficient method for interference excision.

6.5 RLS Algorithm

6.5.1 System Configuration

The implementation of the RLS algorithm on the iWarp system is shown in Fig.6.22. Similar to the problems encountered with the LMS algorithm, it was more efficient to

run this algorithm on a single cell than on several cells. This was done on cell 7. Cells 6, 2 and 1 were used in the same manner as for the LMS algorithm to perform data correlation, bit detection and bit encoding respectively.

Since this algorithm requires several computations for each time data, rounding errors in each variable will occur for long blocks of data resulting in a corrupted signal. The rescue variable in the algorithm is used to monitor this situation, requiring the algorithm to be restarted when it becomes negative. For this experiment, a data block size of 102400 chosen. This means that after every 102400 data transmitted, the RLS algorithm was restarted, ensuring that the round off errors do not play a major factor. For this experiment, the weighting factor λ was chosen to be 0.9999.

6.5.2 Results

The effects of this algorithm are shown in Fig. 6.23. The top graph illustrates the power spectrum of the received data which contains the interference and AWGN. The bottom graph illustrates the power spectrum of the filtered data. As in the case for the LMS algorithm, the interference has been significantly suppressed, approximately 30 dB, from the received signal. However, in this algorithm the distortion introduced is noticeable less than in the LMS case.

Figure 6.24 illustrates the squared error function for a single block of data in this experiment. From this plot, the rapid convergence to the optimal solution is very noticeable. This rate of convergence is much greater than the convergence of the LMS algorithm, Fig.6.18, and should therefore result in better performance in terms of BER.

Figure 6.22 The cell responsibilities on the iWarp system to perform the RLS algorithm.

Figure 6.23 (a) Power spectrum of the unfiltered data with interference at 20 KHz and P_I/P_S approximately 20 dB. (b) Power spectrum of the filtered data using the RLS algorithm to suppress the interference. The number of taps was 15 and $\lambda = 0.9999$.

Figure 6.24 The square error function of the RLS filtered data with $\lambda = 0.9999$.

Figure 6.25 (a) The transmitted bits before spreading with the PN sequence. (b) The decoded data bits of the RLS filtered spread spectrum signal. (c) Squared error between (a) and (b).

Figure 6.26 Comparison of the BER curve of the RLS algorithm with BER curves using no algorithm, signal in AWGN only, and the theoretical BER for a BPSK signal in AWGN only. For the RLS algorithm, 15 taps and $\lambda = 0.9999$ were used.

The results of this algorithm are shown in Fig.6.25 and Fig6.26. The top graph of Fig.6.25 illustrates a block of data bits which was transmitted. The decoded bits for this block are shown in the middle graph with no errors for this series of data. The bit error rate curve for this experiment is shown in figure 6.26. As with the FFT and LMS algorithms, the RLS algorithm significantly improves the performance of a spread spectrum signal subjected to intentional interference. Again the differences in performance between the RLS algorithm and the experimental result with no interference was caused by the distortion due to the algorithm. The periodic restarts of this algorithm also contributed to the errors as well. From this experiment, this algorithm was shown to be very capable of suppressing a narrowband interference and can be used for interference excision.

6.6 Summary

In this chapter, the experimental results of the three algorithms, FFT, LMS, and RLS have been reported which appears to concur with the simulation results obtained in Chapter 4. From these experiments, the FFT, LMS, and RLS algorithms were successful in suppressing a stationary narrowband continuous interference signal to provide adequate performance. Figure 6.27 compares the three algorithms together. The FFT algorithm provided the worst interference excision with approximately 10 dB of suppression, while both the LMS and RLS algorithms provided approximately 30 dB of suppression. The remaining suppression was performed by the despreading of the spread spectrum signal. The LMS algorithm was very easy to implement and provided slightly better BER

performance. This algorithm rapidly converged to the optimal Wiener solution and suppressed the interference completely. However, the distortion caused by this algorithm onto the signal was very noticeable which contributed to the bit errors. The RLS algorithm provided the best performance. This algorithm completely suppressed the interferential without introducing much distortion to the signal. However the implementation was more complex than the other two.

Figure 6.27 Comparison of the experimental BER curves for the FFT, LMS and RLS algorithms using 15 taps.

Chapter 7

Conclusions

7.1 Summary

In this thesis, the effectiveness of several algorithms in suppressing a continuous narrowband stationary interference signal from a direct sequence spread spectrum signal was studied. These interference excision algorithms were the Fast Fourier Transform (FFT), Least Mean Square (LMS), Recursive Least Square (RLS) and Wiener-Hopf. To understand these algorithms, background material in direct sequence spread spectrum communications, linear prediction filter and Wiener filter theory were presented. Each algorithm was then discussed and simulations were performed. A hardware configuration was presented which was capable of performing these algorithms using real data. Experimental results have been obtained and reported in this thesis.

The FFT algorithm uses spectral analysis to design a transversal filter with notches located at the interference frequency. From the results of this thesis the performance of this algorithm was shown to be a function of the interference frequency. If this interference is located at a spectral frequency of the FFT size, stronger suppression will occur, otherwise residue interference will remain and must rely on the nature of spread spectrum systems to remove it during despreading, resulting in a higher BER. This can be controlled by the size of the taps and resolution of the power spectral estimation. A larger FFT size will increase the probability that the interference will land on a spectral frequency. However, this would come at a cost of increased computations and speed. As

a compromise, an FFT size of 15 was used in this thesis, which provided an adequate performance.

The LMS algorithm uses adaptive linear prediction to estimate a highly correlated narrowband interference from an uncorrelated spread spectrum signal. This estimation of the interference can then be removed from the received signal leaving the desired signal. This algorithm was the easiest to implement and suppressed the interference significantly and had a slightly better BER performance than the FFT algorithm. Since a 15 tap size was used, the resulting spread spectrum signal had some correlation between data. This resulted in very noticeable distortions on the filtered data and a higher BER from the theoretical BER for a BPSK signal in AWGN only.

The RLS algorithm, which also uses adaptive linear prediction in the same manner as the LMS algorithm, converges to the optimal Wiener solution at a much faster rate than the LMS algorithm and provides slightly better bit error rates. This results in complete suppression of the interference with a less noticeable distortion on the filtered data. However, this increased performance comes at a cost of programming complexity. This algorithm requires three transversal filters to perform filtering and adapting. After several iterations these filter parameters become susceptible to round off errors and the filtered data becomes corrupted. To overcome this problem at the expense of increased BER, this algorithm was restarted after each 102400 block of data.

The Wiener-Hopf algorithm, which also uses adaptive linear prediction, was performed with simulations only since this algorithm involves the inversion of an $N \times N$ matrix and the iWarp system would not be capable of implementing this at a high sam-

pling rate. Since this algorithm obtains and uses the optimal Wiener solution, it provides a basis to compare the other algorithms with. From Fig.4.37 and Fig.6.27, one can easily see that all these algorithms provide approximately the same performance in terms of bit error rates.

From the results obtained in this thesis, the FFT, LMS, RLS and Wiener-Hopf interference excision algorithms provides efficient methods for suppression of a stationary continuous narrowband interference.

7.2 Future Work

Four algorithms have been presented as possible interference excision algorithms for direct sequence spread spectrum signals. However, the area of intentional jamming is very large and vast amounts of work are still needed to be done. Some of the possible areas will now be discussed.

In this thesis, a stationary continuous narrowband interference signal was used. In fact there are many types of jamming techniques, pulse jamming, multiple jammers, frequency hopping jammers, etc. that can be investigated.

Another area of interest may be the jamming of frequency hopped spread spectrum signals. In this case, the algorithms presented here are not valid and new algorithms must be investigated and implemented.

Appendix A Matrix Inversion Lemma

The matrix inversion lemma states that if **A** and **B** are positive-definite, *M*-by-*M* matrices related by

$$\mathbf{A} = \mathbf{B}^{-1} + \mathbf{C}\mathbf{D}^{-1}\mathbf{C}^T \quad (\text{A.1})$$

where **D** is an *N*-by-*N* positive definite matrix and **C** is an *M*-by-*N* matrix, the inverse of the matrix **A** is

$$\mathbf{A}^{-1} = \mathbf{B} - \mathbf{B}\mathbf{C}(\mathbf{D} + \mathbf{C}^T\mathbf{B}\mathbf{C})^{-1}\mathbf{C}^T\mathbf{B}. \quad (\text{A.2})$$

For the RLS algorithm the following identifications are made:

$$\begin{aligned} \mathbf{A} &= \mathbf{R}_k \\ \mathbf{B}^{-1} &= \lambda \mathbf{R}_{k-1} \\ \mathbf{C} &= \mathbf{X}_k \\ \mathbf{D} &= 1. \end{aligned} \quad (\text{A.3})$$

References

- [1] Loh-Ming Li Lawrence B. Milstein. Rejection of narrowband interference in pn spread-spectrum systems using transversal filters. *IEEE Transactions on Communications*, VOL. COM-30, NO.5, pp 925-928, May 1982.
- [2] Jr. Maurice J. Bouvier. The rejection of large cw interferers in spread spectrum systems. *IEEE Transactions on Communications*, VOL. COM-26, NO.2, pp 254-256, February 1978.
- [3] Frank M. Hsu Arthur A. Giordano. Digital whitening techniques for improving spread spectrum communications performance in the presence of narrowband jamming and interference. *IEEE Transactions on Communications*, VOL.COM-26, NO. 2, pp 209-216, February 1978.
- [4] R.L. Pickholtz et al. Theory of spread-spectrum communications — a tutorial. *IEEE Transactions on Communications*, VOL. COM-30, pp 855-884, May 1982.
- [5] M.K. Simon et al. *Spread Spectrum Communications Vol. 1-3*. Computer Science Press, Rockville, 1985.
- [6] Rodger E. Ziemer Roger L. Peterson. *Digital communications and Spread Spectrum Systems*. Macmillan Publishing Company, New York, 1985.
- [7] R.C. Dixon. *Spread Spectrum Systems*. 2nd edition, John Wiley & Sons, New York, 1984.

- [8] Bernard Widrow Samuel D. Sterns. *Adaptive Signal Processing*. Prentice-Hall, Inc., 1985.
- [9] Simon Haykin. *Adaptive Filter Theory*. Prentice-Hall, 1986.
- [10] Brian Kozminchuk. A fast fourier transform approach to interference suppression in direct sequence spread spectrum signals. Technical report, Defence Research Establishment Ottawa, 1992.
- [11] John W. Ketchum John G. Proakis. Adaptive algorithms for estimating and suppressing narrow-band interference in pn spread-spectrum systems. *IEEE Transactions on Communications*, VOL.COM-30, NO. 5, pp. 913–923, May 1982.
- [12] Alan V. Oppenheim Ronald W. Schafer. *Discrete-Time Signal Processing*. Prentice-Hall, Inc., 1989.
- [13] Intel Corporation. *Introduction to iWarp*.
- [14] Simon Haykin. *Digital Communications*. John Wiley & Sons, 1988.

END

3 1 - 0 5 - 9 5

FIN