**FAST ROBUST QUASI-NEWTON
ADAPTIVE ALGORITHMS FOR
GENERAL ARRAY PROCESSING**

by

Marek Klemes

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

Doctor of Philosophy

**Ottawa-Carleton Institute for Electrical Engineering
Department of Electronics
Carleton University
Ottawa, Ontario**

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 accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

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

ISBN 0-612-19400-0

Name MAREK KLEMES

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.

Fast Robust Quasi-Newton Adaptive Algorithms for General Jmav

0544

U-M-I

SUBJECT TERM

Processing

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 0320
Home Economics 0278
Industrial 0521
Language and Literature 0279
Mathematics 0780
Music 0522
Philosophy of 0998
Physical 0523

Psychology 0525
Reading 0535
Religious 0527
Sciences 0714
Secondary 0532
Social Sciences 0534
Sociology of 0340
Special 0529
Teacher Training 0536
Technology 0710
Tests and Measurements 0288
Vocational 0747

LANGUAGE, LITERATURE AND LINGUISTICS

Language 0679
Ancient 0289
Linguistics 0290
Modern 0291
Literature 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 0318
Biblical Studies 0321
Clergy 0319
History of 0320
Philosophy of 0322
Theology 0469

SOCIAL SCIENCES

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

Ancient 0579
Medieval 0581
Modern 0582
Black 0328
African 0331
Asia, Australia and Oceania 0332
Canadian 0334
European 0335
Latin American 0336
Middle Eastern 0333
United States 0337
History of Science 0585
Law 0398
Political Science 0615
General 0616
International Law and Relations 0617
Public Administration 0814
Recreation 0452
Social Work 0452
Sociology 0626
Criminology and Penology 0627
Demography 0938
Ethnic and Racial Studies 0631
Individual and Family 0628
Studies 0629
Industrial and Labor 0630
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 0473
General 0285
Agronomy 0475
Animal Culture and Nutrition 0476
Animal Pathology 0359
Food Science and Technology 0478
Forestry and Wildlife 0479
Plant Culture 0480
Plant Pathology 0817
Plant Physiology 0777
Range Management 0746
Wood Technology 0306
General 0287
Anatomy 0308
Biosciences 0309
Botany 0379
Cell 0329
Ecology 0353
Entomology 0369
Genetics 0793
Limnology 0410
Microbiology 0307
Molecular 0317
Neuroscience 0416
Oceanography 0433
Physiology 0821
Radiation 0778
Veterinary Science 0472
Zoology 0786
Biophysics 0760
General 0425
Biogeochemistry 0996
Geochemistry

Geodesy 0370
Geology 0372
Geophysics 0388
Hydrology 0411
Mineralogy 0420
Paleobotany 0418
Paleontology 0985
Paleozoology 0427
Physiology 0368
Physical Oceanography 0415

HEALTH AND ENVIRONMENTAL SCIENCES

Environmental Sciences 0768
Health Sciences 0566
General 0300
Audiology 0992
Chemotherapy 0350
Education 0769
Hospital Management 0738
Human Development 0982
Immunology 0564
Medicine and Surgery 0347
Mental Health 0539
Nursing 0570
Nutrition 0380
Osteonics and Gynecology 0354
Therapy 0381
Ophthalmology 0419
Pathology 0572
Pharmacology 0382
Physical Therapy 0574
Public Health 0575
Radiation 0575
Recreation 0575

Speech Pathology 0460
Toxicology 0383
Home Economics 0386

PHYSICAL SCIENCES

Pure Sciences 0485
Chemistry 0749
General 0486
Analytical 0487
Biochemistry 0488
Inorganic 0738
Nuclear 0490
Organic 0491
Pharmaceutical 0494
Physical 0495
Polymers 0754
Radiation 0405
Physics 0605
General 0986
Acoustics 0606
Astronomy and Astrophysics 0608
Atmospheric Science 0748
Atomic 0607
Electronics and Electricity 0798
Elementary Particles and High Energy 0759
Fluid and Plasma 0609
Molecular 0610
Nuclear 0752
Optics 0756
Radiation 0611
Solid State 0463
Statistics 0346
Applied Sciences 0984
Applied Mechanics 0346
Computer Science

Engineering 0537
General 0538
Aerospace 0539
Automotive 0540
Biomedical 0541
Chemical 0542
Civil 0543
Electronics and Electrical 0544
Heat and Thermodynamics 0545
Hydraulic 0546
Industrial 0547
Marine 0794
Materials Science 0548
Mechanical 0743
Metallurgy 0551
Mining 0552
Nuclear 0549
Packaging 0745
Petroleum 0554
Sanitary and Municipal 0790
System Science 0428
Geotechnology 0796
Operations Research 0795
Plastics Technology 0794
Textile Technology

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

**"FAST ROBUST QUASI-NEWTON ADAPTIVE ALGORITHMS FOR
GENERAL ARRAY PROCESSING"**

submitted by

Marek Klemes, B.A.Sc., M.A.Sc.

*in partial fulfillment of the requirements
for the degree of Doctor of Philosophy*

Chair, Department of Electronics and
Thesis Co-Supervisor,
Professor J. S. Wight

Thesis Co-Supervisor,
Professor D. D. Falconer

External Examiner,
Professor Simon Haykin, McMaster University

Carleton University,
November 25 1996

Abstract

This thesis develops a new adaptive algorithm for general array signal-processing. It is a true stochastic Quasi-Newton (QN) Least Mean Squares (LMS) algorithm, as compared to simply an accelerated LMS algorithm. It is inherently faster than LMS but slower than the Recursive Least Squares (RLS) algorithm (which is also QN but optimizes a different performance function). It is unique in that, as LMS, it requires only order-of- N computations per iteration compared to order-of N^2 for RLS, yet is applicable to arbitrary array signals of N dimensions, unlike other order-of- N (Fast) RLS approaches.

Specifically, the new Fast Robust Quasi-Newton (FRQN) algorithm potentially has unique performance advantages over existing algorithms in time-critical digital adaptive antenna arrays, i.e. spatial filtering in a dynamic environment. That is because stable Fast RLS algorithms for general N -element adaptive arrays do not presently exist, only for tapped-delay-line (temporal) adaptive filters [6]. The popular, order-of- N LMS algorithm suffers from slow modes of convergence in commonly-encountered ill-conditioned scenarios, and from high misadjustment. It is shown, analytically and via representative computer simulations, that the FRQN algorithm has order-of- N complexity, fast Quasi-Newton convergence and very small misadjustment. Robustness in this thesis refers to the stability of its properties with respect to type of array signal and scenario. The "price" for these desirable features is a larger tracking error, more bits of precision and twice the complexity as the Normalized LMS (NLMS) algorithm.

Background material includes tutorial developments of the LMS and RLS algorithms, a review of competing algorithms in the recent literature and applications which motivate their development. The FRQN algorithm is derived in detail and

compared to the NLMS and RLS algorithms. Results demonstrate that in highly ill-conditioned and fast dynamic scenarios, FRQN outperforms the NLMS algorithm, and is only slightly worse than ordinary RLS at low to moderately-high signal-to-noise ratios. Analyses include stability in the mean and mean-square, misadjustment and tracking errors, computational aspects and ways of providing reference signals in possible applications.

Acknowledgements

For the original encouragement and motivation the author is indebted to his supervisor, Professor J. S. Wight, chairman of the Department of Electronics. The author is grateful to his co-supervisor, Professor D. D. Falconer of the Department of Systems and Computer Engineering, for his sincere and unwavering guidance throughout the course of this work. For the enthusiastic support with implementation issues the author is indebted to Professor T. A. Kwasniewski.

For their tireless optimism and moral support the author thanks his parents Dr.Sc. Vít and Marie Kleměš, and his brother Dr.Ivo Klemes for "de-mystifying" the matrix-inversion lemma which was a key instrument in advancing this research topic. Equally important as the beginning, the completion of this work would not have been possible without the kind, patient and understanding support of the author's family - wife Tiên, son Martin and daughter Christine.

Financial assistance of the author's employers, Canadian Marconi Company and later Nortel, is gratefully acknowledged and appreciated.

TABLE OF CONTENTS

Chapters and Sections:	Pages:
1.0 Background and Statement of the Problem	1
<i>1.1 General Context</i>	1
<i>1.2 Directions of Previous Research</i>	3
<i>1.3 Statement of Thesis Research</i>	5
<i>1.4 Synopsis</i>	7
2.0 Preliminaries	9
<i>2.1 Principles of LMS and RLS Adaptive Algorithms - A Tutorial</i>	9
<i>2.2 Relations to Continuous-Time Algorithms - The Normalized LMS Algorithm</i>	21
<i>2.3 The FRQN Algorithm in Relation to Similarly-Motivated Approaches - A Review of Recent Literature</i>	26
<i>2.4 Applications and Further Motivation</i>	35
3.0 Algorithm Development and Characterization	40
<i>3.1 Derivation of the FRQN Algorithm and Possible Variants</i>	40
<i>3.2 Relation to RLS and NLMS Algorithms - A Geometric Interpretation</i>	47
<i>3.3 Preliminary Simulation Results</i>	51
<i>3.4 Quantitative Simulation Comparisons of Basic FRQN and NLMS Algorithms</i>	60
<i>3.5 Refinements of NLMS and FRQN Algorithms</i>	71
<i>3.6 Characterization and Performance Comparisons of Algorithms</i>	84
<i>3.7 Computational Aspects</i>	103
4.0 Some Thoughts on Reference Signals	112
<i>4.1 Statistical Requirements of Reference Signals</i>	112
<i>4.2 Techniques</i>	118
<i>4.3 The FRQN Algorithm in a Transmitted Reference Application - The TRAAAP Concept:</i>	122
5.0 Summary and Conclusions	134
6.0 References	144

LIST OF APPENDICES

Appendix Titles:	Pages:
Appendix A: S to Z Transformations	150
Appendix B: Convergence in the Mean.....	162
Appendix C: Derivation of the FRQN Algorithm with More General S-to-Z Transformation	176
Appendix D: Convergence in the Mean Square	179
Appendix E: The Matrix-Inversion Lemma.....	217
Appendix F: Simulated Scenarios	220
Appendix G: Requirements for Reference-Generating Subsystems Operating with NLMS and FRQN Algorithms	234
Appendix H: Toward Implemetation	250
Appendix I: Guide to Abbreviations and Notation	260

LIST OF TABLES

Tables:	Pages:
Table 1: Merits of Adaptive Algorithms	8
Table 2(a): Well-Conditioned Scenario, $\lambda_{\max}/\lambda_{\min} = 28.05$, $N = 4$	63
Table 2(b): Ill-Conditioned Scenario, $\lambda_{\max}/\lambda_{\min} = 6.54 \times 10^6$, $N = 4$	63
Table 3: Basic FRQN Algorithm With Reduced Time-Constants	68
Table 4: BER performance of algorithms in stationary scenarios	99
Table 5: Initial FRQN Parameter Computations	104
Table 6: FRQN Operations per Iteration	105
Table 7: A Frequency Plan for TRAAAP with $M=6$	124

LIST OF ILLUSTRATIONS

Figures:	Pages:
Figure 1: Linear combiner structure	9
Figure 2: Quadratic penalty-surface in W-space	15
Figure 3(a): Evolution of FRQN weights in well-conditioned scenario	52
Figure 3(b): Evolution of FRQN residual output power in well-conditioned scenario	52
Figure 3(c): Initial and adapted antenna directivity patterns for FRQN algorithm in well-conditioned scenario	53
Figure 4(a): Evolution of NLMS weights in well-conditioned scenario	53
Figure 4(b): Evolution of NLMS residual output power in well-conditioned scenario	54
Figure 4(c): Initial and adapted antenna directivity patterns for NLMS algorithm in well-conditioned scenario	54
Figure 5(a): Evolution of FRQN weights in ill-conditioned scenario	56
Figure 5(b): Evolution of FRQN residual output power in ill-conditioned scenario	56
Figure 5(c): Initial and adapted antenna directivity patterns for FRQN algorithm in ill-conditioned scenario	57
Figure 6(a): Evolution of NLMS weights in ill-conditioned scenario	57
Figure 6(b): Evolution of NLMS residual output power in ill-conditioned scenario	58
Figure 6(c): Initial and adapted antenna directivity patterns for NLMS algorithm in ill-conditioned scenario	58

Figure 7(a):	Weights of FRQN algorithm in well-conditioned scenario	64
Figure 7(b):	Squared output error of FRQN algorithm in well-conditioned scenario	64
Figure 8(a):	Weights of NLMS algorithm in well-conditioned scenario.....	65
Figure 8(b):	Squared output error of NLMS algorithm in well-conditioned scenario	65
Figure 9(a):	Weights of FRQN algorithm in ill-conditioned scenario.....	66
Figure 9(b):	Squared output error of FRQN algorithm in ill-conditioned scenario	66
Figure 10(a):	Weights of NLMS algorithm in ill-conditioned scenario	67
Figure 10(b):	Squared output error of NLMS algorithm in ill-conditioned scenario	67
Figure 11(a):	Weights of FRQN of Table 3 in well-conditioned scenario	69
Figure 11(b):	Squared output error of FRQN algorithm of Table 3 in well-conditioned scenario	69
Figure 12(a):	Weights of FRQN of Table 3 in ill-conditioned scenario	70
Figure 12(b):	Squared output error of FRQN algorithm of Table 3 in ill-conditioned scenario	70
Figure 13(a):	Weights of FRQN algorithm in well-conditioned scenario	75
Figure 13(b):	Squared errors of FRQN in well-conditioned scenario	75
Figure 14(a):	Weights of NLMS algorithm in well-conditioned scenario	76
Figure 14(b):	Squared errors of NLMS in well-conditioned scenario	76
Figure 15(a):	Weights of RLS algorithm in well-conditioned scenario	77
Figure 15(b):	Squared errors of RLS in well-conditioned scenario	77

Figure 16(a): Weights of FRQN algorithm in ill-conditioned scenario	78
Figure 16(b): Squared errors of FRQN in ill-conditioned scenario	78
Figure 17(a): Weights of NLMS algorithm in ill-conditioned scenario	79
Figure 17(b): Squared errors of NLMS in ill-conditioned scenario	79
Figure 18(a): Weights of RLS algorithm in ill-conditioned scenario	80
Figure 18(b): Squared errors of RLS in ill-conditioned scenario	80
Figure 19: Averaged square error and mmse for the RLS algorithm in the well-conditioned scenario	95
Figure 20: Averaged square error, theoretical mse and mmse for FRQN algorithm	95
Figure 21: Averaged square error, theoretical mse and mmse for NLMS algorithm	96
Figure 22: Comparison of tracking performance in fast-fading, well-conditioned scenario	99
Figure 23: Comparison of tracking performance in fast-fading, ill-conditioned scenario	100
Figure 24: Comparison of tracking performance with fade-rate in well-conditioned scenario	101
Figure 25: Comparison of tracking performance with fade-rate in ill-conditioned scenario	102
Figure 26: Characterization of an Adaptive Array System with Reference Signal.	114
Figure 27: The receiving adaptive-array configuration for the base-station employing the TRAAAP concept	126

Figure A1:	Sketch of the s-to-z transformation given by (A7)	154
Figure A2:	Sketch of the backward-difference s-to-z mapping	155
Figure A3:	Sketch of the bilinear s-to-z mapping	157
Figure A4:	Sketch of the generalized, stability-preserving s-to-z mapping	161
Figure F1:	Well-conditioned scenario for simulating generalized sidelobe-canceller performance in Figures 3(a), 3(b), 3(c) and 4(a), 4(b), 4(c)	222
Figure F2:	Ill-conditioned scenario for simulating generalized sidelobe-canceller performance in Figures 5(a), 5(b), 5(c) and 6(a), 6(b), 6(c)	223
Figure F3:	Well-conditioned scenario for simulating reference-based Adaptive array algorithms in Figures 7(a), 7(b), 8(a), 8(b), 11(a) and 11(b)	...226
Figure F4:	Ill-conditioned scenario for simulating reference-based Adaptive array algorithms in Figures 9(a), 9(b), 10(a), 10(b), 12(a) and 12(b)	227
Figure F5:	Well-conditioned stationary scenario for comparing adaptation transients of FRQN, NLMS and RLS algorithms in Figures 13(a), 13(b), 14(a), 14(b), 15(a), 15(b) and 19, 20, 21, respectively	229
Figure F6:	Ill-conditioned stationary scenario for comparing adaptation transients of FRQN, NLMS and RLS algorithms in Figures 16(a), 16(b), 17(a), 17(b), 18(a) and 18(b), respectively	230
Figure F7:	Well-conditioned non-stationary scenario to simulate tracking performance of NLMS, FRQN, RLS algorithms versus SNR in Figures 22, 23 and Doppler in Figures 24, 25	232
Figure F8:	Ill-conditioned non-stationary scenario to simulate tracking performance of NLMS, FRQN, RLS algorithms versus SNR in Figures 22, 23 and Doppler in Figures 24, 25	233
Figure G1:	Interference (I, rank M) and SOI (S, rank 1) subspaces of complex N-dimensional array-space, containing steering vectors (and components of weight vector)	241

Figure H1:	SIMULINK [®] block diagram of Adaptive Noise Cancellation Model using a real-valued Normalized LMS algorithm	251
Figure H2:	Block diagram of SIMULINK [®] model of the Normalized LMS algorithm used in Figure H1	252
Figure H3:	SIMULINK [®] block diagram of Adaptive Noise Cancellation Model using a real-valued RLS algorithm	254
Figure H4:	SIMULINK [®] block diagram of Adaptive Noise Cancellation Model using a real-valued FRQN algorithm, including the noise-cancellation junction	255
Figure H5:	Block diagram of SIMULINK [®] model of the real-valued FRQN algorithm used in Figure H4	256

Chapter 1

Background and Statement of the Problem

1.1 General Context

As the number of users of electronic communication channels increases, problems arise due to mutual interference among users, thus limiting the capacity of the channels to transmit information. Higher transmission rates over greater distances have been imposed on these media in order to transmit the ever-increasing volume of information, leading to reception problems caused by channel distortions. In efforts to overcome these limitations, methods of compensating the channel degradations have been devised. Specifically, because the disruptive effects are usually *time-varying*, *adaptive* systems such as interference cancellers and equalizers for radio and wired links have been implemented. However, in many applications their speed is not sufficient to cope with the dynamics of the channel.

Such adaptive signal-processing systems are governed by mathematical algorithms which in some way measure the signal and/or distortion parameters (usually via estimates of the statistics of received signals), and adjust the coefficients of some form of filter (e.g. a linear combiner) so as to optimize a performance measure (typically a quadratic function of the difference between the received and desired signal waveforms).

Adaptation rate is much lower than the signalling rate so the adaptive system is essentially linear with respect to the signal, but it must adapt faster than the rate of change of the channel or interference environment. Adaptation speed (for convergence and/or tracking) of a practical adaptive system is limited by the complexity of its algorithm, speed of the hardware on which it is implemented, the number of filter

coefficients to be adapted and for some algorithms, on the bandwidth occupied by the received signals at the input of the system. The speed of some algorithms is also a function of the input signals' statistics, generally giving rise to fast as well as slow modes of coefficient adaptation which are largely beyond the user's control.

Fast modes are undesirable because they cause the system to depart from a quasi-linear filter, giving rise to such phenomena as partial signal cancellation. Slow modes tend to prolong the response times to intolerable lengths in fast dynamic scenarios. Other algorithms are impractical due to their computational complexity, meaning there is insufficient time between the Nyquist samples to perform the computations of one algorithm iteration. As there is no ideal algorithm, there are many applications for which the performance of current adaptive algorithms is inadequate. Consequently, much past and present research and development effort is directed toward improving upon the performance of some of the well-known algorithms, and/or making them easier to implement in real-time.

Perhaps the most well-known and widely used adaptive algorithm is the so-called Least Mean Squares (LMS) algorithm (see e.g. [1]). It requires in the order of N complex multiply-add operations per iteration, so it is one of the simplest to implement. However, it generally requires many iterations (more than $10N - 100N$) to converge. In fact, its convergence speed is directly at the mercy of the signal statistics. While its performance is sufficient for many applications, it is not acceptable for some of the more recent ones such as adaptive antenna arrays (spatial filters) for mobile digital cellular radio communications [13, 14]. Fast adaptive time-domain filters and equalizers also require a faster, more robust algorithm because only short sequences of known data symbols are available for "training" them and limited computing power is available to compute coefficient updates in the short time intervals between signal samples.

1.2 Directions of Previous Research

Despite the tremendous progress of technology over the past 20 years, the quest for faster and more robust adaptive algorithms continues, because the same technology also allowed the pace and density of electronic information transfer to increase, thus fuelling the need. More complicated and faster algorithms became practical, but they were required in ever higher-speed applications.

Whereas the LMS algorithm follows the path of steepest descent along the surface of the penalty-function in filter-coefficient space (in an average sense), other algorithms such as Recursive Least Squares (RLS) or Direct Matrix Inversion (DMI) follow a more direct path to the optimal coefficients, converging in $2N$ to $3N$ iterations. However, RLS requires in the order of N^2 computations per iteration and DMI requires in the order of N^3 , whereas LMS requires only in the order of N computations.

Many ongoing research efforts are devoted to reducing the order of their computational complexity to N for practical implementation. Specifically, efforts were directed at developing a fast RLS (FRLS) algorithm having order-of- N complexity. Unfortunately most of the results suffered from numerical instability requiring further "rescuing" operations for practical implementation with finite-precision hardware. Performance in nonstationary environments was not reliable. Only recently a stable FRLS algorithm has been devised [6], but only for FIR types of time-domain filters. It relies explicitly on their tapped delay-line structure for greater computational regularity, hence speed. Since spatial adaptive filters (e.g. radar, sonar or communications antenna adaptive arrays) do not give rise to such a regular signal structure, it follows that no FRLS algorithm yet exists for general adaptive arrays. (Signal-vectors of uniform line arrays resemble those of tapped delay-lines but with a different inter-tap delay for each

angle of arrival of the planewave signal. For arbitrary arrays with mutual coupling among the elements, and arbitrary propagation modes, the relationship among the entries of the signal-vectors is even more complicated, though presumably linear.)

An alternative approach to an improved practical algorithm is to increase the speed of the LMS algorithm without sacrificing too much of its appealing simplicity, stability and robustness. Efforts tend towards reducing its dependence on the signal statistics, specifically the eigenvalues of the input-signal covariance matrix. The hope is to make it follow the same convergence path as RLS in fewer iterations than LMS, at least approximately. These are the so-called Quasi-Newton (QN) algorithms [1,3,4,5]. Although RLS is also a QN algorithm [28], the interest is in those having order-of-N complexity.

Besides the FRLS time-domain FIR adaptive algorithm [6], analogue implementations of more general QN algorithms have been developed [4,5]. While suitable for high power and/or high bandwidth applications, analogue hardware is more bulky, costly and difficult to design than current digital hardware. Moreover, analogue signals are susceptible to corruption by noise, signal leakage and radiation, bias and spurious disturbance signals, and the physical idiosyncrasies of analogue components. Systems are hard-wired and difficult to reconfigure for diverse applications, which require costly redesign. A limited variety of physical components is available to perform basic signal-processing functions. By contrast, digital implementations are robust with respect to physical hardware effects, precision is easily retained and computations more readily tailored to specific applications, especially with the support of the extensive hardware and software infrastructure of computer-aided design (CAD) tools. An ever-increasing variety of signal-processing functions is readily programmable into the same compact digital hardware so cost of implementation is presently decreasing.

With the present trend being toward digital implementation of communications and control systems thanks to the advances in technology, many present (and most future) adaptive systems require a digital implementation. In that form, they are also in the best position to make use of digital information which is being transmitted in ever-increasing volume with many types of signals. Presently, the need for fast QN algorithms for general adaptive arrays is unfulfilled in digital radio communications such as from aircraft [4], personal mobile radio¹ and emerging digital cellular mobile-radio systems [13,14], where the LMS algorithm has been found to perform inadequately.

1.3 Statement of Thesis Research

Against this general background, this thesis advances a new, digital adaptive algorithm which is unconditionally stable, robust with respect to arbitrary signal-vector structures, and, with only order-of-N complexity, approximates the quasi-Newton convergence behaviour of RLS algorithms. (The QN property is explained in the Appendix of [5] as well as in the following tutorial chapter herein.) In this thesis, "robustness" is used to mean stability of the convergence properties with respect to type of array signals, their statistics and freedom from analytic divergence, rather than the mathematically formal robustness based on perturbation analysis. As will be shown in subsequent chapters, this FRQN algorithm potentially has unique performance advantages over existing algorithms in time-critical, digital adaptive-array applications, i.e. spatial filtering in a dynamic environment. The hope is that the analysis and optimization of its convergence, tracking and other key properties will result in a useful approximation to an FRLS adaptive algorithm for general array signal processing. Even if its speed does not ultimately match that of RLS, it is a useful contribution to array processing because

¹ Dr. Salim Hanna, DOC (Canadian govt.), private communication

it is inherently general, stable and of low complexity, unlike existing RLS or FRLS algorithms. Specifically, the contributions of the FRQN algorithm developed in this thesis are:

- * Low computational complexity ($6N + 2$, only twice that of NLMS, compared to $4N^2 + 4N + 2$ for RLS);
- * Applicability to arbitrary array signal vectors (unlike present order-of- N FRLS);
- * Quasi-Newton adaptation (faster than NLMS and otherwise accelerated LMS algorithms, similar to but slower than RLS), specifically meaning that adaptation speed is robust with respect to condition of signal-scenario in that it is relatively insensitive to the eigenvalue spread of the array covariance matrix;
- * Lower steady-state misadjustment error than NLMS and lower total mean squared error (despite a higher tracking error) than NLMS even in very fast dynamic scenarios, comparable to RLS at low to moderately-high signal-to-noise ratios.

Adaptive arrays in a mobile-radio network base-station are used as the vehicle to develop and compare the FRQN algorithm in this thesis, as they possess a more general signal structure, require fewer degrees of freedom than time-domain equalizers to mitigate multipath and cancel co-channel interference, making them very popular systems in near-future mobile wireless networks.

1.4 Synopsis

The following chapters explain the origins and motivations of the proposed FRQN algorithm in greater detail. Chapter 2 presents the starting point, namely the LMS algorithm, in a tutorial format, followed by the RLS algorithm which, minus its complexity, represents the goal. With details of its derivation deferred to chapter 3, the FRQN algorithm and some of its salient properties are stated. Alternative approaches found in recent literature are reviewed for comparison and to more clearly define its place in the context of current research. Applications particularly suited to this algorithm are cited as motivation for its development.

Chapter 3 develops the FRQN algorithm in detail (equations (80) or (122), with (93)). Its properties are mathematically derived and explained by appealing to geometric concepts and compared to corresponding features of the LMS and RLS algorithms in section 3.2. Preliminary simulation results serve to illustrate the favourable performance of the algorithm relative to that of LMS in a simple adaptive antenna array (a generalized sidelobe canceller).

Further simulation results quantitatively compare their performances as fully adaptive antenna arrays in a mobile-radio environment. The mathematical treatment in section 3.5 indicates other possible versions of this algorithm, while section 3.6 characterizes key convergence properties and evaluates the performance of the FRQN as well as NLMS and RLS algorithms in stationary and non-stationary adaptive array applications. Section 3.7 shows that the FRQN algorithm requires only twice the computations per iteration as the NLMS algorithm, and slightly more bits to avoid stalling at the same minimum mean squared error. Appendix H includes the block diagram of a possible implementation. For completeness, and to improve continuity of the main text, detailed mathematical derivations are grouped into appendices at the end.

Performance properties of the three algorithms are summarized with reference to the key analytic and simulation results. Chapter 4 outlines possible ways of providing reference signals for the FRQN (or other) algorithm(s) in adaptive array applications, particularly in a high-capacity cellular or mobile personal communications network base-station. It is shown that the FRQN algorithm can be used with the same generic reference-generating subsystems that generate the reference signal from the array output for blind adaptation in the same way as the NLMS algorithm. Chapter 5 summarizes the work and its main contributions and draws the conclusions. It also develops a new interpretation of the mathematical relationship among the FRQN, RLS and NLMS algorithms.

To set the stage for the developments that follow, the context of the FRQN algorithm in terms of its general merits relative to those of the NLMS, RLS and stable FRLS algorithms is summarized in Table 1 below.

Table 1: Merits of Adaptive Algorithms

Algorithm ▶ Merit ▼	NLMS	RLS	FRLS [6]	FRQN
Complexity for N coefficients	$3N$	$4N^2$ + $4N$ + 2	$9N + 23$	$6N + 2$
Performance	Slow, erratic	Fast, accurate	Fast, Accurate	Moderately fast, accurate
Sophistication	General arrays, requires well- conditioned data	Optimal, general arrays	Requires delay- line data structure; not general arrays	General arrays

Chapter 2

Preliminaries

2.1 Principles of LMS and RLS Adaptive Algorithms - A Tutorial

Adaptive systems imply a vast field of research and development so the present discussion is restricted to the most common structures and algorithms. More involved discussions are available in the vast body of literature, e.g. [1],[3],[31],[34]. With only a negligible loss of generality, the structure considered for the present purposes is that of a linear combiner as sketched in Figure 1 below. It is always stable from the signal viewpoint, and, as mentioned in [3], can serve a variety of applications from equalizers to neural networks to antenna arrays. The last application is of primary interest herein.

Figure 1: Linear combiner structure.

Essentially, the linear combiner serves to compute the inner product of its complex coefficient N-vector W_k with the signal-vector Y_k to generate a scalar output signal y_k as

$$W_k^H Y_k = y_k \quad (1)$$

for the k-th iteration at time $t = k \Delta t$, where Δt is the sampling interval. Superscript H denotes the complex-conjugate transpose (Hermitian). As mentioned in chapter 1, the purpose of the coefficients is to compensate for some sort of time-varying signal-corruption, so they must be continually adapted. Without loss of generality, assume that a reference signal r_k is available and is highly correlated with the (uncorrupted) desired signal and negligibly correlated with the interfering signals and noise. It serves to guide the adaptation of the combiner coefficients or "weights" so that, in some average sense (to be clarified shortly), the error signal given by

$$e_k = y_k - r_k \quad (2)$$

is minimized. In the context of adaptive antenna arrays, the input vector consists of narrowband² signals received on each antenna element comprising the desired as well as interfering signals from various directions. Ideally the output of the combiner is only the desired signal. The quantity being minimized is also referred to as the penalty function. (Within each category of applications, the constraint or guiding reference signal may be obtained by various means, but the basic algorithm is analytically still the same, although the particular system applications may be known by different names, e.g. sidelobe canceller, noise canceller, fully-adaptive beamformer. A discussion of reference signals is contained in Chapter 4.) The adaptive coefficients which perform the minimization will also be referred to as the state variables of the system.

² Occupied signal bandwidth B , is much less than the reciprocal of the transit time of the receive signals across the array aperture.

The general problem is therefore to find an optimal set of combiner coefficients so as to minimize the penalty function which in the case of LMS is the mean square (magnitude of) error. (If the *instantaneous* error was to be zero, the signal would not convey any information.) That is

$$\underset{W}{\text{Min}} [P(W)] = \underset{W}{\text{Min}} [\mathcal{E}\{|e_k|^2\}] \rightarrow \text{mmse} \quad (3)$$

where *mmse* denotes the minimum mean square error. All signals are assumed to be ergodic so the expectation over the ensemble $\mathcal{E}\{\}$ is equivalent to the time-average over $\{k\}$, hereafter denoted by an overbar. One may apply the common approach to finding the extremum of a function of several independent variables, i.e. set its gradient to the zero vector:

$$\nabla_W \overline{|e_k|^2} = \nabla_W \overline{|W_k^H Y_k - r_k|^2} = 0 \quad (4)$$

Usually it is assumed that y_k and W_k are practically independent because the $\{W_k\}$ represent a much slower process than the $\{y_k\}$. The average, being a linear operation, is performed first, affecting only the "fast" processes according to

$$\nabla_W \overline{|e_k|^2} = \nabla_W \left[\overline{r_k r_k^*} - W^H \overline{Y_k r_k^*} - \overline{r_k Y_k^H} W + W^H \overline{Y_k Y_k^H} W \right] = 0 \quad (5)$$

The averaged quantities are identified as follows:

$$\sigma_r^2 = \overline{r_k r_k^*} \rightarrow \text{reference-signal power} \quad (6)$$

$$S = \overline{Y_k r_k^*} \rightarrow \text{steering or constraint vector} \quad (7)$$

$$R = \overline{Y_k Y_k^H} \rightarrow \text{array covariance matrix.} \quad (8)$$

Carrying out the operation using the complex gradient operator (see e.g. [34])

$$\nabla_w \triangleq \begin{bmatrix} \frac{\partial}{\partial w_{Re\ 1}} \\ \vdots \\ \frac{\partial}{\partial w_{Re\ N}} \end{bmatrix} + j \begin{bmatrix} \frac{\partial}{\partial w_{Im\ 1}} \\ \vdots \\ \frac{\partial}{\partial w_{Im\ N}} \end{bmatrix} \quad (9)$$

with $w = w_{Re} + j w_{Im}$ results in the so-called Wiener-Hopf or Normal Equations:

$$\nabla_w \overline{|e_k|^2} = R W - S = 0 \quad (10)$$

or, providing the inverse covariance matrix exists³, the optimal weight vector

$$\boxed{W_{opt} = R^{-1} S.} \quad (11)$$

Because in general the signal environment will not be stationary, it is necessary to keep adjusting the weights so as to maintain their optimality, i.e. towards the zero of

³R may be augmented by a small diagonal noise covariance matrix, or the pseudoinverse may be used [34].

the gradient, according to

$$\Delta W_{k+1} = W_{k+1} - W_k = -\beta \nabla_W \overline{e_k^2} . \quad (12)$$

This is the $k+1$ -st iteration of the so-called *gradient algorithm*, where β is a scalar parameter controlling the step-size of W in the direction of the negative gradient of the penalty function. Since R is positive-definite, *the gradient is zero at the minimum of $P(W)$, i.e. the minimum mean squared error*. For stability, the step-size parameter must generally be bounded as (see Appendix B)

$$0 < \beta \leq \frac{2}{\text{Tr}[R]} < \frac{2}{\lambda_{\max}} \quad (13)$$

where $\lambda_{\max}$ is the largest eigenvalue of R , and $\text{Tr}[R]$ is equal to the sum of all eigenvalues of R [34]. The algorithm will track the minimum of P as the signal environment slowly changes with time.

In practice, such a gradient computation cannot be implemented because, in principle, infinite averaging time is required to compute the signal-statistics R and S . In real-time adaptive systems, these must be estimated at each iteration k . Moreover, the estimates should be consistent in the sense that their ensemble averages should equal the actual corresponding statistics in equations (6)-(8) and (10).

A simple and consistent estimate of the whole gradient (10) where all these statistics appear is obtained by using the instantaneous error signal at each iteration as

$$\nabla_w \overline{|e_k|^2} = \nabla_{w_k} |e_k|^2 = Y_k Y_k^H W_k - Y_k r_k^* = Y_k e_k^* \quad (14)$$

which is of course *not* equal to the zero-vector for each k . Note that the *estimated gradient is also equal to the actual gradient of the square magnitude of the instantaneous error signal*. In either case the estimate is consistent, as may be verified by averaging the quantities in equation (14) the same way as was done in equation (5). The result is the popular so-called *steepest-descent or Least Mean Squares (LMS) algorithm, which at iteration k is given by:*

$$\Delta W_k = -\beta Y_{k-1} e_{k-1}^* \quad (15)$$

or equivalently, in a recursive weight-update form,

$$W_k = W_{k-1} - \beta Y_{k-1} (Y_{k-1}^H W_{k-1} - r_{k-1}^*), \quad (16)$$

with the available⁴ output at the same time given by

$$y_{k-1} = W_{k-1}^H Y_{k-1}. \quad (17)$$

Because each step ΔW_k takes the states of the combiner only a short way towards the optimal states, the whole journey incorporates many samples of the input signal vector Y via the recursion (16), so the LMS algorithm intrinsically performs averaging of the signal statistics. Since the signals are assumed ergodic, the averaged LMS algorithm is

⁴Since W_k is still being computed at the k -th iteration, $W_k^H Y_k$ is not an available output. To compute it would incur some $2N$ additional computations. Moreover, as the speed of NLMS is increased, that would not be a useful output because, due to the nature of the gradient estimate (comments regarding equation (14)), it becomes zero *instantaneously*, meaning that it follows r_k *exactly*, thus effectively destroying any information transmitted as part of Y_k .

seen to be the gradient algorithm. It therefore minimizes the squared error in an average sense, "average" implying over the *whole* adaptation time until the steady state of W (hence the name LMS).

Figure 2: Quadratic penalty-surface in W -space.

The step-size parameter is bounded by (13). Too large a step-size results in divergence because the coefficient step overshoots the direction of decreasing P (error power). Convergence time is governed by the actual step-sizes and directions along the penalty surface in W -coordinates. The steps are also proportional to the eigenvalues of R . Equivalently, the time-constants of these modes of convergence of W are inversely proportional to the respective eigenvalues. Clearly, the eigenvalues depend on the signal statistics and cannot be selected by the user. The average adaptation behaviour of W may be visualized as the descent of the tip of the W -vector along the steepest path (negative gradient) of the penalty surface. Figure 2 shows a sketch of such a surface; it is a quadratic function of the weights, corresponding to the bracketed quantity in (5). The W -

vector proceeds rapidly in the direction of the large arrows and slowly along the small arrows, always perpendicular to the contours of constant penalty magnitude. Depending upon the initial weights W_0 and the shape of $P(W)$, it may take a very long time to reach W_{opt} . Frequently such is the case in practice, as for example when two or more point sources of interference are separated by a small angle relative to the far-field beamwidth of the array aperture [34].

As may be evident from the preceding discussion, the direction of adaptation of W may not be toward W_{opt} at every instant. In fact, it is more likely to be in a different direction (except when the eigenvalues are all equal and/or W_0 is on one of the principal axes of the $P(W)$ ellipsoid). Algorithms which always adapt the weights in the optimum direction, i.e. along the shortest, straight line toward W_{opt} , are known as Quasi-Newton (QN) algorithms. They cause W_k to follow Newton's method of approaching the zero of the gradient function of the mean squared error [4,5,28]. Among them are the Fletcher-Powell "conjugate gradient" algorithms, variations of which can also be made order-of- N in complexity in principle, as they rely on past gradients to approximate the inverse of R (the Hessian) [28]. The convergence speed of QN algorithms is therefore independent of the eigenvalues of R , hence more controllable by the user and robust with respect to the signal environment.

One quasi-Newton algorithm is the so-called Recursive Least Squares (RLS) algorithm. *At every iteration it minimizes the sum of squared errors up to that iteration*, using all the past data up to the time $k \Delta t$ [31]. (In that respect it differs from LMS and the algorithm developed in this thesis because the latter two minimize a different penalty-function, namely the squared error intrinsically averaged over the run-times of these algorithms.)

Formally RLS solves the problem

$$\underset{W_k}{\text{Min}} \left[\sum_{q=0}^k \lambda^{k-q} |W_k^H Y_q - r_q|^2 \right] \quad (18)$$

which minimizes the sum of all squared errors from time $q=0$ to the present $q=k$ using the coefficients W_k for all q up to k . The squared errors are exponentially weighted in time by the "forgetting" factor

$$0 \leq \lambda \leq 1 \quad (19)$$

which governs the tracking properties of the algorithm [30]. As new data becomes available at each increment of q , a new set of coefficients is computed to again minimize the newly-incremented sum of squared errors up to the current time.

Minimization is again performed by setting the gradient to zero, but this time no estimation needs to be performed because the problem is not posed in terms of ideal statistics, but rather in terms of their best current estimates. Therefore, setting

$$\nabla_{W_k} \left[\sum_{q=0}^k \lambda^{k-q} |W_k^H Y_q - r_q|^2 \right] = 0 \quad (20)$$

results in

$$\sum_{q=0}^k \lambda^{k-q} Y_q r_q^* = \sum_{q=0}^k \lambda^{k-q} Y_q Y_q^H W_k \quad (21)$$

which is similar to the Wiener-Hopf equations. Indeed, because the signals are assumed to be ergodic, the weighted sums in (21) may be thought of as statistical averages using all available data up to time k , which are the best estimations that can be computed in real time.

Denoting

$$R_k = \sum_{q=0}^k \lambda^{k-q} Y_q Y_q^H \quad (22)$$

and

$$S_k = \sum_{q=0}^k \lambda^{k-q} Y_q r_q^* \quad (23)$$

it is seen that RLS solves the least-squares problem at every iteration k , since according to (21),

$$S_k = R_k W_k . \quad (24)$$

To compare the coefficient steps of RLS with those of LMS, note that according to (22) and (23), the current (estimates of the) statistics are obtained recursively from past estimates and current data as

$$R_k = Y_k Y_k^H + \lambda R_{k-1} \quad (25)$$

and

$$S_k = Y_k r_k^* + \lambda S_{k-1} . \quad (26)$$

(Because W_k is still being computed at the k -th iteration, the output signal will be denoted by $y_k = W_{k-1}^H Y_k$, which also avoids incurring $2N$ extra computations.) Now rewrite (24) using (26) as

$$R_k W_k = Y_k r_k^* + \lambda S_{k-1} , \quad (27)$$

then apply (24) at the previous $(k-1)$ -st iteration to S_{k-1} above, to obtain

$$R_k W_k = Y_k r_k^* + \lambda R_{k-1} W_{k-1} . \quad (28)$$

Isolate R_{k-1} in (25), substitute into (28) above and simplify to

$$R_k W_k = Y_k r_k^* + (R_k - Y_k Y_k^H) W_{k-1} \quad (29)$$

Subtracting $R_k W_{k-1}$ from both sides above results in the Quasi-Newton (QN) form of the algorithm:

$$\boxed{R_k \Delta W_k = -Y_k \alpha_k^*} \quad (30)$$

where α_k is the *a-priori error*, defined as

$$\alpha_k = W_{k-1}^H Y_k - r_k, \quad (31)$$

(as opposed to the *current instantaneous error*, the sum of which is being minimized) given at the same time by

$$e_k = W_k^H Y_k - r_k. \quad (32)$$

It looks much like the LMS algorithm (e.g. equation (16)) when rewritten as

$$\boxed{W_k - W_{k-1} = -R_k^{-1} Y_k (Y_k^H W_{k-1} - r_k^*)} \quad (33)$$

Note that, except for the indices of the signal quantities, the difference between RLS and LMS coefficient steps is that, on average, the magnitudes and directions of the former do not depend on the eigenvalues of R . That is because, on average, the (matrix) coefficient of W_{k-1} in (33) becomes the identity matrix for $k > 2N$:

$$\mathcal{E}\{R_k^{-1} Y_k Y_k^H\} = R^{-1} \mathcal{E}\{Y_k Y_k^H\} = R^{-1} R = I. \quad (34)$$

There is also no time-constant, so when sufficient number of linearly-independent samples from each of the N signal dimensions have been incorporated in the least-squares solutions (after $2N$ to $3N$ iterations), the algorithm converges and (33) becomes the optimal solution

$$W_k - W_{k-1} = -W_{k-1} + R_k^{-1} S_k. \quad (35)$$

It is required only that each successive sample of Y be linearly independent of the previous ones.

The QN form of the RLS algorithm is not usually what is implemented. Because in place of the scalar step-size parameter β , as in LMS, one now has the $N \times N$ matrix R_k^{-1} in its place in RLS, the computational complexity is of order N^3 due to the matrix-inversion. Usually the matrix-inversion lemma (see Appendix E) is employed on (25) to recursively update the array-covariance matrix inverse as

$$R_k^{-1} = [\lambda R_{k-1} + Y_k Y_k^H]^{-1} = \frac{1}{\tilde{\lambda}} \left[R_{k-1}^{-1} - \frac{R_{k-1}^{-1} Y_k Y_k^H R_{k-1}^{-1}}{\lambda + Y_k^H R_{k-1}^{-1} Y_k} \right] \quad (36)$$

thus reducing the computational complexity of the RLS algorithm to order of N^2 . When (36) is substituted into (33) it eventually simplifies to

$$\Delta W_k = - \left(\frac{1}{\lambda + Y_k^H R_{k-1}^{-1} Y_k} \right) R_{k-1}^{-1} Y_k (Y_k^H W_{k-1} - r_k^*) \quad (37)$$

where the first factor on the right-hand side is a scalar. Since $R_{k-1}^{-1} Y_k$ in (33) and $R_{k-1}^{-1} Y_k$ are vector quantities, one may denote them by

$$L_k = R_{k-1}^{-1} Y_k \quad (38)$$

and

$$V_k = R_{k-1}^{-1} Y_k \cdot \quad (39)$$

Defining the vector

$$K_k = \frac{V_k}{\lambda + Y_k^H V_k} \quad (40)$$

allows one to write the recursive update of the matrix inverse as

$$R_k^{-1} = \frac{1}{\lambda} R_{k-1}^{-1} - \frac{1}{\lambda} K_k V_k^H \quad (41)$$

and that of the weights as

$$W_k = W_{k-1} - K_k \alpha_k^* \quad (42)$$

which require order of N^2 operations per iteration. For a faster (order of N) RLS algorithm, one wants to find a simple recursive update relation for L_k in terms of L_{k-1} . So far, a stable way of doing it or its equivalent has been found only for a time-domain FIR type of filter structure [6].

2.2 Relations to Continuous Time Algorithms - The Normalized LMS Algorithm:

It is instructive to develop the discrete-time LMS algorithm from its continuous-time counterpart and to relate that to the conventional discrete-time LMS and normalized LMS (NLMS) algorithms. It helps in understanding the FRQN algorithm and its possible variants.

The conventional LMS algorithm in discrete-time is given by

$$W_k = W_{k-1} - \beta Y_{k-1} e_{k-1}^* \quad (43)$$

where

$$e_{k-1} = W_{k-1}^H Y_{k-1} - r_{k-1} \quad (44)$$

is the past error, r is the corresponding reference signal (correlated with the desired signal and uncorrelated with the interfering signals) and Y is the input signal vector. Note that the past error is utilized in a *causal* updating relation for the current weights vector

(i. e. at the k -th time-interval). This may also be written in terms of the resulting step from W_{k-1} to W_k as

$$\Delta W_k = -\beta Y_{k-1} Y_{k-1}^H W_{k-1} + \beta Y_{k-1} r_{k-1}^* = -\beta Y_{k-1} e_{k-1}^* . \quad (45)$$

This algorithm is stable as long as $0 < \beta < 2/Tr[R]$ where the quantity $Tr[R] = Tr\{E\{YY^H\}\} = \overline{Y^H Y}$, is the input signal power.

It has been suggested that a better version of the LMS algorithm should use the *current* signals, because the conventional LMS algorithm introduces an extra delay by using the past signals [22-24]. This results in a normalized LMS (NLMS) algorithm, which may be derived alternatively as in [31]. Here it is derived from the continuous-time LMS algorithm to show how various versions of the FRQN algorithm can be obtained.

In continuous-time the extra delay in the LMS algorithm vanishes in the limit as $\Delta t \rightarrow dt$. It is given by the differential equation

$$T \frac{d}{dt} W(t) = -g Y(t) Y^H(t) W(t) + g Y(t) r^*(t) = -g Y(t) e^*(t) \quad (46)$$

where T is the open-loop averaging time-constant and the gain is required to be $g > 0$ for stability. The LMS algorithm minimizes the mean square error on average, but at each instant its direction of change of the weight vector is that which reduces the *instantaneous* squared error. That is because it uses the instantaneous gradient of the squared error (which is also the gradient of the instantaneous squared error) to estimate the true gradient of the performance surface. The algorithm itself performs averaging intrinsically by its recursive nature, and the fact that each incremental change in W advances it only a short way to the final solution.

To obtain the normalized discrete-time LMS algorithm, equation (46) is first averaged with the usual assumptions of ergodicity and independence of $W(t)$ from $Y(t)$, resulting in

$$T \frac{d}{dt} W(t) = -g R W(t) + g S. \quad (47)$$

This is then transformed to the Laplace domain and written as

$$s T W(s) = -g R W(s) + g S(s). \quad (48)$$

At this point one of a number of s-to-z transformations (see Appendix A) may be applied to convert it into the z-domain, e. g. the bilinear transformation

$$s = \frac{2}{\Delta t} \left(\frac{1 - z^{-1}}{1 + z^{-1}} \right), \quad (49)$$

or the backward-difference transformation

$$s = \frac{1 - z^{-1}}{\Delta t}. \quad (50)$$

Choosing the latter with Δt being the Nyquist sampling interval and transforming back into the now discrete-time domain produces

$$\tau \Delta W_k = -g R W_k + g S \quad (51)$$

where $\tau = T/\Delta t$ is the normalized open-loop time-constant. Upon "deaveraging" and assuming that the signals or data represented in R and S include the current data at time $t = k\Delta t$, this becomes

$$\tau \Delta W_k = -g Y_k Y_k^H W_k + g Y_k r_k^* = -g Y_k e_k^* \quad (52)$$

which compares with (45). However, this is not a causal updating expression for W_k because e_k also contains W_k . To obtain a useful updating algorithm for W_k , one again makes use of the matrix-inversion lemma (see Appendix E) to isolate it as

$$[I + \mu Y_k Y_k^H] W_k = W_{k-1} + \mu Y_k r_k^* \quad (53)$$

where

$$\mu = \frac{g}{\tau} = \frac{g \Delta t}{T} > 0. \quad (54)$$

This leads to the updating equation (which compares with (16))

$$W_k = W_{k-1} - \beta_k Y_k Y_k^H W_{k-1} + \beta_k Y_k r_k^* \quad (55)$$

with the time-varying step-size parameter given by

$$\beta_k = \frac{\mu}{1 + \mu Y_k^H Y_k}. \quad (56)$$

Since W_k is still being computed at this (k -th) iteration, the available output is given by

$$y_k = W_{k-1}^H Y_k \quad (57)$$

to avoid the problems pointed out in connection with the more conventional form of the LMS algorithm. The indexing of Y and y in the two algorithms is not identical. It can be shown that if β in the conventional LMS algorithm is normalized by $\text{Tr}[R]$, the outputs of both LMS and NLMS are identical except for that of the latter occurring one iteration earlier. Normalization is merely a convenient way of ensuring that (13) is

satisfied, so there is effectively only one normalized LMS algorithm (NLMS), contrary to implications of [22-24]. In this thesis, the NLMS form as derived from continuous-time LMS will be utilized, since it most closely resembles the convention utilized in the FRQN algorithm, making comparisons easier.

Note that on average, β_k is a step-size parameter normalized by the signal power, thus satisfying (13). Consequently (56) gives unconditional stability since the gain is usually $g > 1$. (One would also expect to preserve this kind of stability by the nature of this s-to-z transformation, as shown in Appendix A.) Note that (56) is naturally prevented from becoming infinite when the signal power happens to be zero. (In practice, finite noise power would prevent that.)

Also note that, although (55) minimizes the current error with the current weights according to (52), it does not actually use the current error in computing the current weights, as that is not causally possible. Instead, it uses the a-priori error, (as does RLS), which also automatically avoids the problem of instantaneous following of the reference as its speed is increased, according to (55).

The RLS algorithm also minimizes the present error using the a-priori error. One type of comparison of algorithms should involve NLMS optimized with respect to speed and misadjustment. The common step-size parameter for achieving this in the conventional LMS algorithm is

$$\beta_o = \frac{1}{Tr[R]} .$$

It makes the excess mean square error equal to the minimum mean square error [31]. Reference [26] contains a derivation for the corresponding condition in the continuous-time LMS algorithm from which it can be deduced that

$$\frac{G}{T} \frac{\delta}{2} \text{Tr}[R] = 1 \quad (58)$$

where $G=g$ and $\delta/2 = \Delta$ is the Nyquist sampling interval. This corresponds to

$$\frac{g}{\tau} = \mu = \frac{1}{E\{Y_k^H Y_k\}} \quad (59)$$

in (55). It leads to an average step-size parameter via (56) given by

$$\beta = \frac{1}{2 \text{Tr}[R]} , \quad (60)$$

which is close to the one typically used in the conventional discrete-time LMS algorithm.

Therefore the corresponding optimal open-loop gain in (54)-(56) should be

$$g = \frac{\tau}{\text{Tr}[R]} , \quad (61)$$

or with τ determined as $4N$ from the requirements for Nyquist-rate sampling and sufficient correlation time (to be discussed in chapter 3)

$$g = \frac{4N}{\text{Tr}[R]} . \quad (62)$$

2.3 The FRQN Algorithm in Relation to Similarly-Motivated Approaches

- A Review of Recent Literature:

This algorithm owes its genesis to the work of Dr. R. T. Compton, Jr. of Ohio State University. He published in 1980 a modification for the analogue, continuous-time LMS algorithm implementation popular for adaptive arrays at the time, which cancelled most of the dependence of its time-constants on the covariance-matrix eigenvalues [4]. He pointed out in his introduction, that it was not intended as a digital discrete-time

computational algorithm, because other such algorithms which circumvented the problem of eigenvalue-dependent convergence properties already existed. Presumably he had in mind block-wise matrix-inversion (with order-of- N^3 complexity), and recursive least squares (with order-of- N^2 complexity). Present research and development efforts are directed toward achieving order-of- N complexity.

With the original intent of [4] in mind, the present author modified Compton's analogue loops to more exactly cancel their eigenvalue dependence in the time-constants, and confirmed its constant convergence properties by analysis and hardware experimentation [5]. It has been successfully implemented in some products for military communications applications. These depended on its constant-convergence performance in that it served to protect digital radio links which could not tolerate lengthy jamming intervals without loss of synchronization, or other links which also used fast frequency-hopping to evade jamming. The analogue implementation had the advantage of relatively low complexity and real-time wideband performance. For some types of algorithms, this is still the case today [10].

By applying the matrix-inversion lemma to the finite-difference counterpart of the original second-order differential equation of [5], a discrete-time version of the algorithm was created. However, for stability reasons, the original block diagram now took on the form of a non-computable discrete-time network. This meant that a digital hardware implementation would have to be of a different topology than that of the original continuous-time servo loops, in order to be causal.

The basic version of this (FRQN) algorithm is written as

$$\begin{aligned} \tau_1 \tau_2 (\mathbf{W}_k - 2\mathbf{W}_{k-1} + \mathbf{W}_{k-2}) + \tau_2 [\gamma \mathbf{I} + \mathbf{g} \mathbf{Y}_k \mathbf{Y}_k^H] (\mathbf{W}_k - \mathbf{W}_{k-1}) = \\ = -\mathbf{g} \mathbf{Y}_k \mathbf{Y}_k^H \mathbf{W}_k + \mathbf{g} \mathbf{Y}_k \mathbf{r}_k^* \end{aligned} \quad (63)$$

where k denotes the k -th iteration, $\mathbf{g}$ is a loop-gain factor; $\mathbf{W}$ is the N -vector of complex weights or adaptive coefficients, $\mathbf{Y}$ is the N -vector of complex input or data signals and τ_1 and τ_2 are time-constants normalized by the sampling interval Δt , the latter determining the overall convergence time, and the former being much smaller. The scalar $\mathbf{r}$ represents a reference signal, or a main-channel signal as in the case of a (generalized) sidelobe canceller. Gamma (γ) is a diagonal augmentation term of the input signal covariance matrix.

It was shown in [5] that the original analogue version of this algorithm follows the Newton-Raphson method of minimization of the penalty-function, according to a first-order analysis. It is preferable to the gradient-descent approximated by the LMS algorithm because the former always converges at a fixed and selectable overall rate, whereas the latter is exceedingly slow and jittery in many practical situations [1].

The FRQN algorithm can have several versions, and in fact differs from the original discrete-time version originally developed for simulation of the analogue continuous-time implementation. That one also had two time-constants, but the smaller one was limited to be greater than about $8N$ due to the requirement for Nyquist-rate sampling combined with that of sufficient averaging time for decorrelation of N different signal vectors. With some modifications, the FRQN algorithm runs well with the smaller time-constant set to N and the larger one to about $4N$, and achieves much greater suppression ratios than the original. Other versions can be derived by applying different

s-to-z transformations to the Laplace-transform of the averaged original differential equation in [5]. (Such an approach is also applicable to other continuous-time stochastic adaptive algorithms.)

By specific references to recent papers on similarly-motivated improvements of LMS and RLS algorithms, the unique properties of the FRQN algorithm will next be illustrated. First, the difference equation for the basic (not optimized) version of the FRQN algorithm is rearranged in the form of a programmable, recursive weight-update equation as

$$\begin{aligned}
 W_k = & \left[\frac{a_1}{a_0} I + \left(\frac{b_1/b_0 - a_1/a_0}{a_0/b_0 + Y_k^H Y_k} \right) Y_k Y_k^H \right] W_{k-1} - \\
 & - \left(\frac{\tau_1 \tau_2}{a_0} \right) \left[I - \frac{Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] W_{k-2} + \\
 & + \left(\frac{g}{b_0} \right) \frac{Y_k r_k^*}{a_0/b_0 + Y_k^H Y_k}
 \end{aligned} \tag{64}$$

with $a_0 = \tau_2(\tau_1 + \gamma)$, $b_0 = g + \tau_2 g$, $a_1 = 2\tau_2(\tau_1 + \gamma/2)$ and $b_1 = g\tau_2 = b_0 - g$. The parameters were chosen as $\gamma = 1/\tau_2$, $g = 1/\lambda_{\min}$, $\tau_2 = 4N$ and $\tau_1 = \tau_2/4$. Since $Y_k^H W_{k-1}$ and $Y_k^H W_{k-2}$ are scalars, it is clearly of order N . Its derivation and refinements are postponed until chapter 3.

The FRQN algorithm is numerically stable and works with arbitrary array geometries and coupling structures. Initial comparisons in the same scenarios with LMS (actually NLMS) showed dramatic improvements of convergence rates and weight jitter in the ill-conditioned cases.

Perhaps the best-known numerically stable FRLS type of algorithm is that of reference [6], but it is not applicable to adaptive arrays, because their data vectors do not possess the requisite shifting property. That is true even for perfectly regular array geometries such as uniformly spaced elements along a line without mutual coupling. The technique of reference [7] is applied specifically to adaptive arrays but requires a specific (uniform, linear) array geometry. In addition, there are concerns that its initialization does not guarantee global convergence. It relies on infrequent updates of a spatial pre-filter or beamformer, which are of order-of- N^2 complexity. A related reference [8] addresses the convergence problem, but still relies on the special array geometry. It also refers to Compton's work [4], citing that the convergence is still slow for the smallest eigenvalues. Mention is also made of a fast adaptive-array algorithm operating on a cascade structure known as a Davies tree, but it also requires a specific array geometry (either uniform linear or circular with a phase-mode network) [9]. That technique does not always converge to the optimal state, and the computation is of order N^2 . Like other techniques relying on specific array geometries, the accuracy of the interference cancellation is always compromised by the inevitable phenomenon of mutual electromagnetic coupling among the array elements.

As mentioned earlier, reference [10] describes an analogue implementation of a neural network adaptive-array algorithm. It is rather fast (10 degrees of freedom adapt in 0.1 ns) and cites digital LMS implementations as being slow and suffering from signal-cancellation (due to weight jitter [1]). It is similar to [5] only in that the response time is determined solely by resistance-capacitance (RC) time-constants rather than by computational complexity (which is of order N^2). Being analogue, it does not benefit from the inherent precision and flexibility of digital implementations as the FRQN algorithm does.

A common approach to improving the convergence properties of the LMS algorithm is to vary its step-size, as for example in references [11] and [19]. That can modify the steepest-descent path only in a scalar fashion, i.e. it cannot change its shape to effect a shorter, more direct path to the optimum state. Usually the step-size is varied from large values for fast initial convergence to small values for low steady-state misadjustment and negligible signal-cancellation. It cannot achieve the homogenization of all modal time-constants as in Newton-type algorithms, which are regarded as the most "efficient" in this sense [1], because that requires a matrix operation on the gradient estimate. It is also not always clear when the adaptation is in the "initial" stages and when it is in more "steady" stages in an arbitrary dynamic scenario, i.e. it has little relevance to non-stationary scenarios (footnote 5 of [3]), unlike the FRQN algorithm.

A somewhat more potent approach is to use a diagonal matrix in place of a scalar step-size in an LMS algorithm, as for example in [15]-[18]. In [15] the authors use a fixed diagonal matrix to keep the computational complexity low. It is calculated from the known signal structure, in this case measured echo characteristics of a conference room. Being diagonal and deterministic, this gradient transformation matrix also cannot achieve the homogenization of the LMS modal time-constants. References [16]-[18] are concerned with a varying diagonal step-size matrix, which effectively performs an automatic gain control function on the components of the estimated gradient. That gets around the problem of distinguishing initial adaptation stages from later ones, but its authors acknowledge some of the comments to the effect that it may not work with long filters [18]. In principle, it still cannot achieve a Newton-type of convergence to the optimum because the step-size matrix is diagonal, whereas the ideal step-size matrix is the inverse of the array covariance matrix (as in the RLS algorithm, effectively) which is rarely if ever diagonal.

Attempts at effecting non-diagonal step-size matrices in LMS algorithms sometimes follow the approach of transforming the data vectors so the covariance matrix of the transformed data would become more diagonal. That may be done by passive array feed networks or beamformers, but does not work in general, because the required transformation is data-dependent and the beamformer is constrained to be fixed. Some improvement may result from using power normalization of the beamformer outputs as in [20], but that cannot generally be relied upon to eliminate the eigenvalue disparities. To quote from the abstract of that reference: "Since the Karhunen-Loève transform (KLT) is the ideal transform for this application, and since the KLT is defined in terms of the statistics of the input signal, it is certain that no fixed-parameter transform will deliver optimal learning characteristics for all input signals. However, the simulations suggest that, with a little trial and error, transforms can be found which give much improved performance in a given situation." [20]. The FRQN algorithm does not have such limitations.

Algorithms which inherently orthogonalize the data and thus effectively equalize all the modal convergence rates are studied in [21] in the context of adaptive equalizers. Fast quasi-Newton and conjugate gradient algorithms are quoted as being profoundly affected by extremely small amounts of noise, even to the point of instability in the case of the conjugate gradient method. The self-orthogonalizing algorithms studied in that paper all have a computational complexity of order N^2 and are applied only to tapped-delay-line or FIR types of input signals. The FRQN algorithm is insensitive to noise in the data and is computationally simpler.

Less accurate but computationally simpler approaches to self-orthogonalization are described in [22]-[24]. Actually, reference [22] does not claim to perform self-orthogonalization, but rather to enhance the stability of the LMS algorithm by using the

current weights and data in the weight-update equations. Reference [23] shows how to perform the corresponding weight-vector computations without having to compute the vector entries one at a time while holding the others fixed. It is essentially the same idea as the "noncomputable loop" for the LMS algorithm and may also be derived by application of the matrix-inversion lemma to that kind of loop. That was done in the process of developing the FRQN algorithm. In the case of the LMS algorithm, it has a similar effect as division of the step-size by the total input power, which accounts for the unconditional stability of the resulting MLMS algorithm. However, it does nothing to reduce its eigenvalue disparity, and provides the *same* output as the conventional LMS algorithm, but one iteration earlier. The same idea is responsible for the unconditional stability of the FRQN algorithm. However, the FRQN algorithm also employs other features to effectively achieve self-orthogonalization, i.e. by normalizing its step-size with a higher-rank operator than the scalar signal power. The approach of [24] is to use reduced-rank estimates of this operator as a way of generalizing the power normalization idea. Typically the rank is 1 or 2. While that is clearly suboptimal, its author has some concern that the "implementation complexity might be high." The optimal operator is the inverse of the (up-to-date estimate of the) input covariance matrix, which is closely approximated in the FRQN algorithm without excessive complexity.

The FRQN algorithm is similar to a second-order LMS algorithm described in [26], in the sense that it is also a second-order algorithm with similar complexity. When the gain of the second-order LMS algorithm is such that all its modes converge with decaying oscillations, their envelopes have the same time-constant which is independent of the eigenvalues. The oscillation frequencies, however, are directly related to them, which may compromise its stability with real-life stochastic signal. (Subsequent investigations during the course of this work showed it to be unstable in discrete-time simulations in most scenarios.) Reference [26] was mostly concerned with analyzing its

misadjustment under averaging and independence assumptions, and found it to be about half that of LMS without the oscillatory behaviour (strangely enough).

Other adaptive algorithms which approach the convergence properties of RLS keep appearing in the literature e.g. [25], but they are generally similar in some respects to those already mentioned. Those in [25] have order-of- N^2 complexity. Others involve a variable step-size parameter and an algorithm for adjusting it. In [29] the LMS algorithm incorporates an embedded gradient algorithm to adjust the step-size parameter effectively in proportion to the square magnitude of the gradient in the original LMS algorithm. It minimizes the change in the mean square error. While it reduces the steady-state excess mean square error below that achieved by the LMS algorithm, it really is no faster. Typically it starts with a large step-size and adapts quickly (mostly in a non-optimal direction), then reduces its step-size and slows down near the end of adaptation to minimize coefficient jitter. A refinement of it uses a separate step-size parameter for each gradient component, in effect a diagonal-matrix variable step-size parameter. By contrast, the FRQN algorithm adapts the *actual step-size and direction optimally*, toward the Newton path, by an amount proportional to the difference of the coefficient step from the optimal Newtonian one. As will be seen in a geometric interpretation, this happens relatively early in the adaptation process and is an intrinsic property of the whole algorithm. Thus the stability of both the coefficient adaptation and coefficient-step adaptation is assured.

Reference [32] describes an algorithm which is applicable only to FIR time-domain filters, as it explicitly depends on their structure for its computational advantages over the RLS algorithm. Unlike the FRQN algorithm, it will not work for general adaptive arrays. Reference [33] discusses LMS-type algorithms with diagonal-matrix variable step-size parameters, but with sign-change driven algorithms for adjusting them.

It is not clear that speed is increased, because all the simulation results show the LMS algorithm using the smallest step-size parameter while the innovative ones are started with larger ones, which tend to the smallest LMS one at steady state. The simulation results show the variable step-size algorithms to be little more than twice as fast as the LMS algorithm and to possess the same steady-state errors. No decrease in misadjustment is shown either.

Rather than "Newtonizing" the LMS algorithm, [36] takes the approach of simplifying the RLS algorithm by using only the diagonal entries of the estimated inverse covariance matrix in equation (36). It is thus equivalent to what may be the optimal way of controlling the step size of each weight component in the LMS algorithm. Results in [36] show only small improvements in the learning curves. An improvement in array pattern convergence was shown only for what turns out to be a well-conditioned scenario where the LMS algorithm is known to do well.

2.4 Applications and Further Motivation

The FRQN algorithm is intended for time-critical digital applications, where its convergence time must be known to lie within designated bounds and preferably constant and short. Compton's original motivation was to improve the convergence properties of the then-popular analogue LMS algorithm, because in his experience it had a disappointingly low dynamic range, in the following sense [4]: In a typical aircraft communications application, an LMS adaptive array frequently gives rise to a highly ill-conditioned array covariance matrix. The ratio of its highest to lowest eigenvalues is in the order of the signal- or jammer-to-noise ratio, and so is the ratio of its modal time-constants. System requirements dictate a maximum allowable response (overall convergence) time for tracking signals and interference, and also specify a modulation

bandwidth which must be protected from jamming. Consequently the slowest modes must converge faster than the maximum allowed response time, yet at the same time the fastest modes must not converge faster than some multiple (say 5) of the reciprocal modulation bandwidth, to prevent distorting the signals [4]. In Compton's experience, these requirements are frequently impossible to meet with LMS adaptive arrays over the dynamic range of signals encountered in practice. That is, either the slowest modes are too slow or the fastest ones are too fast, so one needs to reduce the dynamic range of their time-constants or to eliminate their dependence on the eigenvalues. This is equally true in more modern digital implementations.

Other applications of the LMS algorithm also encounter this difficulty, hence the ongoing research efforts to devise algorithms which have a fixed, controlled convergence and low complexity, as evidenced by the numerous references. The main virtue of the LMS algorithm is its order-of-N simplicity of implementation. Recently, numerically stable fast recursive least squares (FRLS) algorithms with order-of-N complexity have been devised for adaptive FIR filters [6,12]. However, they rely upon the shift property of the data signal-vector which gives the covariance matrix a highly regular structure. This is possible only in applications having the FIR filter data structure, which is not the case with adaptive arrays. Therefore, more specific applications of the FRQN algorithm are in time-critical digital adaptive arrays.

For example, adaptive arrays of this kind can be used to improve the spectrum efficiency, hence increase the capacity of, new digital cellular mobile radio communications systems [13,14]. These typically employ digital angle-modulation schemes and either TDMA (time-division, multiple-access) or CDMA (code-division, multiple-access) signalling formats which require synchronization, hence are time-critical. The "job" of the adaptive array is to cancel interference from other cell-sites where the

same frequencies are reused, and to perform diversity combining of desired signals so as to minimize multipath fading. They would typically be installed at the base-stations as receiving antennas and employ reference signals for optimal adaptation. In reference [14] Winters explains the dynamic range limitations of an LMS adaptive array in such an application, where a secure spread-spectrum reference signal is used. As a possible solution, he mentions an early form of Compton's work [4]. Being a digital extension of Compton's work, the FRQN algorithm is directly applicable here. Another application also investigated by Winters recently is for the North-American digital cellular system IS-54 [13], where he found the LMS algorithm incapable of tracking signals and interference even from slowly-moving vehicles. To quote from his abstract, "Results...show that the LMS algorithm has large tracking loss for vehicle speeds above 20 mph, but the DMI (direct matrix inversion) algorithm can acquire and track the weights to combat desired-signal fading and suppress interference with close to ideal tracking performance at vehicle speeds of up to 60 mph." [13]. Assuming, as Winters does, that the RLS algorithm has performance equivalent to that of DMI, the FRQN algorithm is also directly applicable in such a scenario, and is expected to yield similar tracking improvements but with only order-of-N complexity.

One application that came to mind as being particularly suitable for the FRQN algorithm is the base-station antenna system of a high-capacity mobile radio network. Due to the high capacity, the volume of signal-processing dictates a digital implementation. The concept is discussed more fully in chapter 4 on reference signals; see Figure 27 in that chapter.

The basic idea is that, in a mobile-radio network such as the cellular telephone system, each mobile would transmit exactly the same message simultaneously on two carrier frequencies (e.g. narrowband FM channels). At the base-station, an adaptive

array and a fixed-pattern antenna would each receive one channel and convert both of them synchronously to the same IF or to baseband. The signal from the fixed-pattern antenna would serve as the reference signal for the adaptive array which receives the same message on the other frequency of the transmitted pair. At each of the frequencies in the pair, signals from other mobiles would also be received, but only the message transmitted simultaneously on both of them would be correlated among them: the interfering signals would not correlate among the two channels when converted to baseband. It is a situation of "forced" spectral self-coherence. There being a total of $M(M-1)/2$ possible unique pairs of frequencies in a set of M frequency-channels of a cell, this scheme potentially allows $(M-1)/2$ times more users to share the same spectrum. The first frequency is used $M-1$ times, the second $M-2$ times, and so on (see Table 7 in chapter 4). The adaptive array forms optimal beams on all mobiles, nulling out all co-channel users (and other interference) but the intended one corresponding to the correlated message it sends on the other channel to the fixed-pattern antenna. The co-channel interference present in the fixed-pattern antenna's receivers (reference signals) contributes to the minimum mean squared error (mmse). However, only the *excess* mean squared error (emse) contributes to the output noise. It is shown in Appendix D that the FRQN algorithm has an emse about M times smaller than that of the NLMS algorithm.

For transmitting, the weights of the receiving array processors would be used to form orthogonal beams on the mobiles in a transmit band very close to the receive band. (A passive beamformer could be used with the array to make the weights less variable with frequency. Alternatively, transmit signals could be duplexed on the *same* frequencies in time.) This prompted the author to call the system "TRAAAP" (Transmit-Receive Adaptive Antenna Array Processor). It is a form of SDMA (Spatial-Division, Multiple-Access) scheme similar to that proposed by [37] for certain cyclostationary signals. TRAAAP has similar advantages in capacity (see chapter 4). It requires no special

synchronization or signalling format such as in TDMA or CDMA for reference generation and no additional base-stations to increase capacity as in cell-splitting. Handsets would be inherently compatible with existing single-channel-per-user (AMPS) base-stations by transmitting on one carrier by default. The FRQN algorithm offers the required low misadjustment and computational complexity, as well as fast adaptation and robustness desirable in a high-volume digital implementation.

In TRAAAP and applications such as proposed in [37], as well as GSM mobile radio networks [38], tracking ability of the adaptive algorithm is not as important as is its ability to converge in a limited amount of time and with small misadjustment. The FRQN algorithm was found to possess these attributes to a significantly greater extent than the LMS algorithm. Combined with its computational simplicity, they further motivated its development and characterization in this thesis.

Chapter 3

Algorithm Development and Characterization:

3.1 Development of the FRQN Algorithm and Possible Variants

In short, the FRQN algorithm evolved from Compton's algorithm [4] (see also equation (72)) by first introducing a diagonal augmentation term into the matrix coefficient of $W(t)$ resulting in the analogue continuous-time architecture of [5]. The motivation for this was to complete the cancellation of the dependence of the closed-loop modal time-constants on the eigenvalues of R . This was eventually transformed into discrete-time and rearranged. However, the severe effects of insufficient averaging time relative to those in Compton's algorithm became apparent when the time-constants were reduced in attempts to speed it up.

Before presenting the derivation of the present version of the FRQN discrete-time algorithm, it is instructive to examine the effects of insufficient averaging times in the original continuous-time versions [4], [5].

The effects of insufficient averaging time may be shown with the help of a modified final-value theorem of Laplace transforms, applied to the original continuous-time versions of the algorithm. The final-value theorem of Laplace transforms states that

$$\lim_{t \rightarrow \infty} W(t) = \lim_{s \rightarrow 0} s W(s). \quad (65)$$

In the modified, so-called "late-value" theorem, one simply lets s be small in $sW(s)$, thereby allowing some approximations, then divides out s and transforms back to $W(t)$. Small s implies large t . At large values of t , some averaging has taken place

intrinsically within the adaptive algorithm so the higher-frequency terms are written as averaged, with their usual independence from the (lower-frequency) state variables assumed. To see the effects of insufficient averaging, one simply omits this intrinsic averaging in the approximated late-value algorithm expression.

Applying this to the original continuous-time algorithm corresponding to [5], i.e.

$$T_1 T_2 \frac{d^2}{dt^2} \mathbf{W}(t) + T_2 \left[\frac{1}{\delta} I + \mathbf{g} \mathbf{Y}(t) \mathbf{Y}^H(t) \right] \frac{d}{dt} \mathbf{W}(t) + \delta \left[\frac{1}{\delta} I + \mathbf{g} \mathbf{Y}(t) \mathbf{Y}^H(t) \right] \mathbf{W}(t) = (\delta \mathbf{g}) \mathbf{Y}(t) \mathbf{r}^*(t) \quad (66)$$

with $\delta = T_2 / (T_2 + T_1) \approx 1$ since $T_2 \geq 5 T_1$, one employs the usual treatment of ensemble-averaging and diagonalizing before transforming to the Laplace domain, leading to

$$T_1 T_2 s^2 \mathbf{v}_n(s) + T_2 \left(\frac{1}{\delta} + \mathbf{g} \lambda_n \right) s \mathbf{v}_n(s) + \delta \left(\frac{1}{\delta} + \mathbf{g} \lambda_n \right) \mathbf{v}_n(s) = (\delta \mathbf{g}) \mathbf{b}_n(s) \quad (67)$$

where λ_n is the n-th eigenvalue of $\mathbf{R}$, $\mathbf{V}(s) = \{\mathbf{v}_n(s)\}$ is the transformed weights vector and $\mathbf{b}_n(s)$ is the n-th component of the transformed S-vector. The late-valued, n-th diagonalized- weight component in the Laplace domain is then found as

$$\lim_{s \rightarrow \text{small}} s \mathbf{v}_n(s) = \frac{(\delta \mathbf{g}) s \mathbf{b}_n(s)}{s T_2 (1 + \mathbf{g} \lambda_n) + (1 + \mathbf{g} \lambda_n)} \quad (68)$$

where $\mathbf{g}$ is chosen so $\mathbf{g} \lambda_n \gg 1$, and $\delta \approx 1$ is assumed. The effect is seen to be the disappearance of the second-order time-derivative term in the corresponding time-domain equation (as if $\tau_1 \rightarrow 0$), which for "late" values of t becomes

$$T_2 \left[I + g \overline{Y(t) Y^H(t)} \right] \frac{d}{dt} W(t) = - \left[I + g \overline{Y(t) Y^H(t)} \right] W(t) + (g) \overline{Y(t) r^*(t)} \quad (69)$$

(This corresponds to the steady state in the geometric interpretation in the next section.)

When the averaging is omitted, one can apply the matrix-inversion lemma (Appendix E)

$$\left[I + g Y(t) Y^H(t) \right]^{-1} = I - \frac{Y(t) Y^H(t)}{Y^H(t) Y(t) + 1/g} \quad (70)$$

to rearrange this so it becomes

$$\frac{d}{dt} W(t) = \frac{-1}{T_2} W(t) + \frac{g}{T_2} \left(\frac{1}{1 + g Y^H(t) Y(t)} \right) Y(t) r^*(t) \quad (71)$$

Clearly, when $dW(t)/dt = 0$, this does not converge to the correct (Wiener) solution for W . In effect, the absence of averaging incurs a severe misadjustment.

On the other hand, Compton's algorithm without averaging can be written directly from his formulation in [4] as

$$\left[I + 2kc Y(t) Y^H(t) \right] \frac{d}{dt} W(t) = -2k Y(t) Y^H(t) W(t) + 2k Y(t) r^*(t) \quad (72)$$

where k corresponds to gain $g/(2T_2)$, c is a separate time-constant. By the same rearrangement procedure, this simplifies to

$$\frac{d}{dt} W(t) = -Y(t) Y^H(t) W(t) \beta(t) + Y(t) r^*(t) \beta(t) \quad (73)$$

with

$$\beta(t) = \frac{2k}{1 + 2kY^H(t)Y(t)} . \quad (74)$$

This is effectively the normalized LMS algorithm (NLMS) which will converge as close to the optimal (Wiener) solution as the old LMS algorithm [23].

For sufficient averaging time to give reliable correlation estimates and thus to promote quasi-Newton convergence behaviour, Compton derives the criterion in [4] which restricts $T_1 \geq 2N/B$, where B is the bandwidth occupied by the signals in the N -dimensional complex ($2N$ real) vector $Y(t)$. Coupled with the need for Nyquist-rate sampling of those signals in a fully digital (i.e. software) algorithm implementation whereby $2B \leq 1/\Delta t$ (see Appendix A), this leads to

$$\frac{1}{\Delta t} \geq 2B \geq \frac{4N}{T_1}$$

or equivalently to

$$\boxed{\frac{T_1}{\Delta t} = \tau_1 \geq 4N} . \quad (75)$$

However, as Compton points out in [4], smaller averaging times may suffice to correctly estimate the rank of R , suggesting that the degeneration to an LMS algorithm with decreasing T_1 may be rather "graceful".

Realizing that the original loop type of architecture was no longer a constraint in a digital computational implementation, the diagonal augmentation terms were dropped from *both* the $W(t)$ and $dW(t)/dt$ matrix coefficients in the next version. The idea was to maintain exact cancellation of the eigenvalue dependence of convergence times while

allowing the algorithm to converge to the correct (Wiener) solution even with shorter averaging times. In most respects it did this, although the condition (75) was still necessary. Its weights (state variables) also tended to acquire damped oscillations and to wander in ill-conditioned scenarios, although the output residue was not appreciably affected. Stability analysis showed that the gain "g" had to be sufficiently large so that $g \lambda_{\min} > 4T_1/T_2$ to prevent oscillations and to assure a fixed dominant time-constant for convergence speed.

Consequently, an augmentation term of $\gamma = 1/\tau_2 \mathbf{I}$ was added to the matrix coefficient of dW/dt to overwhelm the minimum-eigenvalue term, thus creating the basic version of the FRQN algorithm, equation (63). If transformed back into continuous-time and its parameters suitably adjusted, it corresponds closely to Compton's algorithm with a single-pole averaging filter. It converges to the correct solution without wandering or oscillating, and allows for reduction in averaging times without completely degenerating to the NLMS algorithm. It benefits from the "persistent excitation" property because of the absence of only one leakage term. That term makes the integrator which outputs $W(t)$ "leaky", which effectively depletes its memory about the dimensions of the sampled signal-vectors, so it needs longer averaging times as in equation (69).

One can generalize the basic version of the FRQN algorithm by using various s-to-z transformations to create corresponding discrete-time versions of Compton's algorithm, and by adjusting their parameters (gains and time-constants). Similarly, one can also do this to the LMS algorithm, to e. g. minimize its misadjustment while not reducing drastically its convergence speed.

By following the procedure used to transform the differential equation for the continuous-time LMS algorithm (46) into discrete-time (55), (56), the differential equation for the original algorithm (66) is similarly transformed into discrete-time. (Alternative s-to-z transformations are also possible, giving rise to other version of the FRQN algorithm, as shown in Appendix C.) Equation (63) is obtained when only the derivative-weights' coefficient augmentation term is retained as described above, corresponding to the continuous-time version

$$T_1 T_2 \frac{d^2}{dt^2} W(t) + T_2 [\gamma I + g Y(t) Y^H(t)] \frac{d}{dt} W(t) + g Y(t) Y^H(t) W(t) = g Y(t) r^*(t)$$

The procedure to obtain equation (64) involves a straight-forward but tedious application of the matrix-inversion lemma after all the W_k terms have been isolated on the left-hand side in equation (63):

$$\begin{aligned} & \left[(\tau_1 \tau_2 + \tau_2 \gamma) I + g(1 + \tau_2) Y_k Y_k^H \right] W_k + \left[-2\tau_2(\tau_1 + \gamma/2) I - g\tau_2 Y_k Y_k^H \right] W_{k-1} + \\ & + \tau_1 \tau_2 W_{k-2} = g Y_k r_k^* \end{aligned} \quad (76)$$

Let $a_0 = \tau_2(\tau_1 + \gamma)$, $b_0 = g(1 + \tau_2)$, $a_1 = 2\tau_2(\tau_1 + \gamma/2)$, $b_1 = g\tau_2$, and write

$$\left[a_0 I + b_0 Y_k Y_k^H \right] W_k = \left[a_1 I + b_1 Y_k Y_k^H \right] W_{k-1} - \tau_1 \tau_2 W_{k-2} + g Y_k r_k^* \quad (77)$$

Using the matrix-inversion lemma to isolate W_k on the left side, the above equation becomes

$$\begin{aligned}
W_k = & \left(\frac{1}{a_0} \right) \left[I - \frac{Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] \left[a_1 I + b_1 Y_k Y_k^H \right] W_{k-1} - \\
& - \left(\frac{\tau_1 \tau_2}{a_0} \right) \left[I - \frac{Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] W_{k-2} + \frac{g}{a_0} \left[I - \frac{Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] Y_k r_k^*
\end{aligned} \quad (78)$$

Performing the matrix multiplication which precedes W_{k-1} and in the last term leads to

$$\begin{aligned}
W_k = & \left(\frac{1}{a_0} \right) \left[a_1 I + b_1 Y_k Y_k^H - \frac{a_1 Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} - \frac{b_1 Y_k (Y_k^H Y_k) Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] W_{k-1} - \\
& - \left(\frac{\tau_1 \tau_2}{a_0} \right) \left[I - \frac{Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] W_{k-2} + \frac{g}{b_0} \left[\frac{Y_k r_k^*}{a_0/b_0 + Y_k^H Y_k} \right].
\end{aligned} \quad (79)$$

By regrouping inner-product terms and simplifying, the final weight-updating equation is obtained as

$$\boxed{
\begin{aligned}
W_k = & \left[\frac{a_1}{a_0} I + \left(\frac{b_1/b_0 - a_1/a_0}{a_0/b_0 + Y_k^H Y_k} \right) Y_k Y_k^H \right] W_{k-1} - \\
& - \left(\frac{\tau_1 \tau_2}{a_0} \right) \left[I - \frac{Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] W_{k-2} + \\
& + \left(\frac{g}{b_0} \right) \frac{Y_k r_k^*}{a_0/b_0 + Y_k^H Y_k}
\end{aligned}
} \quad (80)$$

with $a_0 = \tau_2(\tau_1 + \gamma)$, $b_0 = g + \tau_2 g$, $a_1 = 2\tau_2(\tau_1 + \gamma/2)$ and $b_1 = g\tau_2 = b_0 - g$. The parameters were chosen as $\gamma = 1/\tau_2$, $g = 1/\lambda_{\min}$, $\tau_2 = 4N$ and $\tau_1 = \tau_2/4$, and, as with NLMS and RLS, the available output is given by

$$y_k = W_{k-1}^H Y_k.$$

3.2 Relation to RLS and NLMS Algorithms - a Geometric Interpretation:

Convergence rate of a true RLS algorithm is determined by the dimension, N , of the input signal vector, rather than by time-constants. It converges typically in $2N$ to $3N$ iterations. Time-constants determine the tracking properties of RLS algorithms [30]. These roles may be understood by recognizing that a true RLS algorithm solves the least-squares problem *at each iteration*, using each sample of the signal vector [31]. After $2N$ or so samples have been incorporated, the effect of optimizing with each new sample is that the (least-squares) solution is optimized in all $2N$ (complex) dimensions and the algorithm converges. The only condition required for this is that the sample signal vectors be linearly independent. It is analogous to the persistency of excitation condition for complete system identification, in the sense that the sample vectors must contain representations from each of the N dimensions or degrees of freedom. The time-constant of an RLS algorithm determines how fast these representations decay with time and are replaced by new ones, from a changed signal-space, as the (non-stationary) signal-statistics vary slowly over time.

However, the convergence as well as the tracking speed of the FRQN algorithm are determined by time-constants. This aspect is similar to the corresponding properties of the various versions of the LMS algorithm. It relies on the intrinsic averaging of the adaptation algorithm to produce reliable estimates of the requisite signal statistics, as explained by Compton in [4]. This averaging time is determined by open-loop hardware (or in this case software) time-constants which must be set with consideration of the signal bandwidth as well as dimension. They are also factors in the (closed-loop) convergence time-constants.

It is instructive to formulate a geometric interpretation of the FRQN algorithm in order to see the mechanism by which it essentially makes the NLMS algorithm approximate the Quasi-Newton convergence behaviour. For this treatment the FRQN algorithm is written in the form of equation (63) with g having units of inverse power for consistency with later developments:

$$\begin{aligned} \tau_1 \tau_2 \Delta^2 \mathbf{W}_k + \tau_2 \left[\frac{1}{\tau_2} \mathbf{I} + g \mathbf{Y}_k \mathbf{Y}_k^H \right] \Delta \mathbf{W}_k &= -g \mathbf{Y}_k \mathbf{Y}_k^H \mathbf{W}_k + g \mathbf{Y}_k r_k^* \\ &= -g \nabla_{(\mathbf{W}_k)} |e_k|^2 \end{aligned} \quad (81)$$

where $\Delta \mathbf{W}_k = \mathbf{W}_k - \mathbf{W}_{k-1}$ and $\Delta^2 \mathbf{W}_k = \Delta(\Delta \mathbf{W}_k) = \Delta \mathbf{W}_k - \Delta \mathbf{W}_{k-1} = \mathbf{W}_k - 2\mathbf{W}_{k-1} + \mathbf{W}_{k-2}$ are finite differences in the weight vector computed at the k -th iteration. This may be rewritten in the form of a stable difference equation for the quantity $\tau_2 \Delta \mathbf{W}_k$ as

$$\tau_1 \Delta(\tau_2 \Delta \mathbf{W}_k) = - \left[\frac{1}{\tau_2} \mathbf{I} + g \mathbf{Y}_k \mathbf{Y}_k^H \right] (\tau_2 \Delta \mathbf{W}_k) - g \nabla_{(\mathbf{W}_k)} |e_k|^2 \quad (82)$$

which looks like the (leaky, normalized) LMS algorithm with $\mathbf{W}_k$ replaced by $\tau_2 \Delta \mathbf{W}_k$ and the forcing function by the gradient. Its open-loop time-constant is τ_1 whereas that for the weight-updating is τ_2 , as will be shown later in a first-order approximation. It is in fact a stable algorithm which adapts the *actual step*, $\Delta \mathbf{W}_k$, towards the optimal direction proportionally to the amount by which the quasi-Newton step differs from the gradient step, on average. This is unlike the idea typified by reference [29] for example, where a separate algorithm running in parallel is used to adapt only the step-size *parameter*.

The way in which the weight-steps in the FRQN algorithm are adapted may be more clearly seen by averaging (81) and rewriting it as

$$\Delta (\tau_2 \Delta W_k) = -\mu R \left[(\tau_2 \Delta W_k) - (-R^{-1} \nabla_w \overline{e_k^2}) \right] \quad (83)$$

where the augmentation of matrix R was neglected and $\mu = g/\tau_1$ was used. Equation (83) shows that the weight-steps' adjustments (left-hand side) are proportional to their *vector* differences from the QN steps (square-bracketed term on the right-hand side). When these are viewed as the diagonalized modes of the weights and weight-steps (see e.g. appendix of [5]), then the proportionality factors for the modes' step changes are their respective eigenvalues. It might be suggested that this proportionality is not necessary, but it results from the necessity to adapt the weights themselves to the Wiener solution.

Since the second-order difference implies (vector) acceleration (such as is also experienced when descending the gradient), the effect of the algorithm is to force the weights to a straight-line motion toward the optimum, thereby effecting a QN type of behaviour. This acceleration-minimization also has the effect of smoothing the weight fluctuations relative to those of the LMS algorithm, as is apparent in some of the preliminary results in the next section. A possible reason why (82) can be made fast without causing excess jitter in the algorithm may be that the forcing function in (82) is slower and does not contain signal modulations which might interact with the weights. It may also be interesting to see if it is worth extending this smoothing idea to higher-order derivatives, but one must be careful not to cause instabilities.

By the same rationale as is used for explaining the convergence in the mean of the LMS algorithm, one can conclude that, on average in the steady state of (82),

$$\left[\frac{1}{\tau_2} I + g \overline{Y_k Y_k^H} \right] (\tau_2 \Delta W_k) = -g \nabla_{(W_k)} \overline{|e_k|^2} = -g R_k W_k + g S_k \quad (84)$$

where $R_k = \mathcal{E}\{Y_k Y_k^H\}$ and $S_k = \mathcal{E}\{Y_k r_k^*\}$, and independence of W_k from Y_k and d_k has been assumed. Now (84) looks somewhat like the RLS algorithm written in its QN form

$$R_k \Delta W_k = -Y_k Y_k^H W_{k-1} + Y_k r_k^* \quad (85)$$

except that RLS does not possess the time-constant and gain parameters corresponding to τ_2 and g . (The equivalent of a time-constant in the RLS algorithm is $1/(1-\lambda)$, solely for tracking purposes [30].) Also, notice that (85) contains W_{k-1} on its right-hand side whereas (84) uses W_k . An attempt to change this to W_{k-1} led to divergence in the FRQN algorithm, which was confirmed by a stability analysis of the averaged version of equation (81).

Although the right-hand sides of (84) and (85) are not equal because of the different convergence rates and also because the W_k 's are not the same quantity in the two equations, it is apparent that, because the closed-loop time-constant of (82) is much smaller than the adaptation time-constant τ_2 for W , the FRQN algorithm starts following the quasi-Newton convergence path after a short time from the start of adaptation. This is the same path followed by the RLS algorithm, although at a different rate. Because the FRQN algorithm explicitly iterates toward the optimal solution (or its best current estimate) with predetermined time-constants, it will inherently be slower than RLS, which does not iterate in the same sense. Rather than incrementally approach the optimal solution, the RLS algorithm *computes* its best estimate at each of its "iterations" with each new data sample. It therefore obtains that solution when a representative amount of data has been sampled, which depends upon its rank and dimensionality (e.g.

2N samples for data represented by complex N-vectors), rather than on averaging times as for NLMS and FRQN.

The FRQN algorithm may be said to belong to a category of adaptive algorithms characterized by order-of-N computational complexity, second- or higher-order recursion of the state variables (adapting filter coefficients), arbitrary spatial sensor-array geometry, and stable quasi-Newton convergence behaviour. According to the philosophy of reference [28], the RLS algorithm is also a quasi-Newton algorithm.

3.3 Preliminary Simulation Results:

Figures 3 and 4 below represent the reactions of a 4-beam generalized adaptive sidelobe canceller to a signal scenario for which the array covariance matrix R is well-conditioned, its eigenvalue spread being about 10. Three interferers and uncorrelated thermal noise were always present in each simulated scenario. Angular positions of the interference sources are indicated on the antenna pattern plots by " $\dagger$ " at their *relative* transmit power levels. A desired signal was not included in these scenarios so as not to mask the residual output interference power; it would be situated at 0 degrees. (Since all the adaptively-weighted beams have a spatial null at 0 degrees, the desired signal would not affect the adaptation). The next section, as well as Appendix F contain a more quantitative description of the simulated scenarios and adaptive antennas.

Parts (a) of each figure show the evolution of the weights' control-voltages on linear time-scales; parts (b) show the evolutions of the corresponding residual output powers (without a desired signal) on logarithmic scales, and parts (c) show the initial and adapted (after 1000 iterations) antenna directivity patterns.

Figure 3(a): Evolution of FRQN weights in well-conditioned scenario.

Figure 3(b): Evolution of FRQN residual output power in well-conditioned scenario.

Figure 3(c): Initial and adapted antenna directivity patterns for FRQN algorithm in well-conditioned scenario.

Figure 4(a): Evolution of NLMS weights in well-conditioned scenario.

Figure 4(b): Evolution of NLMS residual output power in well-conditioned scenario.

Figure 4(c): Initial and adapted directivity patterns for NLMS algorithm in well-conditioned scenario.

In the basic FRQN algorithm the open-loop gain was $g=67$, and the time-constants were $\tau_1 = 1$ and $\tau_2 = 10$. For comparison, the (normalized) LMS algorithm was empirically set up to have its fastest modes converge in about the same time as the basic FRQN algorithm took to converge, so they would both have the same steady-state misadjustment (at least according to [17]). Its open-loop gain was also $g=67$, but its open-loop time-constant was set to $\tau = 7296$.

The same algorithms, with the same gains and time-constants, were next compared in a scenario for which R was highly ill-conditioned, having an eigenvalue spread of about 10 000. The results appear in the same format in Figures 5 and 6 below. Note that the FRQN algorithm has some slower modes now, but not even an order of magnitude slower. It converges in about 200 iterations compared to 75 in the previous scenario. It clearly does not degenerate to the NLMS algorithm even with $\tau_1 = 1$ rather than $4N$. By contrast, the NLMS algorithm, with the same parameters as in Figures 3 and 4 (except now $\tau = 4032$), now shows a high disparity of modal convergence rates. It would converge even more slowly if its fastest mode was again made as fast as in the previous scenario. (In this scenario the fastest NLMS mode expires in about 10 iterations.) In all Figures the weights start at zero and the same pseudorandom signal sequences are used.

Figure 5(a): Evolution of FRQN weights in ill-conditioned scenario.

Figure 5(b): Evolution of FRQN residual output power in ill-conditioned scenario.

Figure 5(c): Initial and adapted antenna directivity patterns for FRQN algorithm in ill-conditioned scenario.

Figure 6(a): Evolution of NLMS weights in ill-conditioned scenario.

Figure 6(b): Evolution of NLMS residual output power in ill-conditioned scenario.

Figure 6(c): Initial and adapted antenna directivity patterns for NLMS algorithm in ill-conditioned scenario.

When the FRQN and NLMS algorithms were run in the same scenarios without receiver noise, it was found that they performed similarly in the well-conditioned scenario (the FRQN algorithm reached a residue near machine epsilon or, about 2×10^{-16} , in 200 iterations and NLMS in 400 iterations). NLMS was meant to converge about as fast as the FRQN algorithm in this scenario so its time-constant was adjusted to 7296. However, in the ill-conditioned scenario, which was designed to give a very large eigenvalue spread, the FRQN algorithm reached a steady-state residue of 3×10^{-5} in about 400 iterations. The NLMS algorithm had a time-constant of 4032, yet it took 4000 iterations to reach the same residue. In 10 000 iterations it reached a steady-state residue of about 8×10^{-6} . This quick demonstration, though not optimally "tidy", serves to illustrate the advantage of the FRQN algorithm relative to the NLMS algorithm in terms of speed, while not incurring an appreciable penalty in steady-state mean square error. The loop gains, g , of both algorithms were 67 in both scenarios.

The FRQN algorithm has been analyzed for stability and effects of disturbances (e.g. voltage biases). It was found to be stable and robust. (Extensive analyses were performed on the original continuous-time version and verified by analogue hardware implementation in connection with [5].) Further stability and robustness analyses are performed in the appendices and discussed in later sections. The basic FRQN algorithm is numerically stable; it ran for $> 2 \times 10^5$ iterations overnight on an IBM-compatible, '386/33MHz personal computer, using the AT-MATLAB 3.5k software for modelling. No signs of instability or error-accumulation were noticed; the residual error remained constant in stationary scenarios.

Both algorithms were next tested in "optimal", fully-adaptive arrays using reference signals. State-variable and output error evolutions were compared quantitatively. These results are discussed in the next section.

3.4 Quantitative Simulation Comparisons of Basic FRQN and NLMS Algorithms:

The FRQN and the NLMS algorithms were simulated in a more realistic, mobile-radio environment. A perfect replica of the desired signal was assumed to be available to use as the reference signal. The algorithms were configured to minimize the mean square error using 4 directive elements spaced half-wavelength apart along a line. One source of desired and three independent sources of interference signals within 3 dB of the desired-signal level were modelled as 25 kHz band-limited noise at 850 MHz carrier frequency. All sources were capable of motion, but fading and shadowing effects were not modelled. Only 2-ray elevation multipath propagation was modelled, with slightly randomized specular reflections from the ground for each signal, but no delay spread. The signals were modelled as independent complex Gaussian pseudo-random sequences bandlimited by phaseless filters to about 25 kHz, sampled at 60 kHz. They represented the received signals at baseband. Independent random noise corresponding to a noise-figure of 10 dB was injected into each of the 4 receiving channels, resulting in SNR of about 63 dB for the desired signal. (Note that "noise" does not include "interference".)

The adaptive-array antenna was assumed to be mounted on a mobile-radio base-station mast 50 m above a near-perfectly conducting infinite plane earth, and consisted of $N=4$ elements spaced at 18 cm intervals along the 0° radial. Each element was directive in the 90° radial direction and connected to one coherent receiver and complex weight. An attempt was made to normalize the residual error to unity (0 dB) when the weights were in their initial zero states.

Similar scenarios as in the preceding section with the sidelobe-canceller configuration, were tested with these error-minimizing configurations of the algorithms. Antenna patterns were not plotted this time, since in general the sources could be at different elevation angles, requiring a laborious 3-dimensional illustration of the spatial nulls formed on them. The relative performances of the two algorithms were similar as before. When the simulations were repeated without noise and motion, no noticeable differences were observed upon introduction of source motions.

Briefly, the two scenarios consisted of 4 independent mobile signal sources 2 m above ground and 3 km radially distant from the basestation. In the "orthogonal" (well-conditioned) scenario the sources' bearings relative to the line of the array were 90° , 140° , 52° , 32° . Their corresponding transmitter powers were 47, 44, 43 and 50 dBm and velocities were $60 \text{ km/hr @ } 180^\circ$, $30 \text{ km/hr @ } 180^\circ$, $130 \text{ km/hr @ } 0^\circ$, 0 km/hr , respectively. In the "parallel" (ill-conditioned) scenario, only the interferers' bearings were changed, to 72° , 71° , 70° , in the same order. The desired signal remained at 90° , directly in front of the array.

Convergence performance and relevant algorithm parameters are more tidily summarized in Tables 2(a), 2(b) and 3 and the Figures which follow. The last table shows a variation of the FRQN algorithm in which the time-constants were reduced in an attempt to speed it up. In fact, the resulting effect was more the opposite, due to reduction of the averaging time and consequently more NLMS-like convergence behaviour. This illustrates that a trade-off exists between averaging times and convergence times of the FRQN algorithm.

To help interpret the following results, the theoretical values of the NLMS closed-loop time-constants were calculated according to Appendix B as

$$\tau_n = \tau / (g \lambda_n) \quad (86)$$

and theoretical convergence times were given approximately by

$$k \approx 5 \tau_{n \max} \quad (87)$$

In the orthogonal (relatively well-conditioned) scenario the NLMS weights converge in the theoretically expected 250 or so iterations (the squared error in $k=500$). The FRQN algorithm takes longer than the expected $5 \tau_2$ iterations because there is some loss of averaging time in selecting τ_1 and τ_2 smaller than advised by equation (75). Increasing these time-constants eventually causes it to converge in the theoretical number of iterations; unfortunately that is more than NLMS takes in this scenario. The current values of these time-constants represent an empirical near-optimal compromise between faster, more LMS-like (variable) and slower but more constant convergence rates for the FRQN algorithm.

In the ill-conditioned scenario, this selection of time-constants is seen to result in only a moderate increase in convergence time compared to that of the NLMS algorithm. The NLMS algorithm converges practically in 5 times the second-slowest time-constant, as the slowest time-constant governs a mode responsible for only a negligible portion of the output error power.

Table 2(a): Well-Conditioned Scenario, $\lambda_{\max}/\lambda_{\min} = 28.05$, $N = 4$.					
Algorithm	Iterations to converge, k	Open-loop gain, g	Open-loop time-constants	Closed-loop time-constants	Figures: weights (a) error (b)
FRQN	300	5.8×10^{10}	$\tau_1 = 4$	$\tau_2 = 16$	7(a)
					7(b)
NLMS	500	1.71×10^{10}	$\tau = 16$	$\tau_a = 1.87$	8(a)
				$\tau_b = 2.99$	
				$\tau_c = 53.7$	8(b)
				$\tau_d = 8.96$	

Table 2(b): Ill-Conditioned Scenario, $\lambda_{\max}/\lambda_{\min} = 6.54 \times 10^6$, $N = 4$.					
Algorithm	Iterations to converge, k	Open-loop gain, g	Open-loop time-constants	Closed-loop time-constants	Figures: weights (a) error (b)
FRQN	1000	4.9×10^{15}	$\tau_1 = 4$	$\tau_2 = 16$	9(a)
					9(b)
NLMS	15000	9.33×10^9	$\tau = 16$	$\tau_a = 1.26$	10(a)
				$\tau_b = 4.86$	
				2.8×10^3	10(b)
				8.3×10^6

Figure 7(a): Weights of FRQN algorithm in well-conditioned scenario.

Figure 7(b): Squared output error of FRQN algorithm in well-conditioned scenario

Figure 8 (a) Weights of NLMS algorithm in well-conditioned scenario.

Figure 8 (b) Squared output error of NLMS algorithm in well-conditioned scenario

Figure 9 (a) Weights of FRQN algorithm in ill-conditioned scenario

Figure 9 (b) Squared output error of FRQN algorithm in ill-conditioned scenario

Figure 10 (a) Weights of NLMS algorithm in ill-conditioned scenario

Figure 10 (b) Squared output error of NLMS algorithm in ill-conditioned scenario

The Table below and subsequent Figures illustrate that simple reduction of time-constants must be traded off against degeneration to more NLMS-like convergence behaviour. While it results in faster convergence in well-conditioned scenarios, convergence in ill-conditioned scenarios is slower, as is the case for NLMS. The same adaptive array system and scenarios were used as in Table 2(a) and Table 2(b), but the time-constants of the FRQN algorithm were reduced to $\tau_2 = 3N = 12$, and $\tau_1 = \tau_2/5 = 2.4$.

Table 3: Basic FRQN Algorithm With Reduced Time-Constants.				
Scenario	Iterations to Converge, k	τ_1	τ_2	Figures: Weights (a) Error (b)
Well-Conditioned	200	2.4	12	11 (a)
				11 (b)
Ill-Conditioned	2000	2.4	12	12 (a)
				12 (b)

Figure 11 (a): Weights of FRQN of Table 3 in well-conditioned scenario.

Figure 11 (b): Squared output error of FRQN algorithm of Table 3 in well-conditioned scenario.

Figure 12 (a): Weights of FRQN of Table 3 in ill-conditioned scenario.

Figure 12 (b): Squared output error of FRQN algorithm of Table 3 in ill-conditioned scenario.

3.5 Refinements of the Algorithms:

Some refinements of the NLMS and FRQN algorithms become useful in their discrete-time forms. Specifically, it is legitimate to allow $g \rightarrow \infty$ in both algorithms without fear of instability, unlike in their continuous-time counterparts. Furthermore, because the available outputs always use the weights of the previous iteration, there is no danger of producing a useless output signal stripped of information.

The limiting form of the NLMS algorithm as $g \rightarrow \infty$ is obtained by letting $\mu \rightarrow \infty$ in equation (56) and substituting it into equation (55). This limiting form has eliminated the open-loop gain and time-constant as parameters from the NLMS algorithm, thus simplifying it as well as speeding it up to its maximum speed [35]. Slock has a different explanation for "instantaneous" adaptation based on a projection interpretation of NLMS as $g \rightarrow \infty$ in [35]. The refined form of the NLMS algorithm is therefore given by equation (55)

$$W_k = W_{k-1} - \beta_k Y_k Y_k^H W_{k-1} + \beta_k Y_k r_k^* \quad (88)$$

with $\beta_k = 1/(Y_k^H Y_k)$, driven effectively by α_k according to

$$\Delta W_k = -\beta_k Y_k \alpha_k^* \quad (89)$$

and its available output is still as in (57) $y_k = W_{k-1}^H Y_k$. Although now the closed-loop time-constants cannot be determined by analogy with the continuous-time form as in equation (86) with finite g , the discussion following equation (56) to (62) and in Appendix B serve to show that convergence times will be halved by this refinement, as was indeed observed in simulations.

A corresponding form of the FRQN algorithm can be obtained by a similar approach, i.e. letting $g \rightarrow \infty$ in its recursive update equation. It has the effect of setting $a_0/b_0 = 0$ in equation (80). When the simulations in Figures 7 and 9 were repeated with this algorithm, no changes in performance were observed. It was also verified to be numerically stable in long runs, as well as in theory (in the averaged sense), for stationary scenarios.

Another refinement of the FRQN algorithm was motivated by resorting to its continuous-time original form. Looking at equation (69), it may occur to one that if g on the LHS was reduced, then the closed-loop time-constant would also be reduced by the same amount, say q . Carrying this idea into the discrete-time form has the effect of changing the parameters (in equation (76) for example) to

$$b_0 = g(1 + \tau_2/q), \quad b_1 = g\tau_2/q. \quad (90)$$

It was found by simulation that if the equivalent of the entire LHS was reduced by q , excessive ringing of the coefficients would occur. The value of $q=2$ was found to be nearly optimal; higher values tended to increase ringing and convergence times, since the diagonal term on the LHS then became effectively larger, resulting in loss of QN behaviour. (As $q \rightarrow \infty$ with $\gamma = 1$, the FRQN algorithm becomes an "ordinary" second-order LMS algorithm which behaves essentially like the LMS algorithm except for possible ringing oscillations of various frequencies in the coefficients. A version of it is analyzed in [26].) Although the geometric interpretation with $q > 1$ now suggested that the weight steps would converge toward the Newtonian direction more slowly, since equation (83) would become

$$\Delta(\tau_2 \Delta W_k) \approx -\frac{\mu}{q} R[(\tau_2 \Delta W_k) - q(-R^{-1} \nabla_w |e_k|^2)] \quad (91)$$

the steps in that direction would be q times larger. Besides, the infinite g would ensure that division of the RHS by q would not pose a serious threat to the time-constant of adaptation of the weight *steps*. The complete refined FRQN algorithm is therefore given by

$$\begin{aligned}
 W_k = & \left[\frac{a_1}{a_0} I + \left(\frac{\frac{\tau_2}{q + \tau_2} - \frac{a_1}{a_0}}{Y_k^H Y_k} \right) Y_k Y_k^H \right] W_{k-1} - \\
 & - \left(\frac{\tau_1 \tau_2}{a_0} \right) \left[I - \frac{Y_k Y_k^H}{Y_k^H Y_k} \right] W_{k-2} + \\
 & + \left(\frac{q}{q + \tau_2} \right) \frac{Y_k r_k^*}{Y_k^H Y_k}
 \end{aligned} \tag{92}$$

with $a_0 = \tau_2(\tau_1 + \gamma)$, $a_1 = 2\tau_2(\tau_1 + \gamma/2)$, $\gamma = 1/\tau_2$, $q = 2$, $\tau_2 = 4N$, $\tau_1 = \tau_2/4$, and output by

$$y_k = W_{k-1}^H Y_k. \tag{93}$$

It ran with the predicted two-fold increase in speed for $> 10^5$ iterations without signs of divergence. The following simulations illustrate its superior behaviour with respect to that of the refined NLMS algorithm. Theoretical mean squared error (mse) and minimum mean squared error (mmse) are plotted for both algorithms in stationary scenarios. They are derived in detail in Appendix D. Results of simulating the same scenarios with the RLS algorithm are also included. It was refined only in the sense of choosing version II RLS from [31, pp. 484-485] with the time-constant of FRQN used to determine the forgetting factor as $\lambda = 1 - 1/\tau_2$, since both parameters affect the tracking behaviour. It was initialized with $R_0^{-1} = I/(0.001 \mathcal{E}\{Y_k^H Y_k\}/N)$, in keeping with [31].

The scenarios in Figures 13(a) and 13(b) to 18(a) and 18(b) are the same ones as in Figure 7 - 10 but the signal sources are *totally* stationary. (These scenarios were set up to reveal algorithm behaviours near both extremes of scenario conditions and have been useful in early detection of instabilities and excessive residual errors in algorithm variants. Simulation of fast fading is deferred to later sections where the NLMS, FRQN and RLS algorithms will be compared in adapting with binary desired signals whose bit-error-rates (BER) will be computed.) As mentioned earlier, the parameters in Tables 2(a) and 2(b) remain the same in the following simulations except for g being infinite, the closed-loop NLMS time-constants being effectively halved, and $q=2$ halving those of FRQN as well.

The bumps in the approximate theoretical mse curves for the FRQN algorithm are believed to be harmless artefacts caused by the approximation $\mathcal{E}\{W_{k-1} W_{k-2}^H\} \approx \mathcal{E}\{W_{k-1} W_{k-1}^H + W_{k-2} W_{k-2}^H\}/2$ which was used in deriving them. They disappear as $k \rightarrow \infty$.

Figure 13(a): Weights of FRQN algorithm in well-conditioned scenario.

Figure 13(b): Squared errors of FRQN in well-conditioned scenario.

Figure 14(a): Weights of NLMS algorithm in well-conditioned scenario.

Figure 14(b): Squared errors of NLMS in well-conditioned scenario.

Figure 15(a): Weights of RLS algorithm in well-conditioned scenario.

Figure 15(b): Squared errors of RLS in well-conditioned scenario.

Figure 16(a): Weights of FRQN algorithm in ill-conditioned scenario.

Figure 16(b): Squared errors of FRQN in ill-conditioned scenario.

Figure 17(a): Weights of NLMS algorithm in ill-conditioned scenario.

Figure 17(b): Squared errors of NLMS in ill-conditioned scenario.

Figure 18(a): Weights of RLS algorithm in ill-conditioned scenario.

Figure 18(b): Squared errors of RLS in ill-conditioned scenario.

Notice that the RLS algorithm converged in approximately N to $2N$ iterations in both scenarios, but recall that it requires in the order of N^2 operations per iteration, N being 4 here. It also appears to have suffered from some numerical wandering of coefficients after adaptation in the ill-conditioned scenario, perhaps due to lack of persistent excitation, in spite of the care taken to assure reasonable numerical stability as recommended in [31]. (RLS version I diverged when it was tried in this application.) The wandering of RLS coefficients in the unexcited subspace did not, however, seem to affect its output error.

To recapitulate, the FRQN algorithm has been developed to the stage where it converges 10 - 15 times faster than the NLMS algorithm operating at its highest speed (i.e. according to [35]). Both the FRQN and NLMS algorithms were further divorced from their origins as continuous-time analogue servo loops by letting g become infinite in their recursive equations, which simplified them. This refinement should also vaporize any concerns about remaining disparities among the smallest eigenvalues in Compton's algorithm which FRQN resembles [5].

Time-constants have been determined by input-signal bandwidth constraints and reduced empirically to the point of diminishing returns, where further increases in speed are overtaken by degeneration to more LMS-type (slower, variable) behaviour. These are given by $\tau_2 = 4N$, $\tau_1 = \tau_2/4$ in the present version of the FRQN algorithm.

A reduction of the closed-loop time-constant to τ_2/q with $q=2$ was achieved. Further increases in q were ineffective as they tended to make the diagonal covariance-augmentation term γ appear larger, thus departing from QN behaviour. Its value in turn was stabilized at $\gamma = 1/\tau_2$ empirically; a non-zero value is required to avoid unconstrained wandering of degrees of freedom.

Next, a more general s-to-z transformation was developed in the hope of spawning more variants of the FRQN algorithm, but was found to offer no practical advantages to them relative to the backward-difference transformation. Nevertheless, it provides a continuum encompassing the bilinear and backward-difference transformations by variation of a single parameter. Two constraints which moulded it are the requirement that it should map the left half of the s-plane to the inside of a circle within the unit circle in the z-plane to preserve stability and the requirement that it should map the point $s = \sigma + j\omega = 0 + j0$ to the point $z = 1 + j0$ to preserve the correct steady state. It is developed in detail in Appendix A and is given by

$$s = \frac{2\rho}{\Delta t} \left(\frac{z-1}{z-1+2\rho} \right) \quad (94)$$

with the radius of the stability circle corresponding to $\text{Re}\{s\} < 0$ variable as $0 < \rho < 1$. The bilinear transformation is obtained with $\rho = 1$ and the backward-difference transformation with $\rho = 1/2$. The generalized, refined FRQN algorithm resulting from this transformation with finite g to begin with, is developed in detail in Appendix C, and is given by

$$\begin{aligned} W_k = & \left[\left(\frac{a_1}{a_0} \right) I - \left(\frac{a_1/a_0 - b_1/b_0}{a_0/b_0 + Y_k^H Y_k} \right) Y_k Y_k^H \right] W_{k-1} - \\ & - \left[\left(\frac{a_2}{a_0} \right) I - \left(\frac{a_2/a_0 - b_2/b_0}{a_0/b_0 + Y_k^H Y_k} \right) Y_k Y_k^H \right] W_{k-2} + \\ & + \left(\frac{4\rho^2 g}{b_0} \right) \left(\frac{1}{a_0/b_0 + Y_k^H Y_k} \right) Y_k r_k^*, \end{aligned} \quad (95)$$

with $y_k = W_{k-1}^H Y_k$, $\alpha_k = y_k - r_k$, and the parameters

$$\begin{aligned}
a_0 &= 2\rho\tau_2(2\rho\tau_1 + \gamma), & b_0 &= g + 2\rho\tau_2g_d, \\
a_1 &= 4\rho\tau_2(2\rho\tau_1 + \gamma(1 - \rho)), & b_1 &= 2g(1 - 2\rho) + 4\rho\tau_2g_d(1 - \rho), \\
a_2 &= 2\rho\tau_2(2\rho\tau_1 + \gamma(1 - 2\rho)), & b_2 &= g(1 - 2\rho)^2 + 2\rho\tau_2g_d(1 - 2\rho).
\end{aligned}$$

where $g_d = g/q$. With $\rho = 1$ and g equal to reciprocal noise power, it works equally well as with $\rho = 1/2$. However, when $g \rightarrow \infty$ it diverges in the orthogonal scenario with the original time-constants. Reducing ρ to somewhere below $1/2$ predictably slows it down relative to the first (backward-difference) FRQN version, because the root locus (pole trajectories as functions of eigenvalues) becomes restricted to smaller circles near $z = 1$, hence lower closed-loop pole frequencies. Values of $\rho > 1/2$ tend to allow the root locus to approach too close to the unit circle as eigenvalues increase, especially with infinite g . Recognizing that this is a *stochastic* algorithm with *quantization* errors, it is best to keep as far away from the unit circle as possible under such conditions. That is precisely what is accomplished by the original s-to-z transformation with $\rho = 1/2$, because it drives the root locus to $z = 0$, which is as far away from the unit circle as is possible while remaining inside it.

Therefore the backward-difference version of the FRQN algorithm (92) is the most robust with respect to stochastic and quantization errors (at least in a stationary environment), and also does not impose filtering on the forcing term arising from the continuous-time formulation (which can be shown to have no effect since its bandwidth greatly exceeds that of any reconstruction filter). Besides that, it has a nice geometric interpretation. Generalization to higher than second orders was not attempted because of the greater danger of creating an unstable algorithm. A second-order feedback system is the highest order which is theoretically guaranteed to be asymptotically stable with any size of feedback gain.

3.6 Characterization and Performance Comparisons of Algorithms:

Stability and convergence in the mean were analyzed theoretically in Appendix B and verified by simulations for the NLMS and the FRQN algorithm. The necessary conditions and associated time-constants and steady states of the weights were derived in detail therein. It was shown that both algorithms are always stable in the mean in the forms used herein.

Both algorithms were also characterized in terms of convergence in the mean square in Appendix D. (To keep down the size of the thesis, RLS was not characterized in the mean square theoretically. Such a characterization of NLMS is more helpful in understanding the FRQN algorithm, as it optimizes the same kind of cost function, unlike RLS.) The most general version of the FRQN algorithm was analyzed to determine if an optimal set of parameters could be found which minimizes the mean squared error. Stationary and non-stationary signal environments were characterized separately, assuming the misadjustment and tracking errors to be independent. In general the total mean squared error (*mse*) of an adaptive filtering algorithm is given by the sum of three terms which are independent according to the assumptions and conditions stated in Appendices B and D, as

$$mse = mmse + emse + tmse \quad (96)$$

where *mmse* is the inevitable minimum mean squared error, *emse* is the excess mean squared error due to the stochastic adaptation process (coefficient jitter) and *tmse* is the tracking mean squared error due to the lagging of the coefficients with respect to parameters of a weakly non-stationary signal scenario. The *mmse* is a constant quantity. While the steady-state expressions for *emse* and *tmse* were of primary interest, the dynamic evolutions of *emse(k) + mmse* were derived and included, along with *mmse*, in the plots in Figures 13(b) to 16(b), which involve only stationary scenarios.

Therefore the "learning curve" of the NLMS algorithm is characterized in stationary scenarios by

$$mse_k = \mathcal{L}^H C_{k-1} + mmse \quad (97)$$

where $mmse = \sigma_r^2 - S^H W_{opt}$, $\sigma_r^2 = \mathcal{E}[r_k r_k^*]$. W_{opt} is given in Appendix B and so is matrix R , whose eigenvalues are the diagonal entries of matrix Λ and also the entries of the column vector $\mathcal{L}$. These appear in the recursion for the column vector C in (97) as

$$C_{k-1} = [I - 2\beta_k \Lambda + \beta_k^2 (\mathcal{L} \mathcal{L}^H + \Lambda^2)] C_{k-2} + \beta_k^2 \mathcal{L} mmse \quad (98)$$

where $\beta_k = \mu / (1 + \mu Y_k^H Y_k)$, $Y_k^H Y_k = Tr[R]$ and $\mu = g/\tau$ were used. It is initialized with $C_0 = [diagonal(E^H W_{opt} W_{opt}^H E)]$, where the columns of E are the eigenvectors of R corresponding to the eigenvalues $\{\lambda_n\}$ in Λ and $\mathcal{L}$. This characterization is the most general: the refinements are easily incorporated in the parameters.

In the steady state as $k \rightarrow \infty$ its excess mean squared error is given by

$$emse = \frac{\sum_{n=1}^N \frac{\beta \lambda_n}{2 - \beta \lambda_n}}{1 - \sum_{n=1}^N \frac{\beta \lambda_n}{2 - \beta \lambda_n}} mmse \quad (99)$$

where $\beta \lambda_n < 2$ is often assumed, and $\sum \lambda_n = Tr[R]$ is used in approximating $\beta_k \approx \beta = (g/\tau) / (1 - (g/\tau) Tr[R])$.

In non-stationary scenarios where σ_Φ^2 is the variance of the optimal weights and Φ is their parameter of non-stationarity according to the model

$$W_{ok} = \Phi W_{ok-1} + \varphi_k \quad (100)$$

the reference signal is given by the output of the model (optimal) system as

$$r_k = y_{ok} = W_{ok}^H Y_k - v_k \quad (101)$$

where v_k is additive random noise which gives a finite mmse. The corresponding steady-state tracking error of the NLMS algorithm is derived in Appendix D as

$$tmse = \frac{2\sigma_\varphi^2}{(1 + \Phi)\beta} \left[\frac{\sum_{n=1}^N \frac{1}{(2 - \beta\lambda_n)}}{1 - \sum_{n=1}^N \frac{\beta\lambda_n}{(2 - \beta\lambda_n)}} \right] \quad (102)$$

which is inversely proportional to β , while $emse$ is directly proportional to it. As $\beta \rightarrow \infty$, the misadjustment of the NLMS algorithm with $\beta\lambda_n < 2$ is $emse/mmse = 1$. Then the tracking error becomes

$$tmse = \frac{2\sigma_\varphi^2 N \sum_{n=1}^N \lambda_n}{(1 + \Phi)} \quad (103)$$

The "learning curve" of the FRQN algorithm in its general form (95) was characterized by a similar procedure in Appendix D as

$$mse_k = \varphi^H N_k + mmse \quad (104)$$

where the (now second-order) recursion for the column vector N_k is approximated by

$$\begin{aligned}
N_k = & \left(\frac{a_1}{a_0} \right)^2 \left[(I + z_k \Lambda)^2 + z_k^2 \mathcal{Q} \mathcal{Q}^H \right] N_{k-1} - \\
& - 2 \operatorname{Re} \left\{ \left(\frac{a_1 a_2}{a_0^2} \right) \left[(I + z_k \Lambda) (I + x_k \Lambda) + z_k x_k \mathcal{Q} \mathcal{Q}^H \right] N_{1,2(k-1)} \right\} + \\
& + \left(\frac{a_2}{a_0} \right)^2 \left[(I + x_k \Lambda)^2 + x_k^2 \mathcal{Q} \mathcal{Q}^H \right] N_{k-2} + c_k^2 \operatorname{mmse} \mathcal{Q}
\end{aligned} \quad (105)$$

where the vector of diagonal entries of the $E^H W_{k-1} W_{k-2}^H E$ cross-correlation matrix was approximated as

$$N_{1,2(k-1)} \approx \frac{1}{2} (N_{k-1} + N_{k-2})$$

and initialized the same way as C_0 . The parameters are those associated with equation (95) and

$$\begin{aligned}
z_k &= - \frac{1 - b_1 a_0 / (a_1 b_0)}{a_0 / b_0 + Y_k^H Y_k} \\
x_k &= - \frac{1 - b_2 a_0 / (a_2 b_0)}{a_0 / b_0 + Y_k^H Y_k} \\
c_k &= - \frac{4 \rho^2 g / b_0}{a_0 / b_0 + Y_k^H Y_k}
\end{aligned}$$

which were made independent of k by again using the approximation $Y_k^H Y_k \approx \operatorname{Tr}[R] = \sum \lambda_n$. The other quantities are the same as those for the NLMS learning curve in a stationary environment. In steady state the FRQN excess mean squared error was found to be given by

$$emse = \frac{\sum_{n=1}^N \frac{p\lambda_n}{2-p\lambda_n}}{1 - \sum_{n=1}^N \frac{p\lambda_n}{2-p\lambda_n}} mmse \quad (106)$$

with

$$p = \frac{4\rho^2 g}{2\rho\tau_2(2\rho\tau_1 - \gamma) - g(1 + 2\rho\tau_2/q)Y^H Y}$$

serving a role similar to that of β in the NLMS algorithm. It was assumed that in p. $Y^H Y = \text{Tr}[R]$. An optimal value of ρ which minimizes emse was sought within the limits $0 < \rho < 1$ in Appendix D, but was found not to exist. As $g \rightarrow \infty$, the misadjustment of the NLMS algorithm with $\beta\lambda_n < 2$ is $emse/mmse = 1$. Under similar conditions with $p\lambda_n < 2$ and $\rho = 1/2$, the FRQN algorithm *already* has a much smaller misadjustment than NLMS, given by

$$\frac{emse}{mmse} = \frac{1}{1 + 2\tau_2/q} < 1. \quad (107)$$

(The value of $\rho = 1$ creates an explicit potential condition for divergence and will not be used.) Figures 20 and 21 confirm that, for the same convergence speeds, the FRQN algorithm (92) has a much smaller misadjustment in steady state than the NLMS algorithm (88), as predicted by (107) above.

Convergence in the mean square in a non-stationary environment was also analyzed, specifically for tracking error with the performance-surface jittering in position but not in shape. Its random movement was again characterized by a first-order Markov process modelling the optimal coefficients, which jittered much more slowly than the adapting coefficients (quasi-stationary assumption). For the FRQN algorithm with

$\rho = 1/2$ the mean squared tracking error in the quasi-steady state was found in Appendix D to be approximately given by

$$\begin{aligned}
 mse = & \frac{2\sigma_{\Phi}^2(q + \tau_2) \left(\frac{\tau_1}{\tau_1 + \gamma} \right)^2 \sum_{n=1}^N \lambda_n \left\{ \frac{N}{2} \left[\left(1 - \frac{\gamma}{\tau_1} \right)^2 + \Phi \frac{\gamma}{\tau_1} + \Phi^2 \left(2 - \frac{\gamma}{\tau_1} \right) + \Phi^3 \right] - \right. \\
 & \frac{p^2}{2} \left[\left(1 - \frac{\gamma}{\tau_1} + \frac{\tau_2}{\tau_2 + q} \right)^2 - 2 \left(1 - \frac{\gamma}{\tau_1} \right)^2 - \Phi \frac{\gamma}{\tau_1} \left(\frac{q}{\tau_2 + q} \right) - \Phi^2 \left(2 - \frac{\gamma}{\tau_1} + \frac{\tau_2}{\tau_2 + q} \right) + \Phi^3 \right] - \\
 & \left. + \frac{Tr[R^2]}{(Tr[R])^2} \left[\left(1 - \frac{\gamma}{\tau_1} - \frac{\tau_2}{\tau_2 + q} \right)^2 + \Phi \frac{\gamma}{\tau_1} \left(1 - \frac{\tau_2 \tau_1 / \gamma}{\tau_2 + q} \right) + \Phi^2 \left(2 - \frac{\gamma}{\tau_1} + \frac{\tau_2}{\tau_2 + q} \right) - \Phi^3 \right] \right\} \quad (108)
 \end{aligned}$$

where σ_{Φ}^2 is the variance of the optimal weights and Φ is their parameter of non-stationarity. The dependence on τ_1, τ_2 is negligible and Φ is slightly less than 1 and $p \approx q / ((q + \tau_2) Tr[R])$

Since $\tau_2/q > 1$ the FRQN tracking error would be greater than that of NLMS. Indeed simulation results near the end of this section show that the FRQN tracking error grows faster than that of NLMS as the source velocity of the desired signal (hence also its fade rate or Doppler) increases. That corresponds to decreasing Φ , but since Φ and σ_{Φ}^2 are not easily quantifiable from the scenario parameters, the contribution of the tracking error may not dominate the *total mse* in (96).

Some further refinements with respect to improving tracking performance in *non-stationary* environments were also conceived. They do not affect the foregoing expressions for mean squared errors, because the same approximations still hold. To illustrate, it is instructive to consider the instantaneous updating equations in terms of projections parallel to (onto) and orthogonal to the subspace spanned by the data or input

vectors Y_k . (Similar reasoning in [39] inspired this approach.) For example, the NLMS algorithm (88) or (89) can be rewritten as

$$W_k = P_{Y_k}^\perp W_{k-1} + \frac{Y_k r_k^*}{Y_k^H Y_k} \quad (109)$$

where the projection orthogonal to the current data Y_k is recognized as

$$P_{Y_k}^\perp = I - Y_k (Y_k^H Y_k)^{-1} Y_k^H \quad (110)$$

and its complementary projection onto the current data subspace is simply

$$P_{Y_k}^I = I - P_{Y_k}^\perp. \quad (111)$$

Note that

$$P_{Y_k}^\perp (Y_k r_k^*) / (Y_k^H Y_k) = 0 = P_{Y_k}^\perp S_{1,k} \quad (112)$$

so that (109) can be rewritten as

$$(W_k - S_{1,k}) = P_{Y_k}^\perp (W_{k-1} - S_{1,k}). \quad (113)$$

The bracketed expressions in (113) effectively represent the nulling components of the weight vectors. When those are orthogonal to the current data, $P_{Y_k}^\perp$ has no effect on them so they are not updated in (113)⁵. That is the correct behaviour, because being orthogonal to the data means that they are doing their nulling job, as their inner product with Y_k is zero.

⁵ Since all the past and present data vectors Y_k are not mutually *orthogonal*, the *order* of the projections affects the outcome [34]. It is implicitly assumed herein that in a stationary steady state, the history of projections relative to the data vectors has had the same effect as using the orthogonal basis vectors which span the data subspace, in place of the data vectors themselves.

Supposing that this happy state has existed in a stationary scenario, the NLMS algorithm having converged, whereupon the current data suddenly sprouts a component which is *not* orthogonal to the nulling components of the weights, i.e. falls within the subspace spanned by $(W_{k-1} - S_{1,k})$. Then $(W_{k-1} - S_{1,k})$ will no longer equal the LHS of (113) and the LHS will be updated to equal the new RHS, because the new $P_{Y_k}^\dagger$ will affect $(W_{k-1} - S_{1,k})$ with the change in the data. That is to say, the NLMS algorithm will begin tracking this non-stationarity in the data. (As an aside, note that $Y_k^H W_k = r_k^*$ according to (109), but it is not an available output at iteration time k .) Therefore the "gain" in the NLMS algorithm may remain infinite as in (88) and (89).

Now consider the FRQN algorithm. It is instructive to examine it in terms of the geometric interpretation, but on an instantaneous basis. To that end, equation (91) is rewritten in the form of (82) as

$$\tau_1 \Delta(\tau_2 \Delta W_k) = \frac{-1}{\tau_2} \left[I + g \left(\frac{\tau_2}{q} \right) Y_k Y_k^H \right] (\tau_2 \Delta W_{k-1}) - g \nabla_{W_k} |e_k|^2. \quad (114)$$

Recall that $\nabla_{W_k} |e_k|^2 = Y_k e_k^*$, which is parallel to the current data vector Y_k . It was desirable to make $g \rightarrow \infty$ so as to promote Quasi-Newton (QN) convergence in the stationary case. The implications of that in (114) would be to effectively neglect the diagonal augmentation "I" in the middle term, i.e.

$$\tau_1 \Delta(\tau_2 \Delta W_k) = \frac{-g}{q} [Y_k Y_k^H] (\tau_2 \Delta W_{k-1}) - g Y_k e_k^*. \quad (115)$$

Now the middle term is always parallel to the last term in (115). That is because

$(-g/q)[Y_k Y_k^H](\tau_2 \Delta W_{k-1}) = (-g/q)(Y_k^H Y_k) P_{Y_k}^\dagger (\tau_2 \Delta W_{k-1})$, i.e. it is directly proportional to a projection of ΔW_{k-1} onto the subspace spanned by the current data vector Y_k , and $-g Y_k e_k^*$ is simply a scalar multiple of the same data vector.

Supposing now that the happy state exists whereby the scenario is stationary and the weight *steps* move along the QN optimal direction toward W_{opt} i.e. $\Delta^2 W_k = 0$ on the LHS of (114) and (115). If the data suddenly becomes non-stationary, but this time in such a way that it sprouts a new dimension which is orthogonal to $(\tau_2 \Delta W_{k-1})$ in particular, then that will produce no change in (115) because $Y_k^H (\tau_2 \Delta W_{k-1}) = (Y_k^H - Y_{k, new}^H) (\tau_2 \Delta W_{k-1})$, where $Y_{k, new}^H (\tau_2 \Delta W_{k-1}) = 0$. Thus the scalar coefficients of the augmented data vector in both terms on the LHS of (115) remain the same, consequently *no change* in the weight steps' *direction* will occur in the RHS of (115) either. The weights may be adapting, but not tracking the change in the QN direction brought about by the data non-stationarity. However, a change *will* be produced in (114) with finite g , because the weight step is not entirely projected onto the data subspace. Therefore a finite "gain" g is required to maintain near-optimal QN tracking whereby the weight steps are *always* redirected toward the optimal direction leading to W_{opt} in all types of non-stationary scenarios. To remain sufficiently close to QN behaviour in ill-conditioned scenarios, it is desirable to keep

$$\boxed{g \left(\frac{\tau_2}{q} \right) \lambda_{\min} > 1} \quad (116)$$

according to (114), so the optimal FRQN algorithm for non-stationary scenarios would be given by equation (80) with (116), i.e.

$$\boxed{W_k = \left[\frac{a_1}{a_0} I + \left(\frac{b_1/b_0 - a_1/a_0}{a_0/b_0 + Y_k^H Y_k} \right) Y_k Y_k^H \right] W_{k-1} - \left(\frac{\tau_1 \tau_2}{a_0} \right) \left[I - \frac{Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] W_{k-2} + \left(\frac{g}{b_0} \right) \frac{Y_k r_k^*}{a_0/b_0 + Y_k^H Y_k}} \quad (80)$$

with $a_0 = \tau_2(\tau_1 + \gamma)$, $b_0 = g(1 + \tau_2/q)$, $a_1 = 2\tau_2(\tau_1 + \gamma/2)$ and $b_1 = g\tau_2/q$. The parameters are chosen as $\gamma = 1/\tau_2$, $g = 1/\lambda_{\min}$, $\tau_2 = 4N$ and $\tau_1 = \tau_2/4$, and, as with NLMS and RLS, the available output is still given by

$$y_k = W_{k-1}^H Y_k. \quad (93)$$

From another viewpoint, a change in ΔW_{k-1} in a direction orthogonal to the current data Y_k will not affect (115) either, and such a component of the weight step can grow at will.

It is worth noting that a similar interpretation of tracking of the "conventional" (order-of- N^2) RLS algorithm may also be formulated. Starting with its QN form (30), a change in the scenario will cause a change in the performance surface and hence a change in the instantaneous gradient $Y_k \alpha_k^*$, which may be expressed as

$$\Delta(R_k \Delta W_k) = -\Delta(Y_k \alpha_k^*). \quad (117)$$

Expanding the LHS according to the product rule for (time-) derivatives leads to

$$R_k \Delta^2 W_k + (\Delta R_k) \Delta W_k = -\Delta(Y_k \alpha_k^*) \quad (118)$$

where equation (25) will be used to supply $\Delta R_k = Y_k Y_k^H$ with $\lambda = 1$. It somewhat resembles (115) when rearranged as

$$R_k \Delta(\Delta W_k) = -Y_k Y_k^H (\Delta W_k) - \Delta(Y_k \alpha_k^*). \quad (119)$$

Here again, any change in ΔW_k that is orthogonal to the current data Y_k will not affect the equation and may grow at will. Similarly, focusing on the scalar quantity $Y_k^H (\Delta W_k)$, a change in the data which is orthogonal to ΔW_k will not alter the equation, hence not the direction of adapting weight-steps on the LHS either ($\Delta^2 W_k = 0$). Therefore RLS will not begin tracking this non-stationarity.

It is quite clear that RLS with $\lambda \equiv 1$ does not in fact track at all in non-stationary scenarios because it has infinite memory which overwhelms any updates brought in with new data. In the above context, the "gain" g and "forgetting factor" λ affect similar functions in the FRQN and RLS algorithms, respectively, and cannot be set to their respective limits in non-stationary scenarios, but effectively remain close to them.

Returning now to characterization, both the NLMS and FRQN algorithms were determined to converge stably in the mean square in both stationary and non-stationary signal scenarios, ill-conditioned and well-conditioned alike. The mse of the refined FRQN algorithm was found to be much less than that of the NLMS algorithm in its fastest form, in stationary scenarios. That was also a finding of the author of [D1], who investigated their analogue, continuous-time counterparts. The RLS algorithm (version II) also converged stably, although in the ill-conditioned scenario more slowly than the theoretical $2N$ to $3N$ iterations. (The eigenvalue spread with non-unity forgetting factor has some influence on the convergence rate⁶.) Figure 19 shows the averaged learning curves for the well-conditioned scenario; Figures 20 and 21 show the corresponding ones of the FRQN and NLMS algorithms, respectively.

⁶ Dr. M. L. R. de Campos, University of Victoria, private communication.

Figure 19: Averaged square error and mmse for the RLS algorithm in the well-conditioned scenario.

Figure 20: Averaged square error, theoretical mse and mmse For FRQN.

Figure 21: Averaged square error, theoretical mse and mmse for NLMS algorithm

For curiosity, the "ordinary second-order LMS" algorithm as described in [26] was converted to discrete-time using the backward-difference method. It was found to behave well with $g = 1/\lambda_{\min}$ in the well-conditioned scenario, and to have about 10 dB greater mse than FRQN in the ill-conditioned scenario. (Convergence times were similar to FRQN) The theoretical mse was found to diverge in the parallel scenario. With $g \rightarrow \infty$ it was found to be completely unstable. It was therefore excluded from further investigations.

Simulations with independently Rayleigh fading and moving signal sources were next conducted to determine which algorithm has better tracking properties, by comparing their BER performances. Again, three interferers and one desired signal (with a perfect replica serving as the reference) were present (see Appendix F). Pseudo-random QPSK sequences with one symbol per sample were again used for all the signals.

Received-signal sequences were not low-pass filtered as that effectively introduces inter-symbol interference. Flat Rayleigh fading was modelled in the same ill-conditioned and well-conditioned scenarios by simply multiplying the individual source signals by complex zero-mean Gaussian random sequences which have been low-pass filtered to the corresponding source's Doppler frequency with 5-th order Chebyshev filters. The filtering approximates the fading or Doppler spectrum as derived in [40]. It is a simplified model which is deemed adequate where the receiving array is high above the ground and clutter, and its elements are separated by half wavelength so the fading among them is highly correlated. (Space-diversity antennas in such applications are typically spaced more than 10 wavelengths apart to decorrelate the fading among them so it has a correlation coefficient < 0.7 [41].) Equivalently, it is assumed that the delay-spread is a sufficiently small fraction of the symbol interval so as not to cause inter-symbol interference. Full complex multiplications must be used to multiply each sample of the fading sequence with a corresponding sample of the source's signal sequence. The fading sequences were normalized so as not to affect the average received signal powers.

Each scenario was first modelled without fading to determine the BER obtained with the RLS, NLMS and FRQN algorithms as a function of received signal-to-noise ratio (SNR). Relatively fast fading was then introduced into the same scenarios to determine its effects on the BER vs. SNR relationships for each algorithm. In the case of complete loss of Doppler tracking, it is expected that, with moderate to high SNR, each of the quadrature QPSK symbol streams would have the wrong polarity 1/2 of the time, during which binary bit-decisions would be only as good as random guesses, i.e. 1/2 of them wrong. Consequently, the BER would be $(1/2) \times (1/2)$ or close to 0.25 under complete loss of tracking.

According to more detailed simulations of similar scenarios with the LMS algorithm (see footnote # 1), performance was adequate for indoor mobile radio using BPSK data rates of 800 kbps with fading rates of 5 - 9 Hz, but was inadequate for 32 kbps with fading rate of 70 Hz in outdoor mobile radio environments. That translates to relative fade rates to data-rates of $6.25 - 11.25 \times 10^{-6}$ and 2.19×10^{-3} , respectively. In [13] the author found that the LMS algorithm is inadequate with fade-rate to data-rate ratio of 3.3×10^{-3} in IS-54 cellular mobile radio signalling and also with a relative fade rate of 5×10^{-4} in GSM signalling. These results therefore represent "fast" fade rates.

The simulations herein represent symbol rates of 30 ksps (one complex sample worth 2 bits per symbol) and fade rates of 47.2, 23.6, 102.3 and 78.7 Hz for the 4 sources. The relative fade rates are therefore in the region of 2×10^{-3} , i.e. the simulations represent conditions of "fast" fading. Each run consisted of the same 100 blocks of 1000 samples (i.e. the same 200 000 bits). Bit errors were counted only after the adaptation transients expired, i.e. 8000 iterations in all runs. The results represent steady-state performance and are summarized in the following plots. Note that the FRQN algorithm consistently gave 2 to 3 times better BER for the same SNR and fade rate than did the NLMS algorithm in its fastest form. The FRQN algorithm performed almost equally well as RLS at low to moderate SNR. Figure 22 below compares the NLMS, FRQN and RLS algorithms in the well-conditioned scenario.

Figure 22: Comparison of tracking performance in fast-fading, well-conditioned scenario.

To determine how much the fading degraded BER vs. SNR performance, stationary versions of the same scenarios were simulated with the same data and algorithms. The following table summarizes the available results.

Table 4: BER performance of algorithms in stationary scenarios			
SNR, dB	Algorithm	Well Conditioned	Ill-Conditioned
13.25, 13.1	NLMS	1.325×10^{-5}	4.1×10^{-3}
13.4, 13.2	FRQN	$< 5 \times 10^{-6}$	3.894×10^{-4}
13.6, 13.13	RLS	$< 5 \times 10^{-6}$	2.041×10^{-5}

Due to the excessively long run-times of the simulations required to count the smaller error rates at higher SNR values, those results were not computed. Figure 23 compares the BER versus SNR performances of the RLS, NLMS and FRQN algorithms in the ill-conditioned, fast-fading scenario.

Figure 23: Comparison of tracking performance in fast-fading ill-conditioned scenario

Again notice that the FRQN algorithm performs nearly equally well as the more computationally intensive RLS algorithm at low to moderately high SNR levels, and both algorithms track better than the NLMS algorithm throughout the range of SNR. Because the *same* bit streams were used in each simulation run, no error bars are shown on the plots of BER. Thus the *relative* BER figures are legitimate indicators of the relative tracking performance of the algorithms.

In the above comparisons the fade rate of the desired signal was about 1.4×10^{-3} relative to its symbol rate, which qualifies it as "fast" fading with respect to the cited results of other investigators. Although the interfering-signals' relative fade rates were up to twice as fast, the algorithms did not have to track them in the same sense, as they were being nulled.

Bit-error-rate at a moderate SNR is next examined as a function of fade rates for the RLS, NLMS and FRQN algorithms in the same basic scenarios. Only the desired-signal's source velocity (equivalently, fade-rate or Doppler) was varied as shown on the horizontal axes, while the interferers' fade-rates remained the same as in Figures 22 and 23. The SNR of the desired signal was constant and moderately high, so that noise would not mask the effects of the tracking error. (According to Figures 22 and 23, the SNR of 30 dB and greater has only a small effect on the BER in the presence of the fast fading).

Figure 24: Comparison of tracking performance with fade-rate in well-conditioned scenario.

Figure 25: Comparison of tracking performance with fade-rate in ill-conditioned scenario.

Note that the tracking error does not dominate the total mean squared error (which is responsible for the BER in these simulations) until the fade rate is extremely high. Therefore the FRQN enjoys a lower mean-squared error than NLMS at most practical fade rates, even though its tracking error is higher according to Appendix D. Most of the errors are due to the *emse* caused by steady-state stochastic misadjustment, which is much lower for FRQN than for NLMS (as also shown in Appendix D).

3.7 Computational Aspects:

The FRQN algorithm in its final form (e.g. equation (80) or (92)) can be rewritten in a more efficient form for the purpose of computational implementation. When the original parameters (time-constants, gains, etc.) are substituted, the recursive difference equation for coefficient updating becomes

$$W_k = \left(\frac{2\tau_2\tau_1 + \gamma\tau_2}{\tau_2\tau_1 + \gamma\tau_2} \right) \left[I - \frac{Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] W_{k-1} + \left(\frac{\tau_2}{q + \tau_2} \right) \left[\frac{Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] W_{k-1} - \frac{\tau_2\tau_1}{\tau_2\tau_1 + \gamma\tau_2} \left[I - \frac{Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] W_{k-2} + \left(\frac{q}{q + \tau_2} \right) \frac{Y_k r_k^*}{a_0/b_0 + Y_k^H Y_k} . \quad (120)$$

This brings to light some opportunities for manipulation so as to share some of the common factors. Defining the (positive, real) scalar at the k -th iteration

$$P_k = \frac{a_0}{b_0} + Y_k^H Y_k , \quad (121)$$

allows one to eventually simplify the FRQN algorithm equation to

$$W_k = W_{k-1} + \left(\frac{\tau_1\tau_2}{\tau_1\tau_2 + \gamma\tau_2} \right) \left[I - \frac{1}{P_k} Y_k Y_k^H \right] (W_{k-1} - W_{k-2}) - \frac{1}{P_k} \left(\frac{q}{q + \tau_2} \right) Y_k (W_{k-1}^H Y_k - r_k)^* . \quad (122)$$

The computational requirements in terms of operations per iteration are tabulated in Tables 5 and 6 below.

Based on these tables, a block diagram using the SIMULINK[®] software which can be added to MATLAB, from which a real-time implementation of the FRQN algorithm can be developed, is described in Appendix H. Its simulation is compared to those using NLMS and RLS algorithms in the demonstration application of the software.

Table 5: Initial FRQN Parameter Computations	
$p_0 = \left(\frac{\tau_1 \tau_2}{\tau_1 \tau_2 + \gamma \tau_2} \right)$	$\tau_2 = 4N, \tau_1 = \tau_2/4, \gamma = 1/\tau_2$
$p_1 = \left(\frac{q}{q + \tau_2} \right)$	$q = 2$
$p_2 = \frac{a_0}{b_0} = \frac{\tau_2(\tau_1 + \gamma)p_1}{g}$	g satisfies equation (116): $g \gg 1$.
$W_0 = \underline{0}$	-----
$\Delta W_0 = \underline{0}$ or $W_{-1} = \underline{0}$	-----

Table 6: FRQN Operations per Iteration

Step	$\times$	$+$ or $-$	$/$
$p_k = p_2 + Y_k^H Y_k$	N (complex)	N+1 (real)	0
$y_k = W_{k-1}^H Y_k$	N (complex)	N (real)	0
$- \alpha_k = r_k - y_k$	0	1 (complex)	0
$\psi_k = -(p_1/p_k) \alpha_k^*$	1 (complex)	0	1 (real)
$D_k = Y_k \psi_k$	N (complex)	0	0
$\Delta W_{k-1} = (W_{k-1} - W_{k-2})$	0	N (complex)	0
$p_0 \Delta W_{k-1}$	N (complex)	0	0
$e_k = Y_k^H \Delta W_{k-1}$	N (complex)	N (complex)	0
$\zeta_k = (p_0/p_k) e_k$	1 (complex)	0	1 (real)
$G_k = Y_k \zeta_k$	N (complex)	0	0
$W_k = W_{k-1} + p_0 \Delta W_{k-1} - G_k + D_k$	0	3N (complex)	0
TOTAL	6N + 2	7N + 2	2

At each iteration the FRQN algorithm in its preferred form requires $6N + 2$ complex multiplications, $7N + 2$ additions and 2 real divisions. (Note that $Y_k^H W_{k-1} = y_k^*$ and $Y_k^H W_{k-2}$ in equation (95) are scalars.) It is therefore more efficient than the conventional RLS algorithm which requires $4N^2 + 4N + 2$ complex multiplications and additions per iteration, for $N \geq 1/2$ (i.e. all practical array sizes).

Precision requirements for representing the weighting coefficients that comprise W are derived first for the steady state W_{opt} , followed by a treatment which considers the minimum step-sizes required to avoid stalling the algorithm. The dynamic range of

$$W_{opt} = R^{-1} S \quad (123)$$

may be estimated by making use of the eigendecomposition of the Hermitian matrix R and noting that the similarity transformation E is unitary so $E^{-1} = E^H$, whereupon

$$R^{-1} = [E \Lambda E^{-1}]^{-1} = E \Lambda^{-1} E^H \quad (124)$$

where the columns of E are orthonormal eigenvectors $\{U_n\}$ and the diagonal entries of Λ are the corresponding positive eigenvalues of R [34]. Since E is unitary, it is norm-preserving so

$$|W| = |E^H W| = |E^H E \Lambda^{-1} E^H S| \quad (125)$$

which may subsequently be expanded as

$$|W| = \left| \sum_{n=1}^N \frac{1}{\lambda_n} U_n^H S \right| \quad (126)$$

Since all the eigenvalues are real and positive (assuming non-zero uncorrelated receiver noise is included [31,34]), it is clear that $|W|$ is bounded by

$$\frac{1}{\lambda_{\max}} |E^H S| < |W| < \frac{1}{\lambda_{\min}} |E^H S| \quad (127)$$

so it may be said that the ratio of the largest to the smallest weight vector norm is given by

$$\frac{|W_{\max}|}{|W_{\min}|} = \frac{\lambda_{\max}}{\lambda_{\min}} \leq \frac{N p_{\max}}{\sigma^2} \quad (128)$$

where $p_{\max}$ is the maximum incident signal power per element and σ^2 is the per-element receiver noise power, N being the total number of elements in the array. This result is independent of any scalar that may premultiply W_{opt} in the steady state and represents the bounds of the weight coefficients themselves. Since the signal-to-noise ratio (SNR) of a typical receiving system may be as high as 60 dB, they should be represented by at least 12 bits assuming a resolution of roughly 6 dB per bit and 2 bits to specify the quadrant in the complex plane.

Next, the effects of coefficient quantization on the adaptation increments (time steps) of W_k will be determined, with a view to avoid stalling of the algorithm. To that end, equation (122) is rewritten as

$$\begin{aligned} \tau_2 \Delta W_k = & \left(\frac{\tau_1}{\tau_1 + \gamma} \right) \left[I - \frac{1}{p_k} Y_k Y_k^H \right] \tau_2 \Delta W_{k-1} - \\ & - \left(\frac{q}{\tau_2 + q} \right) \frac{\tau_2}{p_k} Y_k \alpha_k^* \end{aligned} \quad (129)$$

in which some components of ΔW_k are quantized to zero when the FRQN algorithm stalls, presumably near its converged state. In that case it may be assumed that the weight steps satisfy $\Delta W_k \approx \Delta W_{k-1}$ and (129) rearranged to isolate them as

$$\left[\left(\frac{\gamma}{\tau_1 + \gamma} \right) I + \frac{1}{p_k} Y_k Y_k^H \right] \Delta W_k = \left(\frac{q}{\tau_2 + q} \right) \frac{1}{p_k} Y_k \alpha_k^* \quad (130)$$

so the matrix-inversion lemma can be applied to its LHS. Suppose the n -th component of ΔW_k stalls, i.e. its magnitude falls below that of the quantization error. The quantization error with B bits is assumed to be uniformly distributed with mean squared value given by

$$\eta_{\Delta W_k}^2 = \frac{\mathfrak{R}_w^2 2^{-2B}}{12} \quad (131)$$

where $\mathfrak{R}_w$ is the range of weight or coefficient voltages. After some manipulations with the help of Appendix E, one finds that the n -th component of the weight-vector step stalls when

$$\left(\frac{q}{\tau_2 + q} \right)^2 \left(\frac{\tau_1 + \gamma}{\gamma} \right)^2 \left(\frac{1}{|p_k|^2} \right) \mathcal{E} \left\{ \left[\left[I - \frac{Y_k Y_k^H}{Y_k^H Y_k + \frac{\gamma p_k}{\tau_1 + \gamma}} \right] Y_k \alpha_k^* \right]_n^2 \right\} \leq \eta_{\Delta W_k}^2 \quad (132)$$

where $|p_k|^2$ is averaged separately and further approximated as $p_k = p_2 + Y_k^H Y_k \approx Y_k^H Y_k$ so that $\mathcal{E}\{|p_k|^2\} \approx \text{Tr}[R]$. Then equation (132) eventually simplifies to

$$\mathcal{E}\{|Y_k \alpha_k^*|_n^2\} \leq \frac{\mathfrak{R}_w^2 2^{-2B}}{12} \left(\frac{\tau_2 + q}{q} \right)^2 \left(\frac{\tau_1 + 2\gamma}{\tau_1 + \gamma} \right)^2 \text{Tr}[R] \quad (133)$$

The expectation on the LHS involves the n -th component of Y_k (which is denoted by $y_{n,k}$). It is evaluated using the Gaussian moment-factoring theorem [31] which states that

$$\mathcal{E}\{y_{n,k} \alpha_k^* \alpha_k y_{n,k}^*\} = \mathcal{E}\{y_{n,k} \alpha_k^*\} \mathcal{E}\{\alpha_k y_{n,k}\} + \mathcal{E}\{y_{n,k} y_{n,k}^*\} \mathcal{E}\{\alpha_k \alpha_k^*\} \quad (134)$$

Close to steady state, where stalling is most likely to occur, α_k is near its minimum so the orthogonality condition (see Appendix B) is assumed to apply, i.e. $\mathcal{E}\{\alpha_k \alpha_k^*\} = mmse$ and $\mathcal{E}\{y_{n,k} \alpha_k^*\} = \mathcal{E}\{\alpha_k y_{n,k}^*\} = 0$ for all n . Therefore (133) simplifies to

$$\mathcal{E}\{|y_{n,k}|^2\} mmse \leq \frac{\mathfrak{R}_w^2 2^{-2B}}{12} \left(\frac{\tau_2 + q}{q} \right)^2 \left(\frac{\tau_1 + 2\gamma}{\tau_1 + \gamma} \right)^2 \sum_{m=1}^N \mathcal{E}\{|y_{m,k}|^2\}. \quad (135)$$

Assuming that each element or beam receives approximately equal average power and taking advantage of simplifications due to the conditions $\tau_2/q > 1$ and $\tau_1 > \gamma$ allows one to express the stalling condition as

$$mmse \leq \frac{\mathfrak{R}_w^2 2^{-2B} N}{12} \left(\frac{\tau_2}{q} \right)^2. \quad (136)$$

It means that τ_2/q should be small so the FRQN algorithm would tend to take larger steps which would not be quantized to zero and would also result in a smaller residual error in the LHS of (136). A similar analysis of the NLMS algorithm with $\beta \approx 1/(Y_k^H Y_k)$ averaged separately leads to

$$mmse \leq \frac{\mathfrak{R}_w^2 2^{-2B}}{12 \beta^2 \mathcal{E}\{|y_{n,k}|^2\}} = \frac{\mathfrak{R}_w^2 2^{-2B}}{12} N^2. \quad (137)$$

Since $\tau_2/q = 4N/2$, the FRQN algorithm inherently has $4N^2$ times greater residual error than the NLMS algorithm using the same number of bits to represent the same range of weights.

Equivalently, the number of bits, B , required to represent the weights of a given range $\mathfrak{R}_w$ and achieve a given $mmse$ is given by

$$B \geq \frac{1}{2} \log_2 \left(\frac{\mathfrak{R}_w^2 N (\tau_2/q)^2}{12 \text{ mmse}} \right) = \frac{1}{2} \log_2 \left(\frac{\mathfrak{R}_w^2 4N^3}{12 \text{ mmse}} \right) \quad (138)$$

for the FRQN algorithm, and by

$$B \geq \frac{1}{2} \log_2 \left(\frac{\mathfrak{R}_w^2 N^2}{12 \text{ mmse}} \right) \quad (139)$$

for the NLMS algorithm. Thus it is seen that the FRQN algorithm requires $1 + \frac{1}{2} \log_2(N)$ more bits than the NLMS algorithm in order to avoid stalling in the same environment.

As for quantization errors, η_Y , in the data, Y_k , they do not destroy the Hermitian property (hence stable eigenvalues) of the array covariance matrix, which is effectively the state-transition matrix in both the FRQN and NLMS algorithms. That is because

$$(Y_k + \eta_Y)(Y_k^H + \eta_Y^H)$$

is always Hermitian. Therefore, unlike in the RLS algorithm, the data quantization errors in NLMS and FRQN are not as serious with regard to stability [31, Ch. 19]. Note also that the state-transition matrix in NLMS (equations (109), (110)) and FRQN (equation (122)) algorithms are exactly (for NLMS) or very nearly (for FRQN) projection operators, which reduce the norms of their target vectors, in this case the weight or weight-steps. Therefore it is unlikely that the weights in these two algorithms could grow without bound; indeed they have been observed to remain within a steady range in each simulation run. In that respect it may also be noted that analogue, continuous-time variants of the FRQN algorithm [5], [D1] with offsets introduced in various parts of their

feedback loops have been analyzed and verified to operate stably, so the transfer-functions for the weight quantization errors in FRQN are also expected to be stable.

Quantization of the various algorithm parameters should intuitively be sufficient to resolve the dominant pole of FRQN from unity because

$$|z_{p_2}| = \frac{\tau_2/q}{1 + \tau_2/q} < 1 \quad (140)$$

is essential for its stability. It implies that the quantization error with B bits should be

$$\frac{1}{2} LSB = 2^{-(B+1)} \leq 1 - \frac{\tau_2/q}{1 + \tau_2/q} \quad (141)$$

so at least $B > \log_2(1 + 2N) - 1$ bits should be used. For most practical array sizes, N, that number of bits is exceeded by other requirements.

Chapter 4

Some Thoughts on Reference Signals:

In this chapter some techniques for providing the reference signal $r(t)$ or r_k will be examined. It is not the intent to investigate in detail the convergence and tracking behaviour with respect to the reference signal in detail, but rather to examine theoretically different techniques in terms of their effects on the output *mse* and stability of the algorithms. The role of the reference signal in an adaptive array is illustrated in Figure 26.

4.1 Statistical Requirements of Reference Signals:

As explained in [42], the reference signal r_k must be more correlated with the signal of interest (SOI) than with any other signals such as interference. That is necessary in order for the statistic S in (B4) to be a (scaled) replica of the steering vector associated with the SOI, and thus to yield the optimal weighting solution in the steady state. (Other steering vectors give suboptimal solutions when substituted for S in W_{opt} , meaning they do not maximize the output signal-to-interference-plus-noise ratio (SINR), only minimize the interference power.)

For completeness this optimality condition is derived in detail; it also serves to reveal the structure of the array covariance matrix R , which will prove useful in subsequent developments. Figure 26 serves to relate the notation to the system configuration. Maximization of the output SINR in terms of W is performed similarly as the mean squared output error minimization was done at the outset (equations (1) - (11)). Assume the steering vector of the SOI is S , and the reference signal is such that

$$\mathcal{E}\{Y_k r_k^*\} = S_1 \quad (142)$$

and all signals are mutually uncorrelated. The average output power is

$$\mathcal{E}\{y_k y_k^*\} = W^H R W \quad (143)$$

with R given in Appendix B by (B3) in which the signal-vectors have the following form

$$Y(t) = \sum_{m=1}^M a_m(t) e^{j\omega_m t} S_m + \eta(t) = Q A(t) + \eta(t) \quad (144)$$

(which is essentially the same in discrete-time), where $a_m(t)$ is the complex envelope and ω_m is the radian carrier frequency of the signal of the m -th source. The m -th column of Q is the steering vector of the m -th incident signal from a point-like source at (θ_m, ϕ_m) in the far field; the n -th entry of that steering vector is the response of the n -th array element at (x_n, y_n, z_n) to the m -th signal, relative to the response of the first element, i.e.

$$s_{m,n} = G_n(\theta_m, \phi_m) e^{j\beta_{n,m}} \quad (145)$$

with

$$\beta_{n,m} = \frac{2\pi}{\lambda_m} (x_n \sin(\theta_m) \cos(\phi_m) + y_n \cos(\theta_m) \cos(\phi_m) + z_n \sin(\theta_m)) . \quad (146)$$

The complex envelopes of the signals $\{a_m(t)\}$ are uncorrelated with each other and with the element noises in $\eta(t)$, the latter being also uncorrelated among the elements. The $N \times N$ array covariance matrix computed with (144) is then seen to possess the following structure [34]:

Figure 26: Characterization of an Adaptive Array System with Reference Signal

$$\begin{aligned}
R &= Q \mathcal{E}\{A(t)A^H(t)\}Q^H + \mathcal{E}\{\eta(t)\eta^*(t)\} \\
&= QPQ^H + \sigma^2 I \\
&= \sum_{m=1}^M P_m S_m S_m^H + \sigma^2 I
\end{aligned} \tag{147}$$

where P is the $M \times M$ diagonal signal-covariance matrix whose diagonal elements are the set of incident signal-powers, $\{p_m\}$, and the steering matrix is

$$Q = [S_1 | \dots | S_M] \tag{148}$$

It is useful to write separately the contribution of the SOI to R as

$$R_S = p_1 S_1 S_1^H \tag{149}$$

and that of the interference and noise as

$$R_I = \sum_{m=2}^M P_m S_m S_m^H + \sigma^2 I \tag{150}$$

so that

$$R = R_S + R_I. \tag{151}$$

Then the output SINR may be expressed as the ratio

$$SINR = \frac{W^H R_S W}{W^H R_I W} \tag{152}$$

which is maximized by setting

$$\nabla_W \{SINR\} = 0. \tag{153}$$

That obtains

$$2R_I W [W^H R_S W] = 2R_S W [W^H R_I W] \quad (154)$$

which leads to

$$R_S W = (\text{SINR}) R_I W. \quad (155)$$

Note that SINR is a scalar and is maximum when $W = W_{opt}$ in (152). When that and (149) and (150) are substituted, the result is

$$p_1 S_1 S_1^H W_{opt} = \left(\frac{p_1 W_{opt}^H S_1 S_1^H W_{opt}}{W_{opt}^H R_I W_{opt}} \right) R_I W_{opt} \quad (156)$$

which may be rearranged through division by $p_1 S_1^H W_{opt} \neq 0$ to obtain

$$S_1 = \left(\frac{W_{opt}^H S_1}{W_{opt}^H R_I W_{opt}} \right) R_I W_{opt}. \quad (157)$$

The optimal weight vector which maximizes the output SINR is therefore given by

$$W_{opt} = (\text{scalar}) R_I^{-1} S_1, \quad (158)$$

where it can be verified that the "scalar" is arbitrary by substituting (158) into (157).

It can now be shown that the solution derived for the optimal weight vector as

$$W_{opt} = R^{-1} S \quad (159)$$

elsewhere in the thesis is in fact optimal in the above sense when $S \equiv S_1$, by substituting

the expanded form of R in (159)

$$W_{opt} = [R_I + p_1 S_1 S_1^H]^{-1} S, \quad (160)$$

then applying the matrix-inversion lemma to obtain

$$W_{opt} = \left[R_I^{-1} - \frac{R_I^{-1} S_1 S_1^H R_I^{-1}}{S_1^H R_I^{-1} S_1 + 1/p_1} \right] S. \quad (161)$$

Replacing S with S_1 allows one to factor out $R_I^{-1} S_1$ and simplify it to

$$W_{opt} = \left(\frac{1}{p_1 S_1^H R_I^{-1} S_1 + 1} \right) R_I^{-1} S_1. \quad (162)$$

Since the scalar in the optimal solution is arbitrary, the above solution also qualifies. For simplicity, that scalar was implicitly taken as unity in other developments in the thesis. In this treatment it is allowed to have any scalar multiple of the SOI steering vector when using the reference signal to obtain it.

Another statistical requirement of r_k is that it should not contribute too much error to the optimized output. In an adaptive array it is permissible to have a large mean squared output error only if the major contribution is to the *mmse*. For example r_k may contain other signals that do not correlate with the SOI, which will make *mmse* large, but the only contribution to the output error (i.e. SINR) that matters is the *emse*, which will be large only if the algorithm causes the weights to have a high jitter variance because *emse* is directly proportional to *mmse* by the misadjustment ratio. Only the *emse* affects the quality of the adaptive-array output, so it is important to use an algorithm with low misadjustment as well as a reference signal with minimum "contamination". (This

may be seen by recognizing that the output y_k is compared to the reference *after* the output is branched out to the receiver, so the receiver never sees the reference signal and its "contamination" - see also Figure 26.)

Finally it should be remembered that r_k must be coherent with the SOI at the radio frequency (RF) or at an intermediate frequency (IF) so the relative phases and amplitudes due to propagation are preserved, prior to demodulation, because demodulation is not possible in the presence of the interfering signals.

4.2 Techniques:

Two common types of techniques are used for providing reference signals in adaptive arrays. One type involves sources of r_k which are external to the adaptive array system, such as local generators of known "training" signals which are the same as transmitted training signals periodically inserted in the SOI. In other "external" arrangements, an unknown training signal matched to the SOI may be transmitted on a different channel simultaneously with it. The other type of technique involves derivation of the reference signal r_k from the output y_k of the adaptive array itself. As such, it may be considered as an "internal" type of technique, so the adaptive array appears to be operating "blindly" with respect to the outside signal environment (see Figure 26). In this chapter the other configurations of algorithms such as a sidelobe canceller and steered gradient, which operate with constraints rather than reference signals, will not be considered.

External reference-provision techniques are relatively straightforward to apply. In the case of a known training signal, one may periodically transmit a known training sequence and synchronously generate the same sequence locally, modulated onto the

same RF or IF carrier. The two sequences are coherent and correlate in the adaptive algorithm to yield a scalar multiple of S_1 . Because they are identical up to a scalar factor, no excess output error is introduced; the scalar determines the overall gain provided by the weights.

In the case of a transmitted reference signal, excess output error does occur in practice, because both the received signal and r_k contain co-channel interference signals not of interest (SNOI)⁷. These SNOI should be uncorrelated with the SOI and with each other. (If either r_k or the received signal did not contain co-channel interference, then it could be reliably demodulated by the receiver and an adaptive spatial filter would not be needed to remove it.) Therefore in practice one must consider the amount of excess mean squared error which such a contaminated reference signal would contribute to the output and choose algorithm parameters so as to bring it within acceptable limits. At the end of this chapter, a method of using transmitted reference signals is described in the context of an adaptive base-station antenna system for mobile radio communications, which allows each frequency channel to be reused by several independent links by removing their mutual interferences. By using the same adapted antenna patterns (or close approximations thereof) for transmitting, it thus obtains a high service capacity with a given band of frequency spectrum. It is the TRAAAP system envisioned in section 2.4.

Internal reference-generating techniques are much more involved in their design and implementation, as no generic practical technique exists. The idea is to perform some nonlinear processing on the output signal y_k so that the component correlated with the SOI is enhanced and the uncorrelated ones are suppressed in the resultant r_k [42]. Such a process is necessarily matched in a loose sense to some known characteristic of the SOI, hence it is also nonlinear.

⁷ The transmitted reference is considered to be a separate additive signal transmitted at the same time and in the same frequency-band as the SOI, not a training sequence embedded in it.

Perhaps the most familiar method of internally generating a reference signal from the output of an adaptive array is the "despread-filter-respread" loop applied to direct-sequence spread-spectrum SOI [42]. The output signal is despread using a local oscillator signal which has been multiplied by the same pseudorandom spreading sequence of $\{+1, -1\}$ chips as the SOI. (When identical, arbitrary and synchronized sequences of $\{+1, -1\}$ are multiplied the result is a sequence of all $+1$ s, so the original SOI spectrum is obtained. Synchronization, code acquisition and other details are referred to the literature.) The process spreads the spectrum of all interfering signals which were not coded by the same sequence, so they will be severely attenuated upon passing through the filter whose bandwidth is just large enough to pass the despread SOI with minimal loss. The filter output is amplitude-limited, then respread by the same coded local oscillator so as to be coherent with the spread-spectrum SOI received by the array. That is then the required reference signal.

Another technique applied to FSK modulated SOI is to Markov-code the source's symbol stream so as to make the symbols more predictable by a linear predictor. It also adds redundancy to the SOI and increases its spectral bandwidth somewhat [42]. Each symbol produces its assigned frequency shift of the RF carrier. The output of the adaptive array is passed through an FSK demodulator and linear predictor, which estimates the next symbol's frequency-shift. That information is used to shift the frequency of a local oscillator (LO) likewise, so the targeted symbol will be shifted to base-band when multiplied by the LO with a higher probability than will any other random signal. This is then filtered to a narrow bandwidth and amplitude-limited before being restored to its original carrier frequency by the same LO. Statistically, the resultant r_k will contain more of the SOI than of any other signal whose frequency is less predictable in each symbol interval.

Yet another arrangement presents itself in the case of a SOI which has a cyclostationary autocorrelation function [43, 44]. Its spectrum exhibits self-coherence at frequency-intervals simply related to the cycle frequency. To preserve this property of the SOI, the signals must be oversampled, i.e. sampling rate must exceed the Nyquist frequency, or at least exceed twice the symbol rate. Among other signals, digitally-modulated [43] and analogue frequency-modulated (FM) signals to which a continuous tone (CW) has been added prior to FM [44], exhibit these properties. That is, parts of their RF spectra which are separated by known frequency-intervals (these are a function of the symbol rate or CW tone frequency) are highly correlated with each other. Parts of the spectra of other signals not of interest are not thusly correlated. This motivates the use of one spectral period of the received or output signal as the reference signal [43]. Alternatively, when each spectral interval is shifted to the same IF and their coherent sum maximized by a linear adaptive predictor structure, the SOI is very much enhanced [44].

Appendix G takes up the task of mathematically modelling a generic internal reference-generating subsystem, at least in its linear range. Aside from developing the condition required to meet the statistical properties of a suitable reference signal, it shows that such a subsystem can be used with the FRQN algorithm equally as well as it has been used for blind adaptation with NLMS algorithms.

The next section explores a particular external-reference technique conceived by the author, with which the FRQN algorithm is theoretically predicted to perform better than the NLMS algorithm.

4.3 The FRQN Algorithm in a Transmitted-Reference Application - The TRAAAP Concept:

Recall from section 2.4 that the TRAAAP concept is founded on the fact that there are $(M-1)/2$ times more unique *pairs* of frequency channels in a set of M of them, as there are single-frequency channels. Therefore the potential exists for serving up to $M(M-1)/2$ mobile communications links using only M frequency channels of spectrum allocation. This constitutes a combinatoric advantage over the M links that can be served with M frequency channels in a conventional frequency-division multiple-access (FDMA) scheme. The TRAAAP scheme requires M -element adaptive antenna arrays to separate co-channel signals on each frequency. It exploits the fact that only the SOI message-signal is the same on each of the frequencies in a pair, the interfering signals being different and uncorrelated between the two frequencies. (It may be regarded as a case of forced, non-overlapping spectral self-coherence.) With the description of section 2.4 in mind, it is recognized as a transmitted (external) reference scheme. In this section, some of the performance aspects relating to this reference scheme will be quantified in the context of a cellular mobile radiotelephone network. Although the TRAAAP concept would perhaps be more applicable to the emerging PCS (personal communications system) type of network where demand for capacity is greater, fading is slower and no standard communications schemes have yet been established, the cellular example is used because quantities such as channel bandwidth, signal (carrier) to interference (C/I) ratio, number of channels per cell and number of cells in a cluster are readily available.

It is assumed that a conventional cellular frequency-reuse plan based on clusters of 7 hexagonal cells is used. The TRAAAP concept is to be applied to each cell in order to increase the capacity of the M carrier-frequencies assigned to it. Because TRAAAP makes it possible to reduce emissions of co-channel interference and to null them out at

the receiving base-station, smaller cell-clusters having 3 or 4 cells may be used to further increase spectrum efficiency. First, it will be described how the frequencies of an M -channel set are used and assigned among the adaptive array processors of a base-station in the TRAAAP system. Not each frequency is reused the same number of times in a given cell, as may be seen in Table 7 below. Note that in this example, the $M=6$ frequencies can in principle support $M(M-1)/2=15$ users, which is more than twice as many as in a conventional FDMA cell.

In Table 7, signal S denotes the SOI is transmitted on that frequency from the mobile user and R denotes the same signal transmitted from the same mobile user on the corresponding frequency as a reference signal. As shall be seen later, TRAAAP favours a Time-Division Duplex (TDD) scheme for two-way communication, so S also denotes the frequency of transmission from the base-station to the mobile user, on one of up to $M-1$ orthogonal beams. As will become apparent later, S is associated with the received-signal vector and R is associated with the corresponding scalar reference signal. (The correlation of the vector containing S and scalar containing R in the same row of Table 7 would yield a steering vector for that user.)

Although S and R are at different frequencies, frequency-selective fading is not a serious issue, as S is associated with the steering vector, so it will not fade the same way at each array element. Fading of R can be mitigated by averaging normalized steering vectors as discussed below.

Table 7: A Frequency Plan for TRAAAP with $M = 6$							
Frequency $\blacktriangleright$ User $\blacktriangledown$	f_1	f_2	f_3	f_4	f_5	f_6	(Rx/Tx, Rx reference)
1	S	R					(f_1, f_2)
2	S		R				(f_1, f_3)
3	S			R			(f_1, f_4)
4	S				R		(f_1, f_5)
5	S					R	(f_1, f_6)
6		S	R				(f_2, f_3)
7		S		R			(f_2, f_4)
8		S			R		(f_2, f_5)
9		S				R	(f_2, f_6)
10			S	R			(f_3, f_4)
11			S		R		(f_3, f_5)
12			S			R	(f_3, f_6)
13				S	R		(f_4, f_5)
14				S		R	(f_4, f_6)
15					S	R	(f_5, f_6)

If each TRAAAP array-processor in Figure 27 has $M-1$ inputs (from the $M-1$ array elements), then this base-station adaptive antenna system can null out additional co-

channel interference from outside the cell: one source at f_1 , two at f_2 , etc., up to $M-1$ at f_{M-1} , in principle. In order not to stress any one frequency disproportionately, the same frequencies could be numbered in reverse order in the next tier of reuse cells.

Therefore it should be kept in mind that the resulting orthogonal beams which accomplish the co-channel signal-separation, will also afford a reduction in the cluster size from the current standard of $K=7$ cells per cluster to $K=4$ or even to $K=3$, thus further increasing the service capacity of a given band of spectrum. The issue of tracking fading signals will not be dealt with in the following exposition, except to acknowledge that, because the reference signals come from a fixed-gain antenna without diversity-combining, they will be fading and hence inducing the adapted array output signals to fade likewise. This may be mitigated by using the signal from the "fixed antenna" to generate a steering vector, and repeating that operation with each of the antennas in the array. The steering vectors from all antennas should each be divided by their first element to normalize their phase reference, then averaged as in [49].

Beginning at the mobile end of a TRAAP radio communications link, it is assumed to be transmitting a message at carrier frequency f_1 , along with $M-2$ other such mobile radios, which transmit their independent messages at the same frequency. Simultaneously, each mobile radio is also transmitting its message on a second carrier frequency derived from the same master oscillator in each mobile radio. Each of these second carriers is unique to each mobile radio, and is assigned to it from the set $\{f_2, \dots, f_M\}$. Thus a unique pair of "synchronous" carriers is associated with each mobile radio, with f_1 being common to at most $M-1$ of them (i.e. reused $M-1$ times). At the base-station, whose array-processing block diagram is shown in Figure 27, an M -element adaptive antenna array receiving system receives all the mobile signals. One antenna array element has a fixed-weight signal path, while the $M-1$ others supply signals to adaptive-array processors' inputs. (For enhanced operation in practice, the $M-1$ array elements would be combined through an orthogonal beamformer such as an FFT (fast

Figure 27: The receiving adaptive-array configuration for the base-station employing the TRAAP concept.

Fourier transform) or Butler matrix, to form $M-1$ beam outputs; that would not alter the course of this discussion.) Each antenna (or beam) output signal is processed by a coherent receiver and synchronously digitized and down-converted to a low intermediate frequency (IF). The digital IF signal is then channel-filtered (and perhaps decimated in time to reflect the Nyquist sampling interval for the narrower-band channels) to form a set of narrowband channels corresponding to the total set of carrier frequencies $f_1, \dots, f_{M-1}$, and $f_2, \dots, f_M$ for the fixed-weight antenna. These are designated by the boxes with partitions having the above frequency labels in Figure 27. The discussion focuses on the treatment of the mobile signals with f_1 as the common carrier in each of their RF pairs.

All the array outputs corresponding to f_1 are passed on as input signals to $M-1$ adaptive-array processors, each having $M-1$ adaptive coefficients or weights. Each one is also supplied with a different reference signal, obtained in a similar manner from the digitized and down-converted outputs of the fixed-weight antenna, i.e. corresponding to $f_2, \dots, f_M$ in this case. Note that all digitized and channelized signals are presumed to have been further converted to the *same* IF or to complex baseband form using "synchronous" oscillators (i.e. sinewaves derived from the same source, timed by a common clock). Each adaptive processor thus converges so as to produce an output signal which consists mostly of just the signal which is the same at f_1 and at the reference-frequency supplied from the fixed-weight antenna, i.e. the message signal modulated onto one unique pair of carriers transmitted by a mobile radio.

It can be seen therefore, that $M-1$ output signals corresponding to signals transmitted with f_1 as the common carrier of a pair will be produced, and $M-2$ outputs similarly corresponding to f_2 , and so on, to $M-(M-1)$ outputs corresponding to f_{M-1} . The total number of separate output signals which can thus be produced by the base-station adaptive arrays is computed as

$$\begin{aligned}
\text{Total outputs} &= (M-1) + (M-2) + \dots + (M-(M-1)) = \\
&= M \sum_{n=1}^{M-1} 1 - \sum_{m=1}^{M-1} m = \\
&= M(M-1) - (M-1)(M-1+1)/2 = \\
&= \frac{M(M-1)}{2},
\end{aligned} \tag{163}$$

which is the same as the number of unique transmitting-frequency pairs that is given, for the same set of M frequencies, as

$$\begin{aligned}
\text{Total pairs} &= \binom{M}{2} = \frac{M!}{(M-2)!2!} = \\
&= \frac{M(M-1)}{2}.
\end{aligned} \tag{164}$$

Note, however, that the reference signals for the " f_1 " adaptive array processors will each also contain all the reference signals for the " f_2 " and all the other sets of processors, which will not correlate with the " f_1 " input signals. These will contribute directly to the *mmse* of each " f_1 " processor, and hence also to its output *emse*. The other sets of processors corresponding to the common carriers of other pairs will contain fewer such uncorrelated "reference-contaminating" signals, so the worst case occurs for the " f_1 " common-carrier set in this discussion.

Assuming thermal noise and same-pair interference from reusing cells are negligible and/or nulled out, respectively, the *mmse* is primarily determined by the power of the reference-contaminating signals, according to its derivation in Appendix D. If all the mobiles are subjected to a power-control scheme whereby their signals arrive at the base-station antenna array with roughly equal powers, σ_r^2 , then

$$\begin{aligned}
mmse &= \mathcal{E}\{r_k r_k^*\} - S_1^H W_{opt} = \\
&= \sigma_1^2 - \sum_{m=2}^M \sigma_m^2 - S_1^H W_{opt} = \\
&= (M-1) \sigma_r^2,
\end{aligned} \tag{165}$$

since in the absence of the reference-contaminants $mmse$ would be negligible. Then, if the NLMS algorithm in its fastest form was used to form each array-processor output, the $emse$ contained therein would, in the worst case (f_1 reused $M-1$ times), be the same as the $mmse$ (based on (99), according to the discussion preceding (107)), i.e.

$$emse_{NLMS} = (M-1) \sigma_r^2. \tag{166}$$

The corresponding $mmse$ and $emse$ when the FRQN algorithm in its fastest form was used, would be given by (107) in which $\tau_2 = 4(M-1)$ and $q = 2$ should be used, resulting in

$$emse_{FRQN} = \frac{(M-1) \sigma_r^2}{1 + 2(4(M-1))/2} = \frac{\sigma_r^2}{4}. \tag{167}$$

That is clearly a superior result for any M . (Note that convergence time would increase with M for both algorithms, since β is inversely proportional to M (being inversely proportional to the trace of the $(M-1) \times (M-1)$ matrix R) and τ_2 in FRQN is directly proportional to M . Therefore, for comparable speed (more consistently fast in the case of FRQN), the FRQN algorithm would give $4(M-1)$ times less output error power than the NLMS algorithm in this application.

For acceptable mobile-telephone voice quality the output C/I ratio should be 18 dB [48]. According to (167), the TRAAAP base-station adaptive array processors would provide

$$(C/I) = 10 \log_{10} \left(\frac{\sigma_1^2}{emse_{FRQN}} \right) = 10 \log_{10} \left(\frac{\sigma_1^2}{\sigma_r^2/4} \right) \approx 6 \text{ dB} \quad (168)$$

so the FRQN algorithm would have to be slowed down by a factor of about 16 if no other signal-processing was used to enhance the final base-station receive C/I ratio. (The NLMS algorithm would have to be further slowed down by a factor of $4(M-1)$ to yield comparable performance.) The FRQN time-constant for AMPS would be $\tau_2 = 16 \times 4(20 - 1) = 1216$ and $\Delta t \leq 1/(2 \times 25 \text{ kHz})$, leading to a convergence time in the order of 50 ms.

For the reverse links (base-station to mobiles, or downlinks), it would be impossible to use an adaptive array at each mobile receiver. Not only is its size too small to accommodate one, but it would not be able to spatially separate its desired signal from all the other co-channel signals because they are all transmitted from the same point in space, i.e. the base-station transmitting antenna array. In order to achieve true spatial-division multiple-access (SDMA) bidirectional communications links, the base-station transmitting antenna would have to transmit on orthogonal beams to the co-channel mobile receivers. That is, each beam should have nulls in the directions of all co-channel mobiles except the intended mobile receiver, where it should have peak gain. Note that only *one* frequency per mobile is used for transmitting from the base-station.

One could use the same antenna array and complex-conjugates of the same weights as were formed by adaptation on the receive signal from the same mobiles. However, because the transmitting frequencies would have to be different from the receiving ones to prevent blocking the base-station receivers, the orthogonal transmit beams would not be as orthogonal as their corresponding receive beams. For new wireless mobile networks such as PCS (Personal Communications Services) the *same*

frequency could be used for transmitting in a Time-Division Duplex (TDD) arrangement. After receiving mobile signals and adapting the antenna patterns to them, the same frequencies and (conjugated) weights would be used in the next time-slot to transmit to all of them. Each time-slot would have to be long enough to allow for the weights to converge, yet short enough for the fading environment to appear stationary. With cellular mobiles that would be about 5 ms, but for hand-held PCS type of mobiles, it could easily be 50 ms. The time-slots should be separated by quiet times which would be the difference in propagation time between the nearest and farthest mobiles, typically 5 to 10 μs . Adaptation would take place while the receive data are collected; the weights would then be applied to the collected data and the result passed on to receivers at the end of each receive slot. The same weights would then also be applied to transmit data on the corresponding frequencies in the following transmit slot. This is similar to the TDD arrangement described in a different scheme proposed in [37].

Compared to Space-Time-Frequency-Division Multiple-Access (STFDMA) scheme of [37], TRAAAP has the same efficiency relative to AMPS (conventional cellular scheme), as shown below:

TRAAAP (a transmitted reference scheme):

- * $N = 32$ channels, each about 30 kHz wide (similar to AMPS, any modulation)
- * $N = 32$ array elements $\rightarrow N(N-1)/2 = 15.5 \times$ AMPS (combinatoric advantage)
- * TDD up/down links $\rightarrow \times 2$ (no Tx band required, assuming 2:1 channel coding)
- * $K = 7$ to $K = 3$ $\rightarrow \times 2.3333$ (more efficient frequency re-use)
- TOTAL capacity increase $\approx$ AMPS $\times 72.333$, with 32 elements & front ends.**

STFDMA (an output-derived reference scheme) [37] (approximate figures):

- Vocoded speech @ 8 kb/s
- Time-duplexing requires doubled Tx rate of 16 kb/s
- 100% excess bandwidth BPSK with Nyquist-shaped pulses --> $F_b = 32$ kHz channels
- Frequency-separation of $F_s = 1$ kHz (10 x max. Doppler, also for convergence, cycle resolution in adaptive algorithm)
- * $L = F_b/F_s$ -----> 31 x AMPS (tighter channel spacing)
- * TDD up/down links -----> x 2 (no Tx band required)
- * $K = 7$ to $K = 3$ -----> x 2.3333 (more efficient spatial frequency re-use)
- * $N = 2 \times L + 1$ -----> 63 array elements to resolve overlapping channels
- TOTAL capacity increase = AMPS x 144.666, with 63 elements & front ends.**

For comparison, TDMA (IS-54) and CDMA (IS-95) have 5 x and 20 x capacity increases over AMPS, respectively, while STFDMA actually has 128 times AMPS capacity [37]. TRAAAP is relatively immune to Doppler shifts which significantly affect STFDMA, but requires mobiles to transmit on 2 carriers whereas STFDMA mobiles require only one carrier each. However, TRAAAP is not limited to BPSK or other conjugate self-coherent modulation; it will work with any modulation.

In retrospect, using a replica of the SOI at a different carrier frequency as a transmitted (external) reference signal is made possible by the application, where the goal is primarily to increase spectrum capacity rather than to extract signals from noise or hostile or even intelligent, intentional interference. Therefore the reference signal need not be secure and sophisticated, which only complicates its acquisition. The spatial nulls need not be exceedingly deep either, because the co-channel interference generated in the network is not as powerful as hostile interference encountered in military

communications. In these respects the TRAAP adaptive arrays differ from the original designs which were used to protect the security and integrity of military radio communications against intentional, powerful and even intelligent, enemy "jamming" signals.

In any case, this chapter served to show that the newly-developed FRQN algorithm can be used with any type of reference signal which can be used with LMS type algorithms, and remains stable when used with a generic reference-generating feedback loop for blind adaptation.

Chapter 5

Summary and Conclusions:

In the foregoing, an improved adaptive algorithm for general array signal-processing has been presented. Its place in the field of research has been defined in terms of general applications and history of adaptive algorithms, current research trends and applications and motivations for its development.

Evolution of this FRQN algorithm has been illustrated with the mathematical developments of related algorithms and of its predecessors. Specifically, the original LMS algorithm was developed in a tutorial manner, illustrating the features which prompted the search for faster algorithms such as RLS (the essentials of which were also developed in the same section). The particular problem of the large disparity among convergence times of the orthogonal modes of the LMS algorithm was elucidated with references to practical examples described in the literature. Orders of computational complexity of alternative algorithms were summarized to point the way toward an order-of-N algorithm having "order-of-RLS" convergence properties as the goal of this work. Against this background and a critical review of recent attempts in related directions found in the literature, the novel FRQN algorithm was developed and characterized.

To recapitulate the novel aspects of this work, recall that presently no such Fast RLS (order-of-N) algorithm exists for general adaptive-array signal-processing. A stable one has recently been developed for regular tapped-delay-line (FIR) filters and requires about $8N$ or $9N$ complex operations per iteration [6]. The "ordinary" order-of- N^2 RLS algorithm does work for general array data but is computationally more costly.

On the other hand, the FRQN algorithm potentially has a unique performance advantage over existing algorithms, especially in time-critical, digital adaptive arrays, i.e. spatial filtering in a dynamic environment. That is because it is unconditionally stable with arbitrary array configurations and has order-of N computational complexity, specifically $6N + 2$ complex multiplications, $7N + 2$ additions/subtractions and 2 divisions per iteration. Compared to the NLMS algorithm (which requires about $3N$ complex multiplications and additions per iteration), it has faster convergence for the same excess mean square error because, as the geometric interpretation showed, it follows the quasi-Newton path along the performance surface from early stages of the adaptation interval. It is largely free of the modal convergence-rate disparities which slow down the NLMS algorithm. FRQN is novel and unique in possessing *all* of these properties. It is of second order in the sense of its state-variable recursion, whereas LMS and RLS are of first order. It converges more slowly than RLS but more quickly than LMS; in these respects it is not unique.

Perhaps its most distinguishing feature which sets it apart from competing algorithms is its ability to converge along the same Quasi-Newton path as the RLS algorithm (albeit more slowly) with only order-of- N complexity and without relying on any special type of data structure for its speed or stability. Because the FRQN algorithm optimizes a different penalty-function (i.e. the mean squared error) than the RLS algorithm (which optimizes the squared error at each iteration), it takes longer to compute the mean, hence to converge. It shares that limitation with the NLMS algorithm, but converges faster with less misadjustment error, as was shown analytically and by simulations. Also shown in both ways was the property that the FRQN tracking error was greater than that of NLMS at fast fading rates.

In more particular terms, the novelty of the FRQN algorithm is periodically reflected in its historical development from an analogue algorithm for adaptive arrays advanced by Compton [4] and implemented in modified form by the present author [5]. Initially it was cast into discrete-time form by the present author by converting its differential equation into a finite-difference equation, then rearranging it into an explicitly recursive and unconditionally stable form while retaining its order-of-N simplicity. That was one unique aspect of the present work.

In this conversion from the continuous-time analogue form to discrete-time digital form, unconditional stability with respect to step-size and eigenvalues of the array covariance matrix was retained, but the original loop-type architecture became non-computable. It was later shown that a similar development of the discrete-time LMS algorithm leads to the (discrete-time) Normalized LMS (NLMS) algorithm which then also loses causality in the loop-type form. That is of negligible practical consequence in digital (software) implementations; causal recursive equations were derived for computing both the NLMS and FRQN algorithms.

As a final stitch for tying together the main mathematical developments in this work, an alternative relationship among the various algorithms considered is developed now, with the benefit of hindsight. To that end, the relevant algorithms are rewritten here in their more-or-less canonical, (but non-computable) forms. For FRQN algorithm, it is

$$\tau_1 \tau_2 \Delta^2 \mathbf{W}_k + \tau_2 \left[\frac{1}{\tau_2} \mathbf{I} + \frac{g}{q} \mathbf{Y}_k \mathbf{Y}_k^H \right] \Delta \mathbf{W}_k = -g \mathbf{Y}_k (\mathbf{Y}_k^H \mathbf{W}_k - r_k^*). \quad (169)$$

For the "ordinary second-order LMS" (denoted by "2LMS") algorithm, it is

$$\tau_1 \tau_2 \Delta^2 \mathbf{W}_k + \tau_2 \Delta \mathbf{W}_k = -g \mathbf{Y}_k (\mathbf{Y}_k^H \mathbf{W}_k - r_k^*). \quad (170)$$

Both of the above are of second order in the recursion of the state variables, or weights W_k , and order-of-N in computational complexity. The next two algorithms are of first order. The Normalized LMS (NLMS) is order-of-N computationally and has the form

$$\tau \Delta W_k = -g Y_k (Y_k^H W_k - r_k^*), \quad (171)$$

while the order-of-N² RLS algorithm has the Quasi-Newton (QN) form

$$R_k \Delta W_k = -Y_k (Y_k^H W_{k-1} - r_k^*). \quad (172)$$

Notice that the RHS of (172) differs from that of equations (169) to (171) in that the index of W in (172) is $k-1$, whereas in the others it is k . To facilitate this development, this difference is eliminated from (172) as

$$\begin{aligned} R_k \Delta W_k &= -Y_k (Y_k^H W_k - r_k^*) - Y_k Y_k^H W_{k-1} + Y_k Y_k^H W_k \\ &= -Y_k (Y_k^H W_k - r_k^*) + [Y_k Y_k^H] \Delta W_k, \end{aligned} \quad (173)$$

where $\Delta W_k = W_k - W_{k-1}$ was used. The recursive relation $R_k = Y_k Y_k^H + \lambda R_{k-1}$ is then applied to finish it as

$$\begin{aligned} [R_k - Y_k Y_k^H] \Delta W_k &= -Y_k (Y_k^H W_k - r_k^*) \\ \lambda R_{k-1} \Delta W_k &= -Y_k (Y_k^H W_k - r_k^*) \end{aligned} \quad (174)$$

The approach now is to evolve the FRQN algorithm out of the RLS algorithm with the help of the others and some approximations to make the result a QN algorithm with order-of-N complexity. To that end, it is useful to compute the difference of the two equations (174) and (173) as

$$[\lambda R_{k-1} - R_k] \Delta W_k = -Y_k Y_k^H \Delta W_k \quad (175)$$

and using the notation $\Delta R_k = R_k - R_{k-1}$, to rewrite it as

$$-\Delta R_k \Delta W_k = -Y_k Y_k^H \Delta W_k + (1 - \lambda) R_{k-1} \Delta W_k. \quad (176)$$

It is instructive to view the LHS as part of the finite-difference approximation of the derivative of a product, namely

$$\Delta(R_k \Delta W_k) = \Delta R_k \Delta W_k + R_k \Delta^2 W_k. \quad (177)$$

By substitution, equation (176) after some rearrangement becomes

$$-\Delta(R_k \Delta W_k) + R_k \Delta^2 W_k + Y_k Y_k^H \Delta W_k = (1 - \lambda) R_{k-1} \Delta W_k. \quad (178)$$

After substituting from the LHS of equation (174) in the RHS of equation (178) and further rearrangement, it becomes

$$R_k \Delta^2 W_k + Y_k Y_k^H \Delta W_k = -\left(\frac{1 - \lambda}{\lambda}\right) Y_k (Y_k^H W_k - r_k^*) + \Delta(R_k \Delta W_k). \quad (179)$$

Equation (179) is still the same RLS algorithm but in a form very similar to that of the 2LMS algorithm. Since the latter is computationally much simpler, it may serve as a guide for some approximations which will simplify (179) to order-of-N complexity, yet retain its QN convergence properties. The QN property is easily recognized in these forms as the matrix coefficient of ΔW_k which best approximates matrix R_k (i.e. in (174) or in (172)). Fortunately, equation (179) does contain something like it and it is even in a form with order-of-N complexity, i.e. $Y_k Y_k^H \Delta W_k$. The order-of-N² complexity arises in (179) due to the second-order difference term $R_k \Delta^2 W_k$, so it is desirable to approximate it by some scalar multiple of the corresponding term from the 2LMS algorithm, equation (170), and to get rid of the last term in (179) as well.

That is vaguely reminiscent of the approach by the author of [36] who approximated the matrix coefficient of the *first-order* weight-difference in the RLS algorithm by its diagonal entries only, in order to get order-of-N computational complexity. By doing it only on the *second-order* weight-difference term here, it is expected to have less effect on the convergence behaviour of the resulting approximation of the RLS algorithm.

In any case, adding a small diagonal perturbation to the second-order term in (179) will not make a big difference, because the result $[(\tau_1/g)I + R_k]\Delta^2 W_k$ is the same as that of adding uncorrelated rms noise voltage $\sqrt{\tau_1/g}$, $g > 1$ to the data vector Y . Suppose then that one keeps only the diagonal perturbation second-order term in (179). (As will be shown later with reference to the geometric interpretation of the second-order form of FRQN, any changes to the second-order difference in the weights affect only the *embedded* algorithm which adjusts the *changes* in the weight steps toward the QN direction. In this case, the embedded algorithm is still stable.)

Furthermore, notice that the last term in (179) is really a first-order difference of equation (172). Except for the index of W , its RHS looks a lot like the second-last term in the RHS of (179). Recall also that $\lambda \approx 1$, $\lambda < 1$. Since it originates from the same second-order difference as the first term on the RHS of (179), which was scaled down by a factor of τ_1/g , one should also scale down this approximation similarly. The choice of this scalar, with some hindsight, is $1/g$. Furthermore, it should represent a stable recursion, so the sign on the RHS of (172) should be changed when substituting it. Therefore the last term in (179) may be approximated as

$$\begin{aligned}
(1/g)\Delta(R_k\Delta W_k) &= \frac{+1}{g}\left(\frac{1-\lambda}{\lambda}\right)Y_k(Y_k^H W_{k-1} - r_k^*) = \\
&= \frac{+1}{g}\left(\frac{1-\lambda}{\lambda}\right)Y_k(Y_k^H W_k - r_k^*) + \\
&\quad + \frac{1}{g}\left(\frac{1-\lambda}{\lambda}\right)Y_k(Y_k^H W_{k-1} - Y_k^H W_k)
\end{aligned} \tag{180}$$

The resulting QN approximation of (179) then becomes, after some rearrangement,

$$\left(\frac{\tau_1}{g}\right)\Delta^2 W_k + \left[\frac{1-\lambda}{g\lambda}I + Y_k Y_k^H\right]\Delta W_k = -(1-1/g)\left(\frac{1-\lambda}{\lambda}\right)Y_k(Y_k^H W_k - r_k^*) \tag{181}$$

which can be moulded into a more familiar form, again with some hindsight. Letting

$\lambda/(1-\lambda) = \tau_2$ represent the dominant equivalent exponential time-constant of the RLS algorithm, $1-1/g \approx 1$ and multiplying through by g and τ_2 yields the FRQN algorithm without the speed-up factor q , as

$$\tau_1 \tau_2 \Delta^2 W_k + \tau_2 \left[\frac{1}{\tau_2} I + g Y_k Y_k^H \right] \Delta W_k = -g Y_k (Y_k^H W_k - r_k^*) . \tag{182}$$

Notice that it gives the same value of $\gamma = 1/\tau_2$ as was arrived at through investigations of the leakage term.

As for the geometric interpretation of (179) along the same lines of manipulation as in (81) - (83), the corresponding embedded algorithm for adapting the weight-steps is

$$\overline{R}_k \Delta \left(\frac{\lambda}{1-\lambda} \Delta W_k \right) = -R \left[\left(\frac{\lambda}{1-\lambda} \right) \Delta W_k - (R^{-1} \nabla_w |e_k|^2) \right] + \Delta(\overline{R}_k \Delta W_k) \tag{183}$$

with $R = \overline{R}_k = \overline{Y_k Y_k^H}$ assumed in the averaging. Recall that the existence of the coefficient matrix R premultiplying the RHS of (83) was not essential, but rather a result of the FRQN algorithm having to adapt the weights themselves correctly. In (183) this coefficient cancels on premultiplication by R^{-1} , which makes the last term on its RHS

$$R^{-1}[\Delta(\overline{R_k} \Delta W_k)] = \Delta^2 W_k - R^{-1} \overline{R_{k-1}} \Delta W_k - \Delta W_k \quad (184)$$

which can be deduced from (177) by premultiplying it with R_k^{-1} and averaging. When the RHS of (184) is approximated as $\Delta^2 W_k$ and substituted into (183), the result is

$$\Delta\left(\left(\frac{\lambda}{1-\lambda} - 1\right) \Delta W_k\right) = -\left[\left(\frac{\lambda}{1-\lambda}\right) \Delta W_k - (R^{-1} \nabla_w |e_k|^2)\right] \quad (185)$$

which can be further simplified with some hindsight, to give

$$2\Delta(\tau_2 \Delta W_k) = -\left[\tau_2 \Delta W_k - (R^{-1} \nabla_w |e_k|^2)\right] \quad (186)$$

The factor of 2 on its LHS suggests the empirically-obtained optimal speed-up factor of $q = 2$, but it is not present in its LHS to make it so. Nevertheless, (186) also represents a stable algorithm for adapting the weight-steps in the QN direction, but along a different path and at different speeds than the corresponding algorithm embedded in FRQN. Since it is a second-order phenomenon, the difference between RLS and FRQN is not catastrophic. However, considering the multitude of other possible assumptions that one could make in pursuing this approach, it is fitting to label it as a "Queasy-Newton" approach for its lack of assurance of stability of the results (without the benefit of hindsight). For such reasons, the more formal approach of transforming a known stable algorithm (originated by Compton) from continuous into discrete-time was developed and pursued in this work. As was shown in this thesis, a more general way of converting analogue continuous-time algorithms into discrete-time digital ones, is to average them (with respect to the data), take their Laplace transforms and apply one of a number of stability-preserving s-to-z transformations, transform the resulting z-domain equations into discrete-time difference equations, then remove the averaging on the (now discrete-time) data. If current data is presumed in the last step, the order of computations should be rearranged from its original servo-loop form so as to be causal. This generalization is a new aspect which was explored in the present work.

Another novel aspect of this work is that the present form of the FRQN algorithm actually contains an *embedded*, stable, fast and robust algorithm which adapts the *actual steps* of the main algorithm toward the Quasi-Newton direction, whereas competing algorithms often employ a *separate*, more ad-hoc type of algorithm to adapt only their steps' *size-determining parameter(s)*.

When it was first converted into discrete-time, the algorithm from [5] was found to possess severe misadjustment when its speed was increased. An important aspect of this work is the modification and optimization of the FRQN algorithm and its variants so as to simultaneously reduce this misadjustment and convergence time to the point where they are superior to those of the fastest form of NLMS and approach those of RLS, in all types of scenarios, without sacrificing any of its other virtues. As illustrated by the developments and simulations presented in the body of the thesis, some success in this direction has been achieved by judiciously manipulating the two time-constants, "leakage" terms and "loop" gains.

In Chapter 4 it was shown that the FRQN algorithm is compatible with all the forms of reference-signal-generating techniques that have been applied to LMS algorithms. Specifically, it remains stable when the same generic kind of reference-generating non-linear feedback loop is added at its output as has been used with LMS algorithms for blind adaptation (e.g. constant-modulus algorithm) applications.

The hope is that a useful approximation of the FRLS algorithm for arrays has been developed. Even if its speed is not as high as that of RLS, it is a useful contribution for array processing because this FRQN algorithm is inherently robust, stable and of low complexity, unlike existing RLS (order-of- N^2) adaptive-array algorithms.

For future work, it is suggested that a detailed characterization of the numerical properties of the FRQN algorithm be undertaken, specifically to determine its numerical stability with fixed-point arithmetic.

Its computational savings and convergence properties could be verified by using it to adapt large arrays, where it is expected to have the most advantages relative to existing algorithms.

Its relationship with the RLS algorithm derived in this chapter with the help of the second-order LMS algorithm could also be investigated more rigorously, with a view to formulating a version of the FRQN algorithm which more closely approximates RLS using higher-order time-differences of the weights, yet retains order-of-N simplicity and stability.

Other non-FIR applications of adaptive algorithms could be investigated with the FRQN algorithm to determine if it has an advantage over existing algorithms in those applications, as well as to perhaps formulate other versions of it in such applications.

Another avenue of further work with the FRQN algorithm would be to implement it in a real-time application, perhaps with the help of Appendix H.

6 References:

- [1] B. Widrow and S. Stearns: Adaptive Signal Processing, Englewood Cliffs, NJ, Prentice-Hall 1985.
- [2] P. D. Ritger and N. J. Rose: Differential Equations with Applications, McGraw-Hill Book Company, Toronto, 1968.
- [3] B. Widrow, J. M. McCool, M. G. Larimore and R. J. Johnson Jr.: "Stationary and Nonstationary Learning Characteristics of the LMS Adaptive Filter", *Proceedings of the IEEE*, Vol. 64, No. 8, August, 1976.
- [4] R. T. Compton, Jr.: "Improved Feedback Loop for Adaptive Arrays", *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-16, No. 2, March 1980.
- [5] M. Klemes: "A Practical Method of Obtaining Constant Convergence Rates in LMS Adaptive Arrays", *IEEE Transactions on Antennas and Propagation*, vol. AP-34, No. 3, March 1986.
- [6] D. T. M. Slock and T. Kailath: "Numerically Stable Fast Transversal Filters for Recursive Least Squares Filtering", *IEEE Transactions on Signal Processing*, vol. 39, No. 1, January 1991.
- [7] C. C. Ko, F. Chin and S. S. Foo: "An Adaptive Algorithm for Separating and Tracking Multiple Directional Sources in Linear Arrays", *IEEE Transactions on Antennas and Propagation*, vol 40, No. 3, March 1992.
- [8] C. C. Ko and R. Bachl: "Adaptive Zero-Tracking Complemented by LMS Weight Update for Guaranteed Convergence", *IEEE Transactions on Antennas and Propagation*, vol. 40, No. 3, March 1992.

- [9] R. Khanna and B. B. Madan: "Adaptive Beamforming Using a Cascade Configuration", *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-31, No. 4, August 1983.
- [10] P.-R. Chang, W.-H. Yang and K. K. Chan: "A Neural Network Approach to MVDR Beamforming Problem", *IEEE Transactions on Antennas and Propagation*, vol. 40, No. 3, March 1992.
- [11] R. H. Kwong and E. W. Johnston: "A Variable Step Size LMS Algorithm", *IEEE Transactions on Signal Processing*, vol. 40, No. 7, July 1992.
- [12] D. F. Marshall and W. K. Jenkins: "A Fast Quasi-Newton Adaptive Filtering Algorithm", *IEEE Transactions on Signal Processing*, vol. 40, No. 7, July 1992.
- [13] J. H. Winters: "Signal Acquisition and Tracking with Adaptive Arrays in Wireless Systems", *Proceedings of the 1993 IEEE Vehicular Technology Conference*, pp 85-88, Secaucus NJ, 18-20 May 1993.
- [14] J. H. Winters: "Spread-Spectrum in a Four-Phase Communication System Employing Adaptive Antennas", *IEEE Transactions on Communications*, vol. COM-30, No. 5, May 1982.
- [15] S. Makino, Y. Kaneda and N. Koizumi: "Exponentially-Weighted Stepsize NLMS Adaptive Filter Based on the Statistics of Room Impulse Response", *IEEE Transactions on Speech and Audio Processing*, vol. 1, No. 1, January 1993.
- [16] T. J. Shan and T. Kailath: "Adaptive Algorithms with an Automatic Gain Control Feature", *IEEE Transactions on Circuits and Systems*, vol. CAS-35, No. 1, January 1988.
- [17] S. Karni and G. Zeng: "Comments on "Adaptive Algorithms with an Automatic Gain Control Feature" ", *IEEE Transactions on Circuits and Systems*, vol. CAS-37, No. 7, pp. 974-975, July 1990.

- [18] T. J. Shan and T. Kailath: "Comments on "Comments on "Adaptive Algorithms with an Automatic Gain-Control Feature" " " *IEEE Transactions on Circuits and Systems*, vol. CAS-37, No. 7, p. 975, July 1990.
- [19] R. W. Harris, D. M. Chabries and F. A. Bishop: "A Variable Step (VS) Adaptive Filter Algorithm", *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-34, No. 2, April 1986.
- [20] D. F. Marshall, W. K. Jenkins and J. J. Murphy: "The use of Orthogonal Transforms for Improving Performance of Adaptive Filters", *IEEE Transactions on Circuits and Systems*, vol. CAS-36, No. 4, April 1989.
- [21] R. D. Gitlin and F. R. Magee, Jr.: "Self-Orthogonalizing Adaptive Equalization Algorithm", *IEEE Transactions on Communications*, vol. COM -25, No.7, July 1977.
- [22] F. F. Kretschmer, Jr. and B. L. Lewis: "An Improved Algorithm for Adaptive Processing" *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-14, No. 1, January 1978.
- [23] C. R. Johnson, Jr.: "An Implementation of the Multidimensional Modified LMS Algorithm", *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-16, No. 3, May 1980.
- [24] R. Nitzberg: "A Generalization of the Normalized LMS Algorithm", *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-23, No. 4, July 1987.
- [25] K. M. Tao: "Statistical Averaging and PARTAN - Some Alternatives to LMS and RLS", *Proceedings of the 1992 IEEE International Conference on Acoustics Speech and Signal Processing, ICASSP'92*, pp. IV-25 - IV-28.

- [26] L. E. Brennan, E. L. Pugh and I. S. Reed: "Control-Loop Noise in Adaptive Array Antennas", *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-7, No. 2, March 1971.
- [27] O. Macchi and M. Jaidane-Saidane: "Stability of Adaptive Recursive Filters", *Proceedings of the 1988 International Conference on Acoustics Speech and Signal Processing, ICASSP'88*, pp. 1503-1505.
- [28] S. S. Oren: "Quasi-Newton Algorithms: Approaches and Motivations", *Proc. 1973 IEEE Conference on Decision and Control, CDC*, paper N. TP2-4, pp. 422-427, 1973.
- [29] V. J. Matthews and Z. Xie: "A Stochastic Gradient Adaptive Filter With Gradient Adaptive Step Size", *IEEE Transactions on Signal Processing*, vol. 41, No. 6, June 1993.
- [30] E. Eleftheriou and D. D. Falconer: "Tracking Properties and Steady-State Performance of RLS Adaptive Filter Algorithms" *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-34, No. 5, October 1986.
- [31] S. Haykin: Adaptive Filter Theory (second edition), Prentice-Hall, Englewood Cliffs New Jersey 07632, 1991.
- [32] C. E. Davila: "Line Search Algorithms for Adaptive Filtering", *IEEE Transactions on Signal Processing*, vol. 41, No. 7, July 1993.
- [33] J. B. Evans, P. Xue and B. Liu: "Analysis and Implementation of Variable Step Size Adaptive Algorithms", *IEEE Transactions on Signal Processing*, vol. 41, No. 8, August 1993.
- [34] J. E. Hudson: Adaptive Array Principles Peter Peregrinus Ltd., U. K. 1981.

- [35] D. T. M. Slock: "On The Convergence Behaviour of the LMS and the Normalized LMS Algorithms", *IEEE Transactions on Signal Processing*, vol. 41, No.9, Sept. 1993.
- [36] Z. Raida: "Steering an Adaptive Antenna Array by the Simplified Kalman Filter", *IEEE Transactions on Antennas and Propagation*, vol. 43, No. 6, June 1995.
- [37] W. A. Gardner, S. V. Schell and P. A. Murphy: "Multiplication of Cellular Radio Capacity by Blind Adaptive Spatial Filtering", in *Proc. of Int'l Conf. on Selected Topics in Wireless Comm.*, pp. 102-106, Vancouver, B.C., Canada, June 1992.
- [38] D. D. Falconer, F. Adachi and B. Gudmundson: "Time Division Multiple Access Methods for Wireless Personal Communications", *IEEE Communications Magazine*, January 1995.
- [39] G. Kubin: "Stabilization of the RLS Algorithm in the Absence of Persistent Excitation", *IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP'88*, Vol. III, pp.1369-1372, 1988.
- [40] D. O. Reudink: "Properties of Mobile Radio Propagation Above 400 MHz", *IEEE Transactions on Vehicular Technology*, vol. VT-23, November 1974.
- [41] J. Salz and J. H. Winters: "Effect of Fading Correlation on Adaptive Arrays in Digital Mobile Radio", *IEEE Transactions on Vehicular Technology*, vol. 43, No. 4, November 1994.
- [42] R. T. Compton, Jr.: Adaptive Antennas - Concepts and Performance, Prentice Hall, Englewood Cliffs, New Jersey 07632, 1988.
- [43] B. G. Agee, S. V. Schell and W. A. Gardner: "Spectral Self-Coherence Restoral: A New Approach to Blind Adaptive Signal Extraction Using Antenna Arrays", *Proceedings of the IEEE*, Vol. 78, No. 4, April 1990.

- [44] R. He and J. H. Reed: "Spectral Correlation of AMPS Signals and its Application to Interference Rejection", *Proc. IEEE Conference on Military Communications MILCOM'94*, Boston MA, October 1994.
- [45] Y. Chen, C. L. Nikias, J. G. Proakis: "Adaptive Blind Equalization", *CRASP Report # 101*, Center for Research on Applied Signal Processing, University of Southern California, Los Angeles, CA 90089, 1992.
- [46] S. Bellini: "Busgang Techniques for Blind Equalization", *Proc. of IEEE GLOBECOM Conference*, pp. 1634-1640, 1986.
- [47] P. M. Hansen: "Application of Adaptive Array Technology to HF Communications Systems", *Proc. of IEEE EUROCON Conference*, pp. 2.3.8.1-2.3.8.9, 1977.
- [48] W. C. Y. Lee: Mobile Cellular Telecommunications Systems, McGraw-Hill Book Company, Toronto, 1989.
- [49] J. Cui, D. D. Falconer and A. U. H. Sheikh: "A Phase Algorithm for Blind Adaptive Diversity Combining", *Proc. IEEE Vehicular Technology Conf., VTC'95*, Chicago IL, July 1995.

Appendix A: S to Z Transformations

Reasons for Using Them:

Although it is quite straight-forward to create a discrete-time version of a continuous-time adaptive algorithm by approximating the continuous-time differential equation by a finite-difference equation as

$$\frac{dW(t)}{dt} \approx \frac{W(kt) - W((k-1)t)}{kt - (k-1)t} = \frac{\Delta W_k}{\Delta t} \quad \dots(A1)$$

where Δt is the sampling interval of discrete time, it is instructive to take a more formal and general approach. That is especially true in the case of *stochastic* algorithms, which are at the same time subjected also to *quantization* of their states and input data, as both processes introduce errors into the resulting algorithm. The formalism of the s-to-z transformation or mapping offers a means to determine stability and robustness of a larger variety of resulting discrete-time, digital algorithms, since the process of converting from continuous-time analogue to discrete-time, digital form is not unique in practice.

Sampling: Effect on Laplace Transform (S-Domain) and Reformulation in Z-Domain [A1]

A continuous-time function $W(t)$ is impulse-sampled at intervals Δt of discrete time according to

$$\{W_k\} = W(t) \sum_{k=0}^{\infty} \delta(t - k\Delta t) \quad \dots(A2)$$

which for all practical purposes may be written as

$$\{W_k\} = \sum_{k=0}^{\infty} W(k\Delta t) \delta(t-k\Delta t) . \quad \dots(A3)$$

State-variables of most linear systems can be expressed as superpositions of complex-exponential basis functions, as for example

$$W(t) = \sum_{n=1}^N b_n e^{s_n t} = \sum_{n=1}^N f_n(t) \quad \dots(A4)$$

where $\{b_n\}$ are termed *residues* and $\{s_n\}$ are complex frequencies. Its Laplace transform is given by

$$\mathcal{L}[W(t)] = W(s) = \sum_{n=1}^N \frac{b_n}{s - s_n} = \frac{B(s)}{\prod_{n=1}^N (s - s_n)} \quad \dots(A5)$$

where $\{s_n\}$ are also the *poles* of $W(s)$. Clearly, when the real part of a pole is > 0 , its contribution in (A4) diverges with time, causing $W(t)$ to be unstable.

Without loss of generality, it suffices to focus on one general basis function in (A4) and examine its sampled form resulting from application of (A3), which with the help of the sum of a geometric series yields

$$\{f_{n_k}(s)\} = b_n \sum_{k=0}^{\infty} e^{s_n k \Delta t} e^{-s k \Delta t} = \frac{b_n}{1 - e^{-(s - s_n) \Delta t}} \quad \dots(A6)$$

Now the poles of $\mathcal{L}[\{W_k\}]$ will be entirely different, since even its n -th component (or *mode*) given by (A6) has infinitely many poles, determined from

$$e^{-(s-s_n)\Delta t} = 1 \quad \text{or} \quad s = s_n \pm \frac{j2\pi}{\Delta t}, \quad p = 0, 1, 2, \dots$$

That is, the poles of each mode repeat in periodic strips parallel to the σ -axis in the $s = \sigma + j\omega$ plane, at intervals of $\pm 2\pi/\Delta t$ along the $j\omega$ -axis.

It is awkward to work with this infinity of poles for every mode. Re-naming the transform variable according to

$$z = e^{s\Delta t} \quad \dots(\text{A7})$$

considerably simplifies the complex-frequency transform of (A6) of the discrete-time (sampled) mode, so that it becomes

$$\{f_n(z)\} = \frac{b_n}{1 - z^{-1}z_n} \quad \dots(\text{A8})$$

with $z_n = e^{s_n\Delta t}$. In the z -plane, effectively only one pole occurs, at $z = z_n$. According to (A6), the geometric series converges as long as $|z_n| < 1$, i.e. the poles remain within the unit circle in the z -plane.

Equation (A7) in effect simplifies (A6) by mapping the repeating periodic strips of the s -plane to separate circular surfaces superimposed on each other in the z -plane, so they appear as one common pole-zero configuration. Consequently, to maintain uniqueness and reversibility of this s -to- z transformation, all poles in the s -domain prior to sampling must lie within one strip, so each repetition of them is mapped to one surface in the z -plane. Since all real signals and systems are characterized by real functions, their transforms must possess poles which are either purely real or occur in complex-conjugate pairs. The highest angular frequency $\omega_{\max}$ of the latter kind must be within the borders

of one periodic strip, so the poles, being separated symmetrically about the real axis, are located at

$$s_p = \sigma_p + j\omega_p = \sigma_p \pm j\omega_{\max} \pm \frac{j2p\pi}{\Delta t}$$

with one period or strip width given by

$$\omega_{\max} = \frac{\pi}{\Delta t}$$

and $p = 0, 1, 2, \dots$. The z -plane surfaces, each containing the image of one strip in the s -plane, are joined at the negative real axes, which represent the images of the borders of those strips, as seen by writing

$$z = u + jv = e^{\sigma \Delta t} (\cos(\omega_p \Delta t) + j\sin(\omega_p \Delta t)) = e^{-\sigma \Delta t} = Re[z] \quad \forall p. \quad \dots (A9)$$

The s -plane strips repeat every $2\pi/\Delta t$ radians/second and the z -plane surfaces repeat every 2π radians, like circles. The left half of the s -plane maps into the unit circle centred at the origin in the z -plane. If all the frequencies of the transformed sampled function are less than the maximum determined by the sampling rate, i.e. $\omega_{\max}$, then it is sufficient to deal only with the $p = 0$ surface in the z -plane, all others being redundant. Then the s -plane function can be uniquely obtained from the z -plane function, meaning that the continuous-time function can be uniquely and completely reconstructed from the sequence of samples forming the discrete-time function. The condition to complete this sampling theorem may be deduced from the preamble to (A9) as

$$2\pi f < \omega_{\max} = \frac{\pi}{\Delta t}$$

or equivalently, as

$$\boxed{2f < \frac{1}{\Delta t}} \quad \dots (A10)$$

Figure A1 illustrates a general s-to-z mapping according to (A7).

Figure A1: Sketch of the s-to-z transformation given by (A7).

VZEXPSHTHS

Rational Approximations:

It is often inconvenient to work with transcendental functions when dealing with poles and zeros, so one is led to search for rational functions which adequately approximate the transcendental s-to-z transformation function (A7). With high enough sampling frequencies, Δt can always be made sufficiently small such that these approximations are very close to the true "z" in some given range of frequencies around zero. For example, one may approximate (A7) as

$$z^{-1} = e^{-s\Delta t} \approx 1 - s\Delta t, \quad \dots(A11)$$

thus effectively defining the s-to-z transformation

$$s = \frac{1 - z^{-1}}{\Delta t}. \quad \dots(A12)$$

This is termed the "backward difference" transformation, because it maps the Laplace transform of a time-derivative (represented by s) to the Z transform of a backward difference (represented by $1 - z^{-1}$, where z^{-1} is recognized as the Z transform of a unit time-sample delay Δt). The geometry of this mapping is sketched in Figure A2 below. Note that there are no strips in the s -plane, since one is not actually sampling, but rather approximating a continuous-time system by a related synthetic, discrete-time system. In this approximation, the left half of the s -plane is mapped to a circle of radius $1/2$, centred at $z = 1/2 + j0$. Since this falls entirely within the unit circle $|z| = 1$, stability is preserved by this mapping.

Figure A2: Sketch of the backward-difference s-to-z mapping.

VZBDTSTHS

Another approximation is obtained by truncating the Taylor series expansions

$$e^{s\Delta t} = 1 + s\Delta t + \frac{1}{2}(s\Delta t)^2 + \dots \quad \dots(A13)$$

and

$$e^{-s\Delta t} = 1 - s\Delta t + \frac{1}{2}(s\Delta t)^2 + \dots \quad \dots(\text{A14})$$

to their linear terms and taking the quotient on both sides, i.e.

$$z^2 = e^{2s\Delta t} \approx \frac{1 + s\Delta t}{1 - s\Delta t} . \quad \dots(\text{A15})$$

Then by analogy, one may also write the approximation of z itself as

$$z = e^{s\Delta t/2} \approx \frac{1 + s\Delta t/2}{1 - s\Delta t/2} , \quad \dots(\text{A16})$$

from which s can be isolated in terms of z as

$$s = \frac{2}{\Delta t} \left(\frac{1 - z^{-1}}{1 + z^{-1}} \right) . \quad \dots(\text{A17})$$

This is the *bilinear* or Smith-chart s -to- z transformation; it maps the left half of the s -plane to the unit circle in (one surface of) the z -plane, so it also preserves stability. Its geometry is sketched in Figure A3 below. Note that in Figures A2 and A3, the regions of low frequencies (s close to $0 + j0$, z close to $1 + j0$) show little distortion due to the mappings, implying that the discrete-time systems behave much like the sampled continuous-time system at sufficiently low frequencies, or equivalently, with sufficiently high sampling rates, as expected by intuition.

Figure A3: Sketch of the bilinear s-to-z mapping

VZBLTSTHS

A More General, Stability-preserving S-to-Z Transformation:

Other s-to-z transformations are of course also possible, the most suitable ones depending upon the application. In this application one is not concerned with preserving the scale of a particular kind of frequency-response, so one need not pre-distort the original system as when one is synthesizing discrete-time filters from continuous-time designs. The primary concerns here are preservation of stability and steady states, while maintaining short transient response-times of all the modes. The first requirement is satisfied by constraining the s-to-z transformation to map the left half of the s-plane inside the unit circle in the z-plane. The second concern is satisfied by requiring that the same transformation also map the zero-frequency (i.e. steady state) image to its *exact* image in the z-plane according to (A7).

Mathematically, this transformation may be developed by generalizing the bilinear and backward-difference transformations so that they map the imaginary-s-axis onto a circle with a general radius equal to ρ , then shifting the circle so that the point $s = 0 + j0$ always maps exactly to the point $z = 1 + j0$. The latter requirement agrees with the final-value theorems in the s- and z-domains, which must give the same steady states.

To begin, postulate a generalization of the backward-difference transformation such that, with some hindsight,

$$s = \frac{z - 2\rho}{z \Delta t}, \quad \dots(A18)$$

and let $s = \sigma + j\omega$ and $z = u + jv$. The image of the $j\omega$ -axis of the s-plane in the z-plane is found by expanding s and z, then equating corresponding real and imaginary parts while $\sigma = 0$ (i.e. the $j\omega$ -axis). Proceeding,

$$\sigma + j\omega = \frac{u + jv - 2\rho}{(u + jv) \Delta t}. \quad \dots(A19)$$

Multiplying the RHS by $1 = (u - jv)/(u - jv)$ and separating into real and imaginary parts yields

$$\sigma + j\omega = \frac{u(u - 2\rho) + v^2}{\Delta t(u^2 + v^2)} + j \frac{2v\rho}{\Delta t(u^2 + v^2)}. \quad \dots(A20)$$

Since $\sigma = 0$ on the $j\omega$ -axis, the corresponding real part above must also = 0, i.e.

$$u^2 - 2u\rho + v^2 = 0 \quad \dots(A21)$$

which looks more familiar after adding ρ^2 to both sides and rearranging to produce

$$(u - \rho)^2 + v^2 = \rho^2 \quad \dots(A22)$$

Clearly, this is a circle with radius ρ centred at $u = \rho$. However, the point $s = 0 + j0$ does not map to $z = 1 + j0$ according to (A20), which implies that $v = 0$ and it instead maps, according to (A21), to

$$u(u - 2\rho) = 0 \quad \dots(A23)$$

or since $u \neq 0$, to $u = 2\rho$. That means the circle (A22) must be shifted by the remaining distance $1 - 2\rho$ to $u = 1$. Substituting $u - (1 - 2\rho)$ into (A22) for u gives the equation for the required circle as

$$(u - 1 + \rho)^2 + v^2 = \rho^2 \quad \dots(A24)$$

It is easily verified that the same substitution in (A19) gives the required generalization of the backward-difference s-to-z transformation as

$$s = \frac{z - (1 - 2\rho) - 2\rho}{(z - (1 - 2\rho))\Delta t} = \frac{z - 1}{(z - 1 + 2\rho)\Delta t} \quad \dots(A25)$$

A similar procedure can be applied to the bilinear s-to-z transformation starting with

$$s = \frac{2}{\Delta t} \left(\frac{z - \rho}{z + \rho} \right) \quad \dots(A26)$$

This maps $\sigma = 0$ to the circle given by

$$u^2 + v^2 = \rho^2 \quad \dots(A27)$$

which has radius ρ and is centred at $u = 0, v = 0$. This also does not map $s = 0 + j0$ to $z = 1 + j0$. It is easy to see by expanding (A26) and equating real and imaginary parts separately, that $s = 0 + j0$ is in fact mapped to $z = \rho + j0$. Again, this is $(1 - \rho)$ away from $u = 1$ along the real z -axis, so $u - (1 - \rho)$ is substituted for u in (A26) to complete the required generalization. This yields the same general circle as before, namely

$$(u - 1 + \rho)^2 + v^2 = \rho^2 \quad \dots(\text{A28})$$

and the required generalization of the bilinear transformation then becomes

$$s = \frac{2}{\Delta t} \left(\frac{z - (1 - \rho) - \rho}{z - (1 - \rho) + \rho} \right) = \frac{2}{\Delta t} \left(\frac{z - 1}{z - 1 + 2\rho} \right) \quad \dots(\text{A29})$$

It differs from the generalized backward-difference transformation (A25) by a factor of 2 on the RHS. When $\rho = 1$, (A29) gives the original bilinear transformation. When $\rho = 1/2$, (A25) gives the original backward-difference transformation. Equations (A25) and (A29) may be recognized as the same equation, where ρ himself is an additional factor which has been disguised as a number, i.e.

$$s = \frac{2\rho}{\Delta t} \left(\frac{z - 1}{z - 1 + 2\rho} \right) \quad \dots(\text{A30})$$

It is easily verified that $\rho = 1$ yields (A17) and $\rho = 1/2$ yields (A12). It is required that $0 < \rho < 1$ to preserve stability. A sketch of this s -to- z transformation is shown in Figure A4 below.

Figure A4: Sketch of the generalized, stability-preserving s-to-z mapping.

VZGRTSTHS

[A1] P. F. Blackman: Introduction to State Variable Analysis (appendix 3), MacMillan Press Ltd., New York, 1977.

Appendix B: Convergence in the Mean

Independence and Other Assumptions:

Because the coefficients in W vary with time much more slowly than the signals in Y , these two processes may be regarded as independent random processes, even though W is clearly a function of (the covariance of) Y in a loose sense. From the point of view of the time-scale of W , while W remains quasi-stationary, Y undergoes many fluctuations unrelated to the major, slower trends visible in W during the same time. Since Y is assumed to consist of zero-mean signals, then this independence assumption implies that

$$\mathcal{E}\{W^H Y\} = 0 \quad \dots(B1)$$

In adaptive filter theory it is also frequently assumed that successive signal samples are independent, although in an FIR adaptive filter they are clearly very much dependent, since only the first entry in the signal vector is replaced by a new one and the last one is discarded, and the remaining $N-1$ of them are simply shifted by one position [31]. In adaptive arrays the signal vector Y_k does not possess such a shifting structure, so it is more valid to assume that Y_k and Y_{k-1} are independent.

By the nature of the adaptation criterion and the gradient estimate, adaptation ceases when the error signal which drives the algorithm is no longer correlated with the input signal, i.e.

$$\mathcal{E}\{Y e_{opt}^*\} = 0 \quad \dots(B2)$$

which means the mean square error has been minimized to $mmse = \mathcal{E}(e_{opt} e_{opt}^*)$ and steady state of W has been reached. This is the well-known *orthogonality condition*.

It is also assumed that r_k does not depend on Y_p except when $p=k$. All processes are also assumed to be jointly Gaussian, with the mean of W being W_{opt} in stationary scenarios, the covariance matrix of Y being

$$R = \mathcal{E}\{YY^H\} \quad \dots(B3)$$

and the covariance vector of Y and r being the steering vector

$$S = \mathcal{E}\{Yr^*\} \quad \dots(B4)$$

in general. The signals are also assumed to be ergodic. In case the step-size parameter is time-varying, it is also assumed to be independent of the other processes.

Conditions for Stability and Steady State – NLMS Algorithm:

Convergence in the mean implies stability of the coefficients in the mean sense, with the above assumptions in mind. After an initial transient period, they should reach an average steady state W_{opt} . To determine what that is and the necessary conditions, one averages the algorithm equations (55) and (56)

$$W_k - W_{k-1} = -\beta_k Y_k [Y_k^H W_{k-1} - r_k^*] \quad \dots(B5)$$

with $\beta_k = \mu / (1 + \mu Y_k^H Y_k)$, $\mu = g/\tau = g \Delta t / T > 1$.

Assuming now that W_k and W_{k-1} represent the averaged quantities, it becomes

$$W_k - W_{k-1} = -\beta R W_{k-1} + \beta S \quad \dots(B6)$$

with $\beta = \mu / (1 + \mu \sum \lambda_n)$, where $\{\lambda_n\}$ are the eigenvalues of R whose sum equals $Tr[R]$. The resulting difference equation is diagonalized using the

similarity $R = E \Lambda E^{-1}$. Since R is positive-definite and Hermitian, $E^{-1} = E^H$ i.e. the columns of E are orthonormal eigenvectors of R and the diagonal entries of Λ are the corresponding eigenvalues [34] (which may always be argued to be distinct in practice). Letting $V = E^H W$ and $B = E^H S$, the n -th diagonalized mode of (B6) becomes

$$v_{n,k} - v_{n-1,k} = -\beta \lambda_n v_{n,k-1} + \beta b_n \quad \dots(B7)$$

whose z -domain transfer function is then easily determined as

$$\frac{v_n(z)}{b_n(z)} = \frac{\beta}{1 - z^{-1}(1 - \beta \lambda_n)} \quad \dots(B8)$$

Its only pole is at $z_n = 1 - \beta \lambda_n$, which must lie inside the unit circle $|z| = 1$ for stability. Consequently, the condition for stability and hence convergence in the mean for the NLMS algorithm is that, for all n ,

$$-1 < (1 - \beta \lambda_n) < 1 \quad \dots(B9)$$

or, since $-\beta \lambda_n < 0$ always, one requires that in the extreme case, $-1 < 1 - \beta \lambda_{\max}$ or equivalently,

$$\boxed{\beta < \frac{2}{\lambda_{\max}}} \quad \dots(B10)$$

With the average value of β_n this condition is always met, because

$$\frac{\mu}{1 + \mu \sum_{n=1}^N \lambda_n} < \frac{1}{\sum_{n=1}^N \lambda_n} < \frac{1}{\lambda_{\max}} < \frac{2}{\lambda_{\max}} \quad \dots(B11)$$

so the NLMS algorithm is always stable in the mean.

NLMS Time-Constants:

Recalling that (B5) corresponded to a backward-difference s-to-z transformation of the continuous-time LMS algorithm whose open-loop time-constant was T and loop gain was $g\lambda_n$, the equivalent closed-loop time-constants for the modes of this discrete-time NLMS algorithm can be related to those of the continuous-time algorithm. Knowing the solution for the modes of the averaged original differential equation, rewritten as

$$T \frac{d}{dt} v_n(t) = -g\lambda_n v_n(t) + g b_n(t) \quad \dots(B12)$$

with $b_n(t)$ regarded as a step-function with Laplace transform b_n/s , the continuous-time-constants may be identified from the solution

$$\begin{aligned} v_n(t) &= \frac{g}{T} \int_0^t e^{-g\lambda_n \xi/T} b_n(\xi) d\xi + v_n(0) e^{-g\lambda_n t/T} \\ &= \frac{-1}{\lambda_n} (e^{-g\lambda_n t/T} - 1) b_n + v_n(0) e^{-g\lambda_n t/T} \end{aligned}$$

or the form

$$v_n(t) - v_n(0) = \left(\frac{b_n}{\lambda_n} - v_n(0) \right) (1 - e^{-g\lambda_n t/T}) \quad \dots(B13)$$

They are identified as the closed-loop modal time-constants

$$T_n = \frac{T}{g\lambda_n}, \quad n=1, \dots, N \quad \dots(B14)$$

which are the negative reciprocals of the s-plane poles. When they are normalized by the sampling interval used in the discrete-time version and β is taken as in (B6), they become

$$\tau_n = \frac{\tau}{g \lambda_n} = \frac{1}{\mu \lambda_n} = \frac{1 + \mu \text{Tr}[R]}{1 - z_n} . \quad \dots(\text{B15})$$

Digression:

When $g \rightarrow \infty$, the LMS algorithm takes on its maximum speed and adapts "instantaneously", owing to the nature of the gradient estimate as noted in the body of the thesis. It also adapts in the average sense, as described by the average of the final form of (B6), whose continuous-time analogy is

$$T \frac{d}{dt} W(t) = -\beta R W(t) + \beta S(t) . \quad \dots(\text{B16})$$

In this situation (as $g \rightarrow \infty$) the reference to time in (B12) or (B6) is lost, so one may obtain the effective time-constants of the discrete-time NLMS algorithm from the modes of (B16) by analogy with (B12). The corresponding n-th mode differential equation is

$$T \frac{d}{dt} v_n(t) = -\beta \lambda_n v_n(t) + \beta b_n(t) \quad \dots(\text{B17})$$

from which the time-constants are obtained similarly as (B15), i.e.

$$\tau_n = \frac{\tau}{\beta \lambda_n} . \quad \dots(\text{B18})$$

These are related to the z-plane poles of (B8) as

$$\tau_n = \frac{\tau}{1 - z_n} . \quad \dots(\text{B19})$$

In (B16) T is the same as in the original continuous-time LMS algorithm (46) and β is the averaged value, i.e.

$$\beta = \frac{\mu}{1 + \mu \text{Tr}[R]} \quad \dots(\text{B20})$$

with $\mu = g \Delta t / T$. As the original LMS algorithm is sped up toward "instantaneous" speed by letting $g \rightarrow \infty$, one obtains $\beta = 1 / \text{Tr}[R]$ in the NLMS discrete-time algorithm, so its time-constants become

$$\tau_n \rightarrow \frac{\tau \text{Tr}[R]}{\lambda_n}.$$

When the original LMS algorithm is "normalized" by making $g = \tau / \text{Tr}[R]$, (B15) gives the corresponding discrete-time NLMS algorithm's time constants, with the help of (B20), as

$$\tau_n = \frac{\tau}{\frac{1}{2 \text{Tr}[R]} \lambda_n} = \frac{\tau (2 \text{Tr}[R])}{\lambda_n},$$

which explains why the time to converge in the mean is halved when g is changed from $g = \tau / \text{Tr}[R]$ to $g \rightarrow \infty$ in the NLMS algorithm (B5) in simulations in the body of the thesis. This diversion shows that continuous-time formulations of adaptive algorithms can be useful for physical interpretation of their behaviour, as in the above case with the change in "g" resulting in the loss of time-related quantities in (B6).

Time-constants are helpful in determining the time an algorithm takes to converge to steady state in a stationary scenario. It is generally accepted for first-order-dominated systems, that convergence to steady state is complete after $5T_{\max}$, i.e. 5 times the longest modal time-constant. That is equal to $5\tau_{\max} \Delta t$ seconds of real time, or $5\tau_{\max}$ iterations.

NLMS Steady States:

In any case, providing that the stability conditions for convergence in the mean are met, in the steady state one must have (in the averaged sense), $W_k - W_{k-1} = Q$, or $dW/dt = Q$, so according to (B6) $RW_{\text{steady-state}} = S$. If the inverse of R exists (if not, the pseudo-inverse can be used), then the steady-state coefficient vector is the optimum solution

$$W_{\text{opt}} = R^{-1} S . \quad \dots(B21)$$

If the scenario is reasonably stationary, the corresponding optimal output signal is

$$y_{\text{opt},k} = W_{\text{opt}}^H Y_k \quad \dots(B22)$$

and the minimum error by $y_{\text{opt},k} - r_k$. Since W_{opt} is assumed not to change, this is also the steady-state a-priori error which causally drives the algorithm. (Note that W_{opt} is the *average* steady-state coefficient vector; the *instantaneous* one causes the instantaneous error to follow r_k as $g \rightarrow \infty$, but the a-priori error still behaves much like the average error, to the extent that the instantaneous deviation of W_k is not the same as that of W_{k-1} , only their *averages*, W_{opt} , are equal). Hence one may also write

$$\alpha_{\text{opt},k} = e_{\text{opt},k} = W_{\text{opt}}^H Y_k - r_k . \quad \dots(B23)$$

FRQN Algorithm:

It is easier to determine the stability conditions for convergence in the mean of the FRQN algorithm when it is written in a form which does not yet contain quotients of stochastic quantities, i.e.

$$[a_0 I + b_0 Y_k Y_k^H] W_k = [a_1 I + b_1 Y_k Y_k^H] W_{k-1} - \tau_1 \tau_2 W_{k-2} + g Y_k r_k^* \quad \dots (B24)$$

where

$$\begin{aligned} a_0 &= \tau_2(\tau_1 + \gamma), & b_0 &= g(1 + \tau_2/q), \\ a_1 &= 2\tau_2(\tau_1 + \gamma/2), & b_1 &= g\tau_2/q, \end{aligned}$$

and $\tau_2 \approx 4N$, $\tau_1 = \tau_2/4$, $\gamma = 1/\tau_2$, $q = 2$ and $g > 1$. (The effect of $g \rightarrow \infty$ will again be explored in the averaged form at the end of this treatment). Here the normalized time-constants are related to their analogous continuous-time counterparts as $\tau_{1,2} = T_{1,2}/\Delta t$ with Δt again being the Nyquist sampling interval and T_1, T_2 are determined with the help of signal bandwidth considerations discussed with respect to equation (75) in the body of the thesis.

Invoking (B1) - (B4) to obtain the averaged form of (B24) produces

$$[a_0 I + b_0 R] W_k = [a_1 I + b_1 R] W_{k-1} - \tau_1 \tau_2 W_{k-2} + g S \quad \dots (B25)$$

which is subsequently diagonalized as was (B6). The n -th mode of that diagonalized equation then obeys the second-order scalar difference equation

$$(a_0 + b_0 \lambda_n) v_{n,k} = (a_1 + b_1 \lambda_n) v_{n,k-1} - \tau_1 \tau_2 v_{n,k-2} + g b_{n,k} \quad \dots (B26)$$

with the notation being the same as in (B7). By regarding $b_n(z) = Z[b_{n,k}]$ as the forcing function again, the z -domain transfer function is easily found to be

$$\frac{v_n(z)}{b_n(z)} = \frac{gz^2}{(a_0 + b_0 \lambda_n)z^2 - (a_1 + b_1 \lambda_n)z + \tau_1 \tau_2} \quad \dots (B27)$$

It remains to find its poles and corresponding residues, and time-constants. The poles are found from the solution of the quadratic equation

$$(a_0 + b_0 \lambda_n) z^2 - (a_1 + b_1 \lambda_n) z + \tau_1 \tau_2 = 0, \quad \dots(B28)$$

which is

$$z_n = \frac{(a_1 + b_1 \lambda_n) \left(1 \pm \sqrt{1 - \frac{4 \tau_1 \tau_2 (a_0 + b_0 \lambda_n)}{(a_1 + b_1 \lambda_n)^2}} \right)}{2(a_0 + b_0 \lambda_n)}. \quad \dots(B29)$$

Recalling that $g > 1$ (and eventually $g \rightarrow \infty$) and assuming that none of the eigenvalues is ever exactly zero (because in reality there is always a finite amount of noise power), allows one to use the approximation that $\sqrt{1 - \epsilon} \approx 1 - \epsilon/2$ for $\epsilon < 1$, thus simplifying the solution to that of

$$z_n = \frac{a_1 + b_1 \lambda_n}{2(a_0 + b_0 \lambda_n)} \left[1 \pm \left(1 - \frac{2 \tau_1 \tau_2 (a_0 + b_0 \lambda_n)}{(a_1 + b_1 \lambda_n)^2} \right) \right]. \quad \dots(B30)$$

It has the two solutions

$$z_{1,n} = \frac{\tau_1 \tau_2}{a_1 + b_1 \lambda_n} = \frac{\tau_1 \tau_2}{2 \tau_1 \tau_2 + \gamma \tau_2 + g \tau_2 \lambda_n / q} \quad \dots(B31)$$

and

$$z_{2,n} = \frac{a_1 + b_1 \lambda_n}{a_0 + b_0 \lambda_n} - z_{1,n} = \frac{2 \tau_1 \tau_2 + \gamma \tau_2 + g \tau_2 \lambda_n / q}{\tau_1 \tau_2 + \gamma \tau_2 + g(1 + \tau_2 / q) \lambda_n} - z_{1,n}. \quad \dots(B32)$$

Both poles must be within the unit circle for stability, with any positive value of λ_n . Note that both $z_{1,n}$ and $z_{2,n}$ are always ≥ 0 . For stable convergence in the mean, one therefore requires only that $z_{1,n}$ satisfies

$$\tau_1 \tau_2 < 2\tau_1 \tau_2 + \gamma \tau_2 + g \tau_2 \lambda_n / q \quad \dots(B33)$$

and $z_{2,n}$ satisfies

$$-1 < \frac{a_1 + b_1 \lambda_n}{a_0 + b_0 \lambda_n} - z_{1,n} < 1 \quad \dots(B34)$$

Inequality (B33) is always satisfied since all quantities in it are > 0 . Inequality (B34) will also be seen to be always satisfied, via the continuous-time analogy of equation (63), i.e.

$$\begin{aligned} T_1 T_2 \frac{d^2}{dt^2} W(t) + T_2 \left[\gamma I + \frac{g}{q} Y(t) Y^H(t) \right] \frac{d}{dt} W(t) = \\ = -g Y(t) Y^H(t) W(t) + g Y(t) r^*(t) \end{aligned} \quad \dots(B35)$$

By the same process of averaging and diagonalization, followed by a Laplace transformation into s-domain, the n-th mode is found to have the second-order transfer function

$$\frac{v_n(s)}{b_n(s)} = \frac{g}{s^2 T_1 T_2 + s T_2 (\gamma + g \lambda_n / q) + g \lambda_n} \quad \dots(B36)$$

whose s-plane poles, with the same approximation in the square roots, are given by the solutions

$$s_n = -\frac{(\gamma + g \lambda_n / q)}{2 T_1} \left[1 \pm \left(1 - \frac{2 g \lambda_n T_1}{T_2 (\gamma + g \lambda_n / q)^2} \right) \right] \quad \dots(B37)$$

They are

$$s_{1,n} = - \left(\frac{\gamma + g\lambda_n/q}{T_1} \right) + \frac{g\lambda_n}{T_2(\gamma + g\lambda_n/q)} \quad \dots(B38)$$

and

$$s_{2,n} = - \left(\frac{g\lambda_n}{\gamma + g\lambda_n/q} \right) \frac{1}{T_2} . \quad \dots(B39)$$

The continuous-time system and, by virtue of the backward-difference s-to-z transformation (see Appendix A), the discrete-time system (B24), are stable when $s_{1,n}$ and $s_{2,n}$ are in the LHP, i.e. $\text{Re}[s_{1,n} \text{ and } s_{2,n}] < 0$ for any $\lambda_n \geq 0$. By inspection, that is always true for the pole at $s_{2,n}$. The pole at $s_{1,n}$ must satisfy the equivalent condition that

$$\left(\frac{T_1}{T_2} \right) \frac{g\lambda_n}{\gamma + g\lambda_n/q} < \gamma + g\lambda_n/q . \quad \dots(B40)$$

With $g > 1/\lambda_{\min}$ and $T_2/T_1 = 4$, it is always satisfied, for in the worst case (lowest $g\lambda_n$), it is required to satisfy $1 \leq 4(\gamma + 1/2)^2$ which is always true because $\gamma > 0$. Therefore it is concluded that the FRQN algorithm is always stable in the mean.

FRQN Time-Constants:

Because of the generally simpler form of s-domain poles and transfer functions, it is again instructive to derive the time-constants of the discrete-time FRQN algorithm from the corresponding continuous-time "prototype" algorithm. In the latter they are related to the poles as in the continuous-time NLMS algorithm. When $g \rightarrow \infty$ in the final form, they simplify considerably, but in general, the n-th mode continuous time-

constants are

$$T_{1,n} = \frac{-1}{s_{1,n}} \quad \dots(\text{B41})$$

and

$$T_{2,n} = \frac{-1}{s_{2,n}} \quad \dots(\text{B42})$$

Each mode has two first-order components whose exponential time-constants are (B41) and (B42) and amplitudes are proportional to the corresponding residues multiplied by the time-constants [5]. When $g \rightarrow \infty$ the time-constants tend to

$$T_{1,n} \rightarrow 0 \quad \dots(\text{B43})$$

and

$$T_{2,n} \rightarrow \frac{T_2}{q} \quad \dots(\text{B44})$$

The corresponding case in the final form of the discrete-time FRQN algorithm is that the poles tend to

$$z_{1,n} \rightarrow 0 \quad \dots(\text{B45})$$

and

$$z_{2,n} \rightarrow \frac{\tau_2/q}{1 + \tau_2/q} \quad \dots(\text{B46})$$

If the modal time-constants of the discrete-time algorithm are again defined as the normalized modal time-constants of the backward-difference continuous-time counterpart, then they are given by

$$\tau_n = \tau_{2,n} = \frac{T_{2,n}}{\Delta t} = \frac{\tau_2}{q} = \frac{z_{2,n}}{1 - z_{2,n}} \quad \dots(\text{B47})$$

and 0. (The non-dominant component of each mode approaches zero amplitude and zero time-constant as $g \rightarrow \infty$).

These modal time-constants of the FRQN algorithm play similar roles in the two components of each mode as do the continuous time-constants in their respective components. Since the residues in the s-domain and z-domain are the same (and approximately the same in the backward-difference approximation of z), then according to Appendix A, this correspondence is expected. The corresponding residues for the FRQN algorithm are found from the partial-fraction expansion of its transfer function according to (A8), which is

$$\frac{v_n(z)}{b_n(z)} = gz \left(\frac{Az}{z - z_{1,n}} + \frac{Bz}{z - z_{2,n}} \right). \quad \dots(B48)$$

In (B48), after factoring out z, one must obtain $A(z - z_{2,n}) + B(z - z_{1,n}) = 1 + 0z$, which leads to the two equations

$$A = -B \quad \dots(B49)$$

and

$$Az_{2,n} + Bz_{1,n} = -1. \quad \dots(B50)$$

With $g \rightarrow \infty$ one obtains $A = -1/z_{2,n} = -B$. Recalling that the geometric series a^k has Z-transform $z/(z - a)$, one finds that each component of (B48) corresponds to one such series. The component with $a = z_{1,n}$ decays very quickly (instantly) to zero, and its starting value $a = z_{1,n}$ also $\rightarrow 0$ as $g \rightarrow \infty$. The component with $a = z_{2,n}$ settles more slowly to its non-zero steady state and has a much larger starting value (i.e. $z_{2,n}$, close to 1), so it clearly dominates the n-th mode's response. Since τ_2/q is constant for all n, the FRQN algorithm is also seen to possess quasi-Newton convergence behaviour in discrete-time.

FRQN Steady States:

Since the FRQN algorithm was determined to always converge stably in the mean, its state variables reach their steady states, in the mean, when $W_k = W_{k-1} = W_{k-2}$. It is readily apparent from equations (81) or (84) in the body of the thesis, that these are the same optimal steady states reached by the NLMS algorithm, namely

$$W_{opt} = R^{-1} S \text{ .} \quad \dots(B51)$$

with the corresponding available output signal at time k given by

$$y_{opt,k} = W_{opt}^H Y_k \text{ .} \quad \dots(B52)$$

Appendix C: Derivation of the FRQN Algorithm with More General S-to-Z Transformation

In this appendix the most general version of the FRQN algorithm considered in this thesis, equation (95), is derived using the method outlined in section 2.2 for the NLMS algorithm. As the initial steps starting from the continuous-time differential equation preceding equation (76) are straightforward, this treatment begins in the Laplace-transform domain with

$$T_1 T_2 s^2 W(s) + s T_2 [\gamma I + g_d R] W(s) + g R W(s) = g S \quad \dots(C1)$$

which also incorporates the speed-up refinement equation (90) and the signal-statistics as defined in (B3) and (B4). It is a simple matter now to replace s with equation (94) which is derived in detail in Appendix A. After multiplying both sides by $(z - 1 + 2\rho)^2$ the result is

$$4\rho^2 \tau_1 \tau_2 (z^2 - 2z + 1) W(z) + 2\rho \tau_2 (z - 1)(z - 1 + 2\rho) [\gamma I + g_d R] W(z) + (z - 1 + 2\rho)^2 [g R] W(z) = g (z - 1 + 2\rho)^2 S \quad \dots(C2)$$

where $\tau_{1,2} = T_{1,2}/\Delta t$ are the continuous-time time-constants normalized by the Nyquist sampling interval. Notice that when $\rho \neq 1/2$ and both sides are multiplied by z^{-2} to make the system causal, the RHS ends up with a z -dependent transfer-function. However, recall that S , like R , is treated as a constant quantity at this stage, so the filtering of S should be evaluated at zero frequency or steady state, when $z = 1$. Consequently the development of the general FRQN algorithm may be significantly simplified by approximating this filtering of S on the RHS by its steady state, i.e. replacing that filter transfer-function with its value when $z = 1$. After doing that and grouping all the like

subscripts of W , the result in discrete-time domain is

$$\begin{aligned}
 & [I(4\rho^2\tau_1\tau_2 + 2\rho\gamma\tau_2) + (g + 2\rho\tau_2g_d)R]W_k + \\
 & + [I(-8\rho^2\tau_1\tau_2 - 2(1-\rho)2\rho\gamma\tau_2) + R(-4\rho\tau_2g_d(1-\rho) - 2g(1-2\rho))]W_{k-1} + \\
 & + [I(4\rho^2\tau_1\tau_2 + 2\rho\gamma\tau_2(1-2\rho)) + R(2\rho\tau_2g_d(1-2\rho) + g(2\rho-1)^2)]W_{k-2} = \\
 & = 4\rho^2gS \quad \dots(C3)
 \end{aligned}$$

When the signal-statistics are "de-averaged" to just the current samples and the parameters are defined as

$$\begin{aligned}
 a_0 &= 2\rho\tau_2(\gamma + 2\rho\tau_1), & b_0 &= 2\rho\tau_2g_d + g \\
 a_1 &= 4\rho\tau_2(2\rho\tau_1 + \gamma(1-\rho)), & b_1 &= 4\rho\tau_2g_d(1-\rho) + 2g(1-2\rho) \\
 a_2 &= 2\rho\tau_2(\gamma(1-2\rho) + 2\rho\tau_1), & b_2 &= (1-2\rho)(2\rho\tau_2g_d + g(1-2\rho))
 \end{aligned}$$

with $\tau_2 = 4N$, $\tau_1 = \tau_2/4$, $\gamma = 1/\tau_2$, $g = 1/\lambda_{\min}$, $g_d = g/q$ (then $q = 2$, $g \rightarrow \infty$), the result simplifies to

$$\begin{aligned}
 [a_0I + b_0Y_kY_k^H]W_k &= [a_1I + b_1Y_kY_k^H]W_{k-1} - \\
 & - [a_2I + b_2Y_kY_k^H]W_{k-2} + \\
 & + 4\rho^2gY_kr_k^* \quad \dots(C4)
 \end{aligned}$$

It remains to isolate W_k by premultiplying both sides of (C4) by

$$[a_0I + b_0Y_kY_k^H]^{-1} = \frac{1}{a_0} \left[I - \frac{Y_kY_k^H}{a_0/b_0 + Y_k^HY_k} \right] \quad \dots(C5)$$

which is obtained by applying the matrix-inversion identity derived in Appendix E. When that is done, it yields W_k as

$$\begin{aligned}
W_k = & \frac{1}{a_0} \left[a_1 I + b_1 Y_k Y_k^H - \frac{a_1 Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} - \frac{b_1 Y_k (Y_k^H Y_k) Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] W_{k-1} - \\
& - \frac{1}{a_0} \left[a_2 I + b_2 Y_k Y_k^H - \frac{a_2 Y_k Y_k^H}{a_0/b_0 + Y_k^H Y_k} - \frac{b_2 Y_k (Y_k^H Y_k) Y_k^H}{a_0/b_0 + Y_k^H Y_k} \right] W_{k-2} + \dots (C6) \\
& + \frac{4\rho^2 g}{a_0} \left[Y_k r_k^* - \frac{Y_k (Y_k Y_k^*) r_k^*}{a_0/b_0 + Y_k^H Y_k} \right].
\end{aligned}$$

Recognizing that $Y_k^H Y_k$ and the denominators are scalar factors which can be taken outside of the vector and matrix factors affords further simplification to the final form as

$$\boxed{
\begin{aligned}
W_k = & \left[\left(\frac{a_1}{a_0} \right) I + \left(\frac{b_1/b_0 - a_1/a_0}{a_0/b_0 + Y_k^H Y_k} \right) Y_k Y_k^H \right] W_{k-1} - \\
& - \left[\left(\frac{a_2}{a_0} \right) I + \left(\frac{b_2/b_0 - a_2/a_0}{a_0/b_0 + Y_k^H Y_k} \right) Y_k Y_k^H \right] W_{k-2} + \\
& + \left(\frac{4\rho^2 g}{b_0} \right) \left(\frac{1}{a_0/b_0 + Y_k^H Y_k} \right) Y_k r_k^*.
\end{aligned}
} \quad \dots (C7)$$

As discussed in the body of the thesis, the available output signal is given by $y_k = W_{k-1}^H Y_k$.

Appendix D: Convergence in the Mean Square

In this Appendix, conditions for both the NLMS and the FRQN algorithms to converge in the mean square sense (i.e. toward the minimum mean squared error) are derived for both stationary as well as non-stationary signals. First, the minimum mean squared error itself, $mmse$, is derived, since it is the same for all least-squares algorithms. For the stationary cases, the time-step evolution of the total output mean squared error, mse_k , is derived. It is composed of the $mmse$ plus the excess mean squared error, $emse_k$, which is due to "weight jitter", i.e. the fact that W_k themselves are based on stochastic processes which fluctuate in time. An algorithm-independent expression for the $emse_k$ is also derived in this first section, and expanded for each algorithm in the two subsequent sections. The steady-state misadjustment ratio, $\lim_{k \rightarrow \infty} (emse_k / mmse)$, is derived for both algorithms in the case of stationary signal statistics.

For the case of non-stationary signals, a model for the dynamic weights is defined and used to derive expressions for an additional mean squared error term due to the lagging of W_k behind the non-stationary optimal weights. Steady-state mean squared tracking errors, $tmse$, are derived for the NLMS and FRQN algorithms. Simplified expressions for selected algorithm parameters (comparable maximum speeds) are highlighted and compared. Only the steady-state tracking error expressions are derived, and the applicability of the model of non-stationarity to adaptive arrays is briefly discussed.

Stationary Signals:

The observation that the *emse* of the FRQN algorithm is much smaller than that of the NLMS algorithm was also made by the author of [D1], who investigated their analogue continuous-time counterparts. The approach for deriving the various components of the output mean squared error used herein is not the same as in [D1], but rather follows that used in [31].

The independence assumptions introduced in Appendix B will be invoked to derive the expressions for the minimum and the excess mean squared errors in the outputs of least-squares types of adaptive algorithms. The former is defined as

$$mmse \triangleq \mathcal{E}\{\alpha_{opt,k} \alpha_{opt,k}^*\} \quad \dots(D1)$$

where $\alpha_{opt,k}$ is the available (or driving) output error signal when the weights have reached their optimal steady states given by (B21) (assuming they converged and don't jitter, $\alpha_{opt,k} = e_{opt,k}$, as in (B23)). When these weights are substituted in the expression for the output signal at time k , the output error relative to the reference signal at the same time is

$$\alpha_{opt,k} = W_{opt}^H Y_k - r_k. \quad \dots(D2)$$

The definition (D1) can then be expanded as

$$\begin{aligned} mmse &= \mathcal{E}\{(W_{opt}^H Y_k - r_k)(Y_k^H W_{opt} - r_k^*)\} = \\ &= \mathcal{E}\{W_{opt}^H Y_k Y_k^H W_{opt}\} - 2 \operatorname{Re}[\mathcal{E}\{W_{opt}^H Y_k r_k^*\}] + \mathcal{E}\{r_k r_k^*\}. \end{aligned} \quad \dots(D3)$$

Realizing that W_{opt} is a constant vector (given by (B21) as $W_{opt} = R^{-1}S$), the expectations are applied only to the time-varying (k -dependent) quantities. With the help

of (B3) and (B4), they simplify (D3) as

$$mmse = W_{opt}^H R W_{opt} - 2 \operatorname{Re} \{ W_{opt}^H S \} + \sigma_r^2. \quad \dots(D4)$$

Defining $\sigma_r^2 \triangleq \mathcal{E}\{r_k r_k^*\}$ as the power contained in the reference signal and invoking (B21) finally yields, for all k , the constant

$$mmse = \sigma_r^2 - S^H R^{-1} S = \sigma_r^2 - S^H W_{opt}. \quad \dots(D5)$$

In real-time stochastic algorithms W_{opt} is never actually realized as a purely constant vector, because W_k is always computed with incomplete statistics in place of R and S , which do contain errors. Thus (D5) is only a theoretical quantity, as is W_{opt} , to which the mean-square and mean, respectively, convergence properties may be compared. The actual mean squared error at sample time k is given by

$$mse_k \triangleq \mathcal{E}\{\alpha_k \alpha_k^*\} = \mathcal{E}\{|W_{k-1}^H Y_k - r_k + (W_{opt}^H Y_k - W_{opt}^H Y_k)|^2\} \quad \dots(D6)$$

which allows one to write it in terms of the error in the weight vector

$$X_{k-1} \triangleq W_{k-1} - W_{opt} \quad \dots(D7)$$

by regrouping terms and using (D2) to obtain

$$\begin{aligned} mse_k &= \mathcal{E}\{|X_{k-1}^H Y_k + \alpha_{opt,k}|^2\} = \\ &= \mathcal{E}\{X_{k-1}^H Y_k Y_k^H X_{k-1}\} + \\ &\quad + 2 \operatorname{Re} \{ \mathcal{E}\{X_{k-1}^H Y_k \alpha_{opt,k}^*\} \} + \\ &\quad + \mathcal{E}\{\alpha_{opt,k} \alpha_{opt,k}^*\} \end{aligned} \quad \dots(D8)$$

Recall that, by the assumed independence of W_{k-1} and Y_k in (B1), one may simplify the middle term as

$$\mathcal{E}\{X_{k-1}^H Y_k \alpha_{opt,k}^*\} = \mathcal{E}\{X_{k-1}^H\} \mathcal{E}\{Y_k \alpha_{opt,k}^*\} = 0 . \quad \dots(D9)$$

That is because the gradient of the error-power surface is zero in the steady state, as the expectation of its estimate (to which it is *equal*, according to the discussion following equation (14) in the body of the thesis) is also zero. Hence the orthogonality condition (B2) arises, which makes the middle term of (D8) zero, as (D9) demonstrates. Finally, the last term on the RHS of (D8) is recognized as the *mmse* given by (D1), so (D8) simplifies to

$$mse_k = \mathcal{E}\{X_{k-1}^H Y_k Y_k^H X_{k-1}\} + mmse . \quad \dots(D10)$$

The form of (D10) suggests a definition for another mean-squared error contribution in excess of the inevitable minimum mean squared error (which will be algorithm-dependent via the weight-error vector), i.e. the excess mean squared error

$$emse_k \triangleq \mathcal{E}\{X_{k-1}^H Y_k Y_k^H X_{k-1}\} , \quad \dots(D11)$$

so that, for stationary signal statistics, one always has at sample time k the relation

$$mse_k = emse_k + mmse . \quad \dots(D12)$$

But recall that, according to (B1), X_{k-1} and Y_k are assumed to be independent. Therefore they can be averaged separately. That is, the *total* expectation implied in (D11) is, by ergodicity, the expectation over the X_{k-1} and Y_k ensembles. Since these may be performed separately, one may explicitly write the RHS of (D11) as

$$\mathcal{E}\{X_{k-1}^H Y_k Y_k^H X_{k-1}\} = \mathcal{E}_X\{X_{k-1}^H \mathcal{E}_Y\{Y_k Y_k^H\} X_{k-1}\} = \mathcal{E}_X\{X_{k-1}^H R X_{k-1}\} , \quad \dots(D13)$$

where (B3) was invoked for the separate $\mathcal{E}_Y$ expectation. Now observe that X_{k-1}^H and $R X_{k-1}$ are vectors which form a scalar inner product. It is equal to its own *Trace*. Equivalently,

it is also equal to

$$X_{k-1}^H (RX_{k-1}) = \text{Tr}[(RX_{k-1})X_{k-1}^H] \quad \dots(\text{D14})$$

and consequently, because R is now a constant matrix unaffected by $\mathcal{E}_X$, the subscript X will be dropped and (D11) rewritten as

$$emse_k = \mathcal{E}\{\text{Tr}[RX_{k-1}X_{k-1}^H]\} = \text{Tr}[R\mathcal{E}\{X_{k-1}X_{k-1}^H\}] . \quad \dots(\text{D15})$$

Defining the covariance matrix of the weight-error vector as

$$K_{k-1} \triangleq \mathcal{E}\{X_{k-1}X_{k-1}^H\} , \quad \dots(\text{D16})$$

which depends on the particular adaptive algorithm under consideration, leads to a general expression for the stationary excess mean squared output error

$$\boxed{emse_k = \text{Tr}[RK_{k-1}] .} \quad \dots(\text{D17})$$

This expression will next be expanded for the NLMS, and in the succeeding section for the FRQN, algorithms in stationary signal scenarios.

NLMS Algorithm - Transient and Steady-State mse:

According to the preceding treatment based on the independence assumptions and orthogonality condition, the total stationary output mean squared error at time k is given by (D12) in terms of signal-dependent and algorithm-dependent quantities as

$$mse_k = \text{Tr}[RK_{k-1}] + \left(\sigma_r^2 - S^H R^{-1} S \right) . \quad \dots(\text{D18})$$

It is of interest in this section to derive the algorithm-dependent term K_{k-1} in terms of known algorithm parameters and signal statistics. Because W_{k-1} and X_{k-1} are

recursive, K_{k-1} is also recursive. For clarity of notation upon substitution from the algorithm's recursive equations, the expressions for $Tr[RK_k]$ will be derived.

First, the NLMS algorithm will be reformulated in terms of the weight-error vector rather than the weight vector, by adding and subtracting W_{opt} in (55) - (57) and manipulating, as

$$\begin{aligned} (W_k - W_{opt}) &= (W_{k-1} - W_{opt}) - \beta_k Y_k Y_k^H (W_{k-1} - W_{opt}) - \\ &\quad - \beta_k Y_k (Y_k^H W_{opt} - r_k^*) , \end{aligned} \quad \dots(D19)$$

which is recognized with the help of (D7) and (D2) as

$$X_k = [I - \beta_k Y_k Y_k^H] X_{k-1} - \beta_k Y_k \alpha_{opt,k}^* . \quad \dots(D20)$$

Now the definition (D16) is invoked to derive the recursive equation for K_k . To make the stochastic aspects tractable, it will be assumed that $\beta_k = \beta$ is a constant:

$$\begin{aligned} K_k = \mathcal{E}\{X_k X_k^H\} &= \mathcal{E}\{[I - \beta Y_k Y_k^H] X_{k-1} X_{k-1}^H [I - \beta Y_k Y_k^H]\} - \\ &\quad - 2\beta^2 \text{Re}\left[\mathcal{E}\{\alpha_{opt,k}^* Y_k X_{k-1}^H [I - \beta Y_k Y_k^H]\}\right] + \\ &\quad + \beta^2 \mathcal{E}\{\alpha_{opt,k}^* Y_k Y_k^H \alpha_{opt,k}\} . \end{aligned} \quad \dots(D21)$$

It will be necessary to invoke the Gaussian moment-factoring theorem for jointly-Gaussian, zero-mean scalar complex Gaussian random variables [31], which states that

$$\mathcal{E}\{z_1 z_2^* z_3 z_4^*\} = \mathcal{E}\{z_1 z_2^*\} \mathcal{E}\{z_3 z_4^*\} + \mathcal{E}\{z_1 z_4^*\} \mathcal{E}\{z_2 z_3^*\} . \quad \dots(D22)$$

Starting with the last term on the RHS of (D21) and invoking the orthogonality condition yields

$$\beta^2 \mathcal{E}\{\alpha_{opt,k}^* Y_k Y_k^H \alpha_{opt,k}\} = \beta^2 (0 \cdot 0 + R_{mmse}) . \quad \dots(D23)$$

Since $K_k = K_k^H$, the middle term of (D21) must be zero, as it is the only non-Hermitian

term. That may also be verified using the independence assumptions of X_{k-1} and Y_k , leading to

$$\mathcal{E}\{\alpha_{opt,k}^* Y_k X_{k-1}^H - \beta \alpha_{opt,k}^* Y_k X_{k-1}^H Y_k Y_k^H\} = \mathcal{E}_X\{\mathcal{E}_Y\{\alpha_{opt,k}^* Y_k\} X_{k-1}^H\} - \beta \mathcal{E}\{\alpha_{opt,k}^* Y_k (X_{k-1}^H Y_k) Y_k^H\} \quad \dots (D24)$$

Then, by orthogonality and rearrangement of the scalar quantity $X_{k-1}^H Y_k$, one obtains

$$\mathcal{E}\{\alpha_{opt,k}^* Y_k X_{k-1}^H [I - \beta Y_k Y_k^H]\} = 0 - \beta \mathcal{E}_X\{X_{k-1}^H \mathcal{E}_Y\{Y_k \alpha_{opt,k}^* Y_k Y_k^H\}\} \quad \dots (D25)$$

Applying the Gaussian moment-factoring theorem (D22) then leads to

$$\mathcal{E}_Y\{Y_k \alpha_{opt,k}^* Y_k Y_k^H\} = \mathcal{E}_Y\{Y_k \alpha_{opt,k}^*\} R + R \mathcal{E}_Y\{\alpha_{opt,k}^* Y_k\} \quad \dots (D26)$$

which is again equal to zero by the orthogonality condition. Then what remains of (D25) is expressed as

$$K_k = \mathcal{E}\{[I - \beta Y_k Y_k^H] X_{k-1} X_{k-1}^H [I - \beta Y_k Y_k^H]\} + \beta^2 R mmse \quad \dots (D27)$$

It remains to unravel the first term above. The first step leads to the expression

$$K_k = K_{k-1} - \beta \mathcal{E}\{X_{k-1} X_{k-1}^H Y_k Y_k^H\} - \beta \mathcal{E}\{Y_k Y_k^H X_{k-1} X_{k-1}^H\} + \beta^2 \mathcal{E}\{Y_k Y_k^H X_{k-1} X_{k-1}^H Y_k Y_k^H\} + \beta^2 R mmse \quad \dots (D28)$$

It is shown in [31], using the same independence assumptions, that this simplifies to

$$K_k = K_{k-1} - \beta (K_{k-1} R + R K_{k-1}) + \beta^2 (R Tr[R K_{k-1}] + R K_{k-1} R) + \beta^2 R mmse \quad \dots (D29)$$

The matrix R in this, and in the expression for $emse$, may be diagonalized using the same similarity transformation as was used in Appendix B. The matrix K is also transformed, to matrix C as $K_{k-1} = E C_{k-1} E^H$. Then the expression (D17) is manipulated as

$$Tr[RK_{k-1}] = Tr[(E\Lambda E^H)(EC_{k-1}E^H)] = Tr[C_{k-1}E^HE\Lambda] = Tr[\Lambda C_{k-1}] \dots (D30)$$

because the trace of a product of matrices is independent of their order (these being $N \times N$). This in turn can be written more simply by defining the vectors of the diagonal elements of the matrices, $\underline{\lambda} \triangleq diag[\Lambda]$ and $\underline{C}_{k-1} \triangleq diag[C_{k-1}]$, as

$$Tr[RK_{k-1}] = \underline{\lambda}^H \underline{C}_{k-1}, \dots (D31)$$

which may be substituted in (D17), then in (D12) to yield equation (97) in the thesis body. Therefore it is required to find a recursive relation for only the diagonal terms of C_k . To that end, equation (D29) is rewritten as

$$C_k = C_{k-1} - \beta (C_{k-1}\Lambda + \Lambda C_{k-1}) + \beta^2 \Lambda [\underline{\lambda}^H C_{k-1}] + \beta^2 \Lambda C_{k-1} \Lambda + \beta^2 \Lambda mmse. \dots (D32)$$

Since only the diagonal elements are of interest in (D32), one may also write it as

$$\underline{C}_k = \underline{C}_{k-1} - 2\beta \Lambda \underline{C}_{k-1} + \beta^2 \underline{\lambda} [\underline{\lambda}^H \underline{C}_{k-1}] + \beta^2 \Lambda^2 \underline{C}_{k-1} + \beta^2 \underline{\lambda} mmse. \dots (D33)$$

That is essentially equation (98) one iteration later. Using the boxed relations in this and in the first section, the dynamic evolution of the stationary $mmse_k$ was computed and plotted in Figures 13 - 18 in the body of the thesis.

The steady state misadjustment may be computed from these same relations as $k \rightarrow \infty$. In that case, $\underline{C}_k = \underline{C}_{k-1} = \underline{C}_{k-2}$ in (D33) by definition of steady state, so the equivalent $\underline{C}_\infty$ may be isolated on the LHS. But that achieves only the following form

$$[I - (I - 2\beta\Lambda + \beta^2[\mathcal{Q}\mathcal{Q}^H - \Lambda^2])] \mathbf{C}_\infty = \beta^2 \mathcal{Q} mmse, \quad \dots (D34)$$

in which $\mathbf{C}_\infty$ must still be separated from its matrix coefficient. Fortunately, that matrix is in a "friendly" form with respect to the matrix inversion lemma (see Appendix E). Accordingly, its inverse is computed simply as

$$[(2\beta\Lambda - \beta^2\Lambda^2) - \beta^2\mathcal{Q}\mathcal{Q}^H]^{-1} = (2\beta\Lambda - \beta^2\Lambda^2)^{-1} - \frac{(2\beta\Lambda - \beta^2\Lambda^2)^{-1}\mathcal{Q}\mathcal{Q}^H(2\beta\Lambda - \beta^2\Lambda^2)^{-1}}{\mathcal{Q}^H(2\beta\Lambda - \beta^2\Lambda^2)^{-1}\mathcal{Q} - 1/\beta^2} \quad \dots (D35)$$

where the only remaining matrix inverse is diagonal. Substitution into (D34) and some manipulations result in

$$\begin{aligned} \mathbf{C}_\infty &= (\beta^2 mmse)(2\beta\Lambda - \beta^2\Lambda^2)^{-1}\mathcal{Q} \left[I - \frac{\mathcal{Q}^H(2\beta\Lambda - \beta^2\Lambda^2)^{-1}\mathcal{Q}}{\mathcal{Q}^H(2\beta\Lambda - \beta^2\Lambda^2)^{-1}\mathcal{Q} - 1/\beta^2} \right] = \\ &= \frac{\beta^2(2\beta\Lambda - \beta^2\Lambda^2)^{-1}\mathcal{Q} mmse}{1 - \beta^2\mathcal{Q}^H(2\beta\Lambda - \beta^2\Lambda^2)^{-1}\mathcal{Q}} \quad \dots (D36) \end{aligned}$$

After substitution in (D31) for the purpose of evaluating (D17) at $k \rightarrow \infty$, the final result is obtained more expediently than in [31] as

$$emse_\infty = \frac{mmse \beta^2 \mathcal{Q}^H (2\beta\Lambda - \beta^2\Lambda^2)^{-1} \mathcal{Q}}{1 - \beta^2 \mathcal{Q}^H (2\beta\Lambda - \beta^2\Lambda^2)^{-1} \mathcal{Q}} = \frac{mmse \beta \sum_{n=1}^N \frac{\lambda_n^2}{2\lambda_n - \beta \lambda_n^2}}{1 - \beta \sum_{n=1}^N \frac{\lambda_n^2}{2\lambda_n - \beta \lambda_n^2}} \quad \dots (D37)$$

This is equation (99) of the thesis body with $\beta_k \approx \beta = (g/\tau)/(1 + (g/\tau)Tr[R])$.

If the algorithm is to be convergent in the mean square sense, then mse_{∞} must be finite. That in turn implies that $emse_{\infty}$ should be finite. The necessary condition is clearly seen to be that $2 > \beta \lambda_n$ for all n . That will be true providing that $\lambda_n \neq 0$ and

$$\boxed{\beta < \frac{2}{\lambda_{\max}}} \quad \dots(D38)$$

Note that, since $Tr[R] = \sum_{n=1}^N \lambda_n > \lambda_{\max}$, the step-size parameter β chosen in the thesis will always result in a stable NLMS algorithm. Then one may make the approximation that $2 > \beta \lambda_n$ and simplify (D38) to

$$\frac{emse_{\infty}}{mmse} \approx \frac{\frac{\beta}{2} Tr[R]}{1 - \frac{\beta}{2} Tr[R]} \quad \dots(D39)$$

which is termed the *misadjustment ratio* (stationary). When $\beta = 1/Tr[R]$ as in the fastest version of the NLMS algorithm compared in the thesis, then the steady-state stationary misadjustment ratio becomes approximately unity, i.e.

$$M_{\infty} \doteq \frac{emse_{\infty}}{mmse} \approx 1 \quad \dots(D40)$$

FRQN Algorithm - Transient and Steady-State mse:

Having derived the expressions for the various components of the mean squared output error in the development for the NLMS algorithm, one can now apply them also to the FRQN algorithm. The expression for $mmse$ is still the same as (D5) and (D17) also holds for the $emse_{\infty}$, which may be evaluated as an inner product according to (D31). To that end, it is of interest to first obtain the weight-error covariance matrix

corresponding to K_k , which will be designated as H_k for the FRQN algorithm. The corresponding weight-error vector will be designated by Z_k (not to be confused with Z in Appendix B), so that

$$Z_{k-1} \doteq W_{k-1} - W_{opt} \quad \dots(D41)$$

and

$$H_{k-1} \doteq \mathcal{E}\{Z_{k-1}Z_{k-1}^H\} \quad \dots(D42)$$

(The weights are still designated by W as before, with the understanding that they are different from those in the NLMS algorithm, except for W_{opt} .)

As in the NLMS analysis, one first obtains a recursive difference equation for Z_k (for deriving H_k rather than H_{k-1} , again) by rewriting the algorithm's defining difference equation in terms of the weight-error vector. In the case of FRQN, the most general form corresponding to equation (95) of the thesis will be used, in order to allow for any opportunities for optimization of parameters. For convenience however, one would like to start with a form similar to (63), which was shown in the thesis to lead to the final form (64) or (80). That differs from the more general one (95) only in the values of parameters of the latter. By following the same derivation for (63), rewritten in terms of Z , one can see the substitutions for Z and α which can be made in the general final form directly. To begin, rewrite (63) as

$$\begin{aligned} \tau_1 \tau_2 (Z_k - 2Z_{k-1} + Z_{k-2}) + \tau_2 [\gamma I + g Y_k Y_k^H] (Z_k - Z_{k-1}) &= \\ &= -g Y_k Y_k^H (W_k - W_{opt}) - g Y_k Y_k^H W_{opt} + g Y_k r_k^* = \\ &= -g Y_k Y_k^H Z_k - g Y_k (\alpha_{opt,k}^*), \end{aligned} \quad \dots(D43)$$

which is the FRQN counterpart of (D19). It has exactly the same form as the original equation (63) in the thesis, with the substitution of Z for W and $-\alpha_{opt}$ for r . Those substitutions can therefore be made directly in (64). Because that in turn is in the same

form as the more general equation (95), one can also make the same kind of substitutions in it and thus obtain the general FRQN counterpart of (D20) as

$$Z_k = A_k Z_{k-1} + B_k Z_{k-2} + c_k Y_k \alpha_{opt,k}^* \quad \dots(D44)$$

where

$$A_k = \left(\frac{a_1}{a_0} \right) \left[I + z_k Y_k Y_k^H \right], \quad \dots(D45)$$

$$B_k = - \left(\frac{a_2}{a_0} \right) \left[I - x_k Y_k Y_k^H \right], \quad \dots(D46)$$

and

$$c_k = \frac{-4 \rho^2 g j b_0}{a_0 / b_0 + Y_k Y_k^H}. \quad \dots(D47)$$

The variables z_k , x_k and c_k are defined in the thesis following equation (105), in terms of the parameters which follow equation (95) itself. For tractability through the Gaussian moment-factoring theorem and other simplifications, they are assumed to be independent of k (constants) by the same reasoning as for the step-size parameter β in the NLMS treatment. (For computing the plots in Figures 13 - 18 of the thesis body, they were left as variables dependent on k as in equation (105)). Now the FRQN counterpart of (D21) is derived using the simple form (D44):

$$\begin{aligned} H_k = & \mathcal{E}\{A_k Z_{k-1} Z_{k-1}^H A_k^H\} + \mathcal{E}\{A_k Z_{k-1} Z_{k-2}^H B_k^H\} + c \mathcal{E}\{A_k Z_{k-1} Y_k^H \alpha_{opt,k}^*\} + \\ & + \mathcal{E}\{B_k Z_{k-2} Z_{k-1}^H A_k^H\} + \mathcal{E}\{B_k Z_{k-2} Z_{k-2}^H B_k^H\} + c \mathcal{E}\{B_k Z_{k-2} Y_k^H \alpha_{opt,k}^*\} + \dots(D48) \\ & - c \mathcal{E}\{\alpha_{opt,k}^* Y_k Z_{k-1}^H A_k^H\} + c \mathcal{E}\{\alpha_{opt,k}^* Y_k Z_{k-2}^H B_k^H\} + c^2 \mathcal{E}\{\alpha_{opt,k}^* Y_k Y_k^H \alpha_{opt,k}^*\}. \end{aligned}$$

Now the same pattern as that leading from (D21) in the NLMS development may be followed, except that a greater variety of cross-expectation terms must be expanded. For

example, it is readily seen by comparing the last term above to (D23), that it is equal to

$$c^2 \mathcal{E}\{\alpha_{opt,k}^* Y_k Y_k^H \alpha_{opt,k}\} = c^2 Rmmse . \quad \dots(D49)$$

Similarly, the familiar forms of the cross-terms which contain Y_k and $\alpha_{opt,k}$ have expectations of zero, by the same reasoning as in (D24)-(D26), i.e.

$$\begin{aligned} \mathcal{E}\{A_k Z_{k-1} Y_k^H \alpha_{opt,k}\} &= 0 , \\ \mathcal{E}\{B_k Z_{k-2} Y_k^H \alpha_{opt,k}\} &= 0 , \end{aligned} \quad \dots(D50)$$

and similarly for their complex conjugates in (D48). Then what remains of (D48) may be expressed as

$$\begin{aligned} H_k &= \mathcal{E}\{A_k Z_{k-1} Z_{k-1}^H A_k^H\} + \mathcal{E}\{B_k Z_{k-2} Z_{k-2}^H B_k^H\} + c^2 Rmmse + \\ &\quad + \mathcal{E}\{B_k Z_{k-2} Z_{k-1}^H A_k^H\} + \mathcal{E}\{A_k Z_{k-1} Z_{k-2}^H B_k^H\} . \end{aligned} \quad \dots(D51)$$

Except for the last two terms on the RHS, the above equation has a familiar form and its terms may be readily expanded by the same procedure as for (D27) in the NLMS development. Accordingly, the first term on the RHS is expanded as

$$\begin{aligned} \mathcal{E}\{A_k Z_{k-1} Z_{k-1}^H A_k^H\} &= \left(\frac{a_1}{a_0}\right)^2 \mathcal{E}\{[I + z Y_k Y_k^H] Z_{k-1} Z_{k-1}^H [I + z Y_k Y_k^H]\} = \\ &= \left(\frac{a_1}{a_0}\right)^2 \left[H_{k-1} + z \mathcal{E}\{Y_k Y_k^H Z_{k-1} Z_{k-1}^H\} + z \mathcal{E}\{Z_{k-1} Z_{k-1}^H Y_k Y_k^H\} + \right. \\ &\quad \left. + z^2 \mathcal{E}\{Y_k Y_k^H Z_{k-1} Z_{k-1}^H Y_k Y_k^H\} \right] \end{aligned} \quad \dots(D52)$$

and

$$\begin{aligned} \mathcal{E}\{B_k Z_{k-2} Z_{k-2}^H B_k^H\} &= \left(\frac{-a_2}{a_0}\right)^2 \mathcal{E}\{[I + x Y_k Y_k^H] Z_{k-2} Z_{k-2}^H [I + x Y_k Y_k^H]\} = \\ &= \left(\frac{a_2}{a_0}\right)^2 \left[H_{k-2} + x \mathcal{E}\{Y_k Y_k^H Z_{k-2} Z_{k-2}^H\} + x \mathcal{E}\{Z_{k-2} Z_{k-2}^H Y_k Y_k^H\} + \right. \\ &\quad \left. + x^2 \mathcal{E}\{Y_k Y_k^H Z_{k-2} Z_{k-2}^H Y_k Y_k^H\} \right] . \end{aligned} \quad \dots(D53)$$

These are again simplified using the results in [31] as

$$\begin{aligned} \mathcal{E}\{A_k Z_{k-1} Z_{k-1}^H A_k^H\} &= \\ &= \left(\frac{a_1}{a_0}\right)^2 \left[H_{k-1} + z(H_{k-1}R + RH_{k-1}) + z^2(RTr[RH_{k-1}] + RH_{k-1}R) \right] \end{aligned} \quad \dots(D54)$$

and

$$\begin{aligned} \mathcal{E}\{B_k Z_{k-2} Z_{k-2}^H B_k^H\} &= \\ &= \left(\frac{a_2}{a_0}\right)^2 \left[H_{k-2} + x(H_{k-2}R + RH_{k-2}) + x^2(RTr[RH_{k-2}] + RH_{k-2}R) \right]. \end{aligned} \quad \dots(D55)$$

Define the result of the similarity transformation of H as N , i.e.

$$H_{k-1} = E N_{k-1} E^H \quad \dots(D56)$$

and let $\mathbf{N}$ be the column vector of its diagonal entries, so that $\mathbf{N}_{k-1} \triangleq \text{diag}[\mathbf{N}_{k-1}]$.

Then one can use the same procedure as was used to obtain (D33) from (D29), to arrive at the result

$$\begin{aligned} \mathbf{N}_k &= \left(\frac{a_1}{a_0}\right)^2 \left[\mathbf{N}_{k-1} + 2z\Lambda \mathbf{N}_{k-1} + z^2 \mathcal{E}[\mathcal{E}^H \mathbf{N}_{k-1}] + z^2 \Lambda^2 \mathbf{N}_{k-1} \right] + \\ &+ \left(\frac{a_2}{a_0}\right)^2 \left[\mathbf{N}_{k-2} + 2x\Lambda \mathbf{N}_{k-2} + x^2 \mathcal{E}[\mathcal{E}^H \mathbf{N}_{k-2}] + x^2 \Lambda^2 \mathbf{N}_{k-2} \right] + \\ &+ \text{diag} \left[\mathcal{E}\{B_k Z_{k-2} Z_{k-1}^H A_k^H\} - \mathcal{E}\{A_k Z_{k-1} Z_{k-2}^H B_k^H\} \right] + c^2 \mathcal{E} \text{mmse}. \end{aligned} \quad \dots(D57)$$

It remains to expand and simplify the terms containing the cross-products of Z_{k-1} and Z_{k-2} above. Since $\mathbf{N}_k$ must ultimately be real, they must be complex conjugates of each other so it is sufficient to expand only one of them:

$$\begin{aligned} \mathcal{E}\{A_k Z_{k-1} Z_{k-2}^H B_k^H\} &= \left(\frac{-a_1 a_2}{a_0^2}\right) \mathcal{E}\{[I + zY_k Y_k^H] Z_{k-1} Z_{k-2}^H [I + xY_k Y_k^H]\} = \\ &= \left(\frac{-a_1 a_2}{a_0^2}\right) \left[H_{k1,2} + z \mathcal{E}\{Y_k Y_k^H Z_{k-1} Z_{k-2}^H\} + x \mathcal{E}\{Z_{k-1} Z_{k-2}^H Y_k Y_k^H\} + \right. \\ &\quad \left. + zx \mathcal{E}\{Y_k Y_k^H Z_{k-1} Z_{k-2}^H Y_k Y_k^H\} \right] \end{aligned} \quad \dots(D58)$$

where it was decided to define $H_{k1,2} \doteq \mathcal{E}\{Z_{k-1}Z_{k-2}^H\}$. With the help of this definition and [31] again, equation (D58) above is simplified to

$$\begin{aligned} \mathcal{E}\{A_k Z_{k-1} Z_{k-2}^H B_k^H\} = & \dots (D59) \\ = \left(\frac{-a_1 a_2}{a_0^2} \right) & \left[H_{k1,2} + (x H_{k1,2} R + z R H_{k1,2}) + z x (R T r [R H_{k1,2}] + R H_{k1,2} R) \right]. \end{aligned}$$

Now the straightforward similarity-transformation substitution and subsequent vectorization of the diagonal entries leads to

$$\begin{aligned} \text{diag}[\mathcal{E}\{B_k Z_{k-2} Z_{k-1}^H A_k^H\} + \mathcal{E}\{A_k Z_{k-1} Z_{k-2}^H B_k^H\}] = & \dots (D60) \\ = \left(\frac{-a_1 a_2}{a_0^2} \right) & 2 \Re e [N_{k1,2} + (x + z) \Lambda N_{k1,2} + x z (\mathcal{L}[\mathcal{L}^H N_{k1,2}] + \Lambda^2 N_{k1,2})]. \end{aligned}$$

In the case of the FRQN (and other second-order algorithms), one is faced with the unknown quantity $N_{k1,2}$. One should not attempt to define it in terms of the presumed first-order-dominant, quasi-Newton convergence (i.e. exponential with time-constant τ_2), in order not to inadvertently force that kind of convergence on mse_k , whose convergence behaviour is yet to be revealed through this development. One can, however, make an intelligent approximation by defining $N_{k1,2}$ in terms of "known" or familiar quantities. Recall that W_k is a much slower process than Y_k , as noted in Appendix B. Accordingly, the autocorrelation function of W_k will be much broader, meaning that W_{k-1} and W_{k-2} are highly correlated. That in turn implies that only relatively minor artefacts would occur if one was substituted for the other in the cross-covariance term. Seeing thus that $N_{k1,2}$ may be approximated by either N_{k-1} or N_{k-2} and not wishing to bias the outcome, the democratic approximation

$$N_{k1,2} \approx \frac{1}{2} (N_{k-1} + N_{k-2}) \quad \dots (D61)$$

is made in (D60). Upon substitution in (D57) and some regrouping of terms, that leads to what is essentially equation (105) of the thesis, i.e.

$$\begin{aligned}
 N_k = & \left(\frac{a_1}{a_0} \right)^2 \left[(I + z\Lambda)^2 + z^2 \mathcal{Q} \mathcal{Q}^H \right] N_{k-1} - \\
 & - 2 \Re \left\{ \left(\frac{a_1 a_2}{a_0^2} \right) \left[(I + z\Lambda)(I + x\Lambda) + zx \mathcal{Q} \mathcal{Q}^H \right] \left(\frac{1}{2} N_{k-1} + \frac{1}{2} N_{k-2} \right) \right\} + \\
 & + \left(\frac{a_2}{a_0} \right)^2 \left[(I + x\Lambda)^2 + x^2 \mathcal{Q} \mathcal{Q}^H \right] N_{k-2} + c^2 \mathcal{Q} mmse .
 \end{aligned} \quad \dots (D62)$$

with

$$mse_k = \mathcal{Q}^H N_{k-1} + mmse . \quad \dots (D63)$$

The transient evolution of the stationary $mmse_k$ of the FRQN algorithm was computed using the boxed expressions in this section, except N_k was used for N_{k-1} in (D63) and the variables x_k , z_k , and c_k were allowed to vary with k as defined in (D45)-(D47), with parameters a_0 , a_1 , a_2 , b_0 , b_1 and b_2 defined as in equation (95) of the thesis. In those parameters, the special case of $\rho = 1/2$, $g \rightarrow \infty$, $\gamma = 1/\tau_2$, $\tau_2 = 4N$, $\tau_1 = \tau_2/4$ and $q = 2$ was used for plotting the theoretical mean squared errors in Figures 13-18 of the thesis. Note that the artefacts incurred by the approximation (D61) appear as periodic lobes in $mmse_k$, but they disappear in the steady state.

In the steady state as $k \rightarrow \infty$, $N_k = N_{k-1} = N_{k-2} = N_{k+2} = \dots = N_\infty$ by definition, regardless of how N_{k+2} is approximated. Therefore the stationary steady-state misadjustment error of the FRQN algorithm may be computed similarly as it was for the NLMS algorithm, beginning with the isolation of the resulting N_∞ terms on the LHS in (D62). As $k \rightarrow \infty$, (D62) simplifies to

$$\begin{aligned} \mathbf{N}_\mathbf{u} = & \left[\left(\frac{a_1}{a_0} \right) (I + z\Lambda) - \left(\frac{a_2}{a_0} \right) (I + x\Lambda) \right]^2 \mathbf{N}_\mathbf{u} + \\ & + \mathfrak{L} \mathfrak{L}^H \left[\left(\frac{a_1}{a_0} \right) z - \left(\frac{a_2}{a_0} \right) x \right]^2 \mathbf{N}_\mathbf{u} + c^2 \mathfrak{L} mmse . \end{aligned} \quad \dots(D64)$$

Therefore,

$$\left[I - \left[\left(\frac{a_1 - a_2}{a_0} \right) I + \left(\frac{a_1 z - a_2 x}{a_0} \right) \Lambda \right]^2 - \mathfrak{L} \mathfrak{L}^H \left(\frac{a_1 z - a_2 x}{a_0} \right)^2 \right] \mathbf{N}_\mathbf{u} = c^2 \mathfrak{L} mmse . \quad \dots(D65)$$

The matrix on the LHS has the same friendly form as (D34) in the NLMS development, so the procedure to obtain the steady-state, stationary excess mean squared error of the FRQN algorithm is the same, only more tedious. To minimize the notational baggage, temporarily let

$$h = \frac{a_1 - a_2}{a_0} \quad \dots(D66)$$

and

$$f = \frac{a_1 z - a_2 x}{a_0} . \quad \dots(D67)$$

Then identify the *emse* in (D63) and write it as

$$emse_\mathbf{u} = \mathfrak{L}^H \mathbf{N}_\mathbf{u} = \mathfrak{L}^H [I - [hI + f\Lambda]^2 - f^2 \mathfrak{L} \mathfrak{L}^H]^{-1} \mathfrak{L} c^2 mmse \quad \dots(D68)$$

where one may notice the purely diagonal matrix which will be denoted as

$$\mathbf{D} = I - [hI + f\Lambda]^2 . \quad \dots(D69)$$

Then the expression for the FRQN steady-state misadjustment error becomes simply

$$\mathbf{M}_{\mathbf{u}} = \mathbf{c}^H [D - f^2 \mathbf{c}^H \mathbf{c}]^{-1} \mathbf{c}^2, \quad \dots(\text{D70})$$

where the matrix-inversion is again performed with the help of the matrix-inversion lemma (see Appendix E), as it was in (D35) for the NLMS algorithm. Omitting the tedious details, the result is found to be in the same form as (D37):

$$\mathbf{M}_{\mathbf{u}} = \frac{\left(\frac{c}{f}\right)^2 \sum_{n=1}^N \frac{(f\lambda_n)^2}{1 - (h + f\lambda_n)^2}}{1 - \sum_{n=1}^N \frac{(f\lambda_n)^2}{1 - (h + f\lambda_n)^2}}. \quad \dots(\text{D71})$$

After completion of the tedious task of substitution for h , f and c , then for a_0, a_1, a_2, b_0, b_1 and b_2 , it is found that

$$\begin{aligned} c &= \frac{\left(\frac{-4\rho^2}{1 + 2\rho\tau_2/q}\right)}{\frac{2\rho\tau_2(2\rho\tau_1 + \gamma)}{g(1 + 2\rho\tau_2/q)} + Y^H Y} = \\ &= \frac{-4\rho^2 g}{2\rho\tau_2(2\rho\tau_1 + \gamma) + g(1 + 2\rho\tau_2/q)Y^H Y}, \end{aligned} \quad \dots(\text{D72})$$

$$h = \frac{4\rho\tau_2(2\rho\tau_1 + \gamma(1 - \rho)) - 2\rho\tau_2(2\rho\tau_1 + \gamma(1 - 2\rho))}{2\rho\tau_2(2\rho\tau_1 + \gamma)} \quad \dots(\text{D73})$$

and

$$\begin{aligned} f &= \frac{a_1}{a_0} \left(\frac{b_1 a_0 / (a_1 b_0) - 1}{a_0 / b_0 + Y^H Y} \right) - \frac{a_2}{a_0} \left(\frac{b_2 a_0 / (a_2 b_0) - 1}{a_0 / b_0 + Y^H Y} \right) = \\ &= \frac{(b_1 / b_0 - a_1 / a_0) - (b_2 / b_0 - a_2 / a_0)}{a_0 / b_0 + Y^H Y}. \end{aligned} \quad \dots(\text{D74})$$

The last two equations simplify further: firstly,

$$h = \frac{2\rho\tau_1 + \gamma}{2\rho\tau_1 + \gamma} = 1, \quad \dots(D75)$$

which, along with c , subsequently becomes useful in simplifying f as

$$\begin{aligned} f &= \frac{b_1 - b_2 + b_0(a_2 - a_1)/a_0}{a_0 + b_0 Y^H Y} = \dots(D76) \\ &= \frac{\left[2g(1 - 2\rho) + 4\rho\tau_2 \left(\frac{g}{q} \right) (1 - \rho) \right] - \left[g(1 - 2\rho)^2 + 2\rho\tau_2 \left(\frac{g}{q} \right) (1 - 2\rho) \right] - hb_0}{2\rho\tau_2(2\rho\tau_1 + \gamma) + g(1 + 2\rho\tau_2/q)Y^H Y}. \end{aligned}$$

After further manipulation, (D76) is found to finally simplify to c as

$$\begin{aligned} f &= \frac{g(1 - 2\rho)(2 - (1 - 2\rho)) + 2\rho\tau_2 \left(\frac{g}{q} \right) (2 - 2\rho - (1 - 2\rho)) - b_0}{2\rho\tau_2(2\rho\tau_1 + \gamma) + g(1 + 2\rho\tau_2/q)Y^H Y} = \\ &= \frac{g(2 - 4\rho - 1 + 4\rho - 4\rho^2) + 2\rho\tau_2 \left(\frac{g}{q} \right) - g(1 + 2\rho\tau_2/q)}{2\rho\tau_2(2\rho\tau_1 + \gamma) + g(1 + 2\rho\tau_2/q)Y^H Y} = \\ &= \frac{-4\rho^2 g}{2\rho\tau_2(2\rho\tau_1 + \gamma) + g(1 + 2\rho\tau_2/q)Y^H Y} = \dots(D77) \\ &= c. \end{aligned}$$

That helps to use the positive quantity $p = -c$ to reduce the misadjustment expression (D71) to

$$\mathbf{M}_m = \frac{\sum_{n=1}^N \frac{(-p\lambda_n)^2}{1 - (1 - p\lambda_n)^2}}{1 - \sum_{n=1}^N \frac{(-p\lambda_n)^2}{1 - (1 - p\lambda_n)^2}} = \frac{\sum_{n=1}^N \frac{(p\lambda_n)^2}{2p\lambda_n - (p\lambda_n)^2}}{1 - \sum_{n=1}^N \frac{(p\lambda_n)^2}{2p\lambda_n - (p\lambda_n)^2}}$$

or finally, letting

$$p = \frac{4\rho^2 g}{2\rho\tau_2(2\rho\tau_1 + \gamma) + g(1 + 2\rho\tau_2/q)Y^H Y} \quad \dots(D78)$$

it becomes essentially equation (106) of the thesis:

$$M_{\infty} = \frac{\sum_{n=1}^N \frac{p \lambda_n}{2 - p \lambda_n}}{1 - \sum_{n=1}^N \frac{p \lambda_n}{2 - p \lambda_n}} . \quad \dots(D79)$$

For convergence of the FRQN algorithm in the mean square, it is clearly required to ensure that

$$p < \frac{2}{\lambda_{\max}} . \quad \dots(D80)$$

That will clearly be satisfied if $\lambda_{\max}$ is replaced with $Tr[R] = \sum_{n=1}^N \lambda_n = \mathcal{E}\{Y^H Y\}$, which can also be used in p so that

$$\frac{4\rho^2 g}{2\rho\tau_2(2\rho\tau_1 + \gamma) + g(1 + 2\rho\tau_2/q)Tr[R]} < \frac{2}{Tr[R]} , \quad \dots(D81)$$

or equivalently

$$\frac{4\rho^2 g - 2g - 4\rho g \tau_2/q}{4\rho\tau_2(2\rho\tau_1 + \gamma)} < \frac{1}{Tr[R]} . \quad \dots(D82)$$

Since g can be arbitrarily large and is always > 0 , the LHS of (D82) must always be < 0 . That condition leads to the equivalent stability requirement for the FRQN algorithm, namely that

$$2\rho^2 - 1 - \frac{2\rho\tau_2}{q} < 0 . \quad \dots(D83)$$

For the selected backward-difference s-to-z transform version of the algorithm $\rho = 1/2$, requiring that the effective nominal "closed-loop" time-constant

satisfies

$$\frac{\tau_2}{q} > -\frac{1}{2} . \quad \dots(D84)$$

Since that time-constant is always positive, this version of the FRQN algorithm is always stable in the mean square, as well as in the mean.

For the bilinear s-to-z transform version $\rho = 1$, requiring that the time-constant should satisfy

$$\frac{\tau_2}{q} > \frac{1}{2} , \quad \dots(D85)$$

which should be achieved in most circumstances. However, the special case with $g \rightarrow \infty$ in (D78) reveals that

$$p \rightarrow \frac{4\rho^2}{(1 + 2\rho\tau_2/q)Tr[R]} \quad \dots(D86)$$

and assuming $p\lambda_n < 2$ in (D79), that

$$M_{\infty} = \frac{\frac{1}{2} \left(\frac{4\rho^2}{1 + 2\rho\tau_2/q} \right)}{1 - \frac{1}{2} \left(\frac{4\rho^2}{1 + 2\rho\tau_2/q} \right)} = \frac{2\rho^2}{1 + 2\rho\tau_2/q - 2\rho^2} \quad \dots(D87)$$

For $\rho = 1/2$, this becomes $M_{\infty} = 1/(1 + 2\tau_2/q)$ and for $\rho = 1$ it becomes $M_{\infty} = 2/(\tau_2/q - 1)$, which requires that

$$\frac{\tau_2}{q} > 1 . \quad \dots(D88)$$

This is not difficult to meet in the deterministic sense with $q = 2$ and $\tau_2 = 4N$, with $N > 1$ being typically used, but it being a stochastic algorithm in practice, one may be justified in expecting a finite potential for instability to exist on the basis of (D88) in light of the foregoing developments. That potential clearly does not exist for the case of $\rho = 1/2$.

For completeness, the possibility of the existence of an optimal s-to-z transformation in the sense of minimizing $M_{\mathbf{u}}$ in (D87) was investigated, by differentiation of (D87) with respect to ρ , and the restriction $0 < \rho < 1$. None was found. Therefore the version of FRQN algorithm with $\rho = 1/2$ (i.e. the backward-difference s-to-z transformation) was selected as the most robust as well as fast algorithm for characterization in this thesis. With the approximation of large gain $g \rightarrow \infty$, its steady-state excess mean squared error is given simply by

$$emse_{\mathbf{u}} = \frac{mmse}{1 + 2\left(\frac{\tau_2}{q}\right)} . \quad \dots(D89)$$

Note that if $q \rightarrow \infty$ with $\gamma = 1$, the FRQN algorithm becomes the "ordinary" second-order LMS and its excess mean squared error will become the minimum mean squared error, like in the NLMS algorithm, thus achieving no advantage.

Non-Stationary Signals:

In the following developments β_k will again be treated as a constant quantity, approximated as $\beta_k \approx \beta = \mu/(\mu + Tr[R])$. Non-stationarity will be modelled in a system-identification context where W_k must track a *time-varying* optimal weight vector, $W_{opt,k}$, which evolves according to a first-order Markov process as

$$W_{opt,k} = \Phi W_{opt,k-1} + \varphi_k , \quad \dots(D90)$$

where $0 < \Phi < 1$ is the parameter of non-stationarity (taken as a real scalar for simplicity) and φ_k is a zero-mean Gaussian random vector process driving it. The "model" system accepts the input data signal Y_k and produces the optimal output signal

$$y_{opt,k} = W_{opt,k-1}^H Y_k \quad \dots(D91)$$

which is subsequently perturbed by a scalar zero-mean complex additive white Gaussian noise (AWGN) process v_k , and becomes the reference signal for the system identifier, as

$$r_k = y_{opt,k} + v_k \quad \dots(D92)$$

The adaptive system-identifier strives to minimize the difference between r_k and the output of the "plant" system which operates on the same input signal Y_k with adaptive coefficients W_k , in the mean square sense.

NLMS Algorithm - Steady-State Weakly Non-Stationary Tracking Error, $tmse$:

The idea is that the plant coefficients will track the model coefficients with minimal error. That error vector now replaces the one defined in (D7) by

$$X_{k-1} = W_{k-1} - W_{opt,k-1} \quad \dots(D93)$$

from which W_{k-1} (and similarly W_k), along with r_k from (D92), are substituted into the NLMS equation similarly as in (D19). After some straightforward manipulation the result is found to be very similar to (D20) but for an additional term

$$X_k = [I - \beta Y_k Y_k^H] X_{k-1} + \beta Y_k v_k^* - \Delta W_{opt,k} \quad \dots(D94)$$

in which the additional term is the *lag error* defined as

$$\Delta W_{opt,k} \triangleq W_{opt,k} - W_{opt,k-1} \quad \dots(D95)$$

Again, the same independence assumptions are applied in computing the covariance of

the weight-error vector, as on (D21), with the additional assumption that the lag error is independent of everything else. In this case $mmse = \mathcal{E}\{v_k v_k^*\}$, and by following the same procedure as from (D21) to (D29) with the above in mind, one arrives at the result

$$K_k = K_{k-1} - \beta (RK_{k-1} - K_{k-1}R) + \beta^2 (RTr[RK_{k-1}] + RK_{k-1}R) + \beta^2 Rmmse + \mathcal{E}\{\Delta W_{opt,k} \Delta W_{opt,k}^H\} \quad \dots (D96)$$

By applying the same "diagonalization" principles as embodied within (D30) and (D31) for the stationary case, one obtains the non-stationary counterpart of (D33) as

$$\boxed{C_k = [I - 2\beta\Lambda + \beta^2(\mathcal{Q}\mathcal{Q}^H + \Lambda^2)]C_{k-1} + \beta^2\mathcal{Q}mmse + diag[\mathcal{E}\{\Delta W_{opt,k} \Delta W_{opt,k}^H\}]} \quad \dots (D97)$$

which can be used in (D31) (one iteration later) to compute the *emse* as in (D17), but it will include a *tracking error* due to the lag error in the last term of (D97). In this development the steady-state tracking error is of interest; it can be computed from (D97) by letting $k \rightarrow \infty$ so as to obtain a form similar to (D34), i.e.

$$[I - (I - 2\beta\Lambda + \beta^2[\mathcal{Q}\mathcal{Q}^H + \Lambda^2])]C_\infty = \beta^2\mathcal{Q}mmse + diag[\mathcal{E}\{\Delta W_{opt,k} \Delta W_{opt,k}^H\}] \quad \dots (D98)$$

Omitting the recognizable steady-state *stationary emse* contribution in (D98), the steady-state *non-stationary* or *tracking error* contribution can be identified as

$$tmse_\infty = \mathcal{Q}^H [(2\beta\Lambda - \beta^2\Lambda^2) - \beta^2\mathcal{Q}\mathcal{Q}^H]^{-1} diag[\mathcal{E}\{\Delta W_{opt,k} \Delta W_{opt,k}^H\}] \quad \dots (D99)$$

The matrix inverse in (D99) is easily obtained with the help of the matrix-inversion lemma (see Appendix E) and is given in (D35). It remains to find the covariance of the lag error and to "diagonalize" it. To that end, first substitute (D95) into (D90) to obtain

$$\Delta W_{opt,k} = W_{opt,k} - W_{opt,k-1} = (\Phi - 1) W_{opt,k-1} + \varphi_k . \quad \dots(D100)$$

Then compute the covariance matrix of the above as

$$\begin{aligned} \mathcal{E}\{\Delta W_{opt,k} \Delta W_{opt,k}^H\} &= (\Phi - 1)^2 \mathcal{E}\{W_{opt,k-1} W_{opt,k-1}^H\} + \\ &+ (\Phi - 1) [\mathcal{E}\{W_{opt,k-1} \varphi_k^H\} + \mathcal{E}\{\varphi_k W_{opt,k-1}^H\}] + \dots(D101) \\ &+ \mathcal{E}\{\varphi_k \varphi_k^H\} . \end{aligned}$$

Assuming that φ_k is a vector of independent, zero-mean Gaussian random processes with equal variances, σ_φ^2 , the last term in (D101) above is equal to $\sigma_\varphi^2 I$. Furthermore $\mathcal{E}\{W_{opt,k-1} \varphi_k^H\} = [0]$, because, as in any autoregressive (AR) process, $W_{opt,k-1}$ depends *only* on $\{\varphi_{k-1}, \varphi_{k-2}, \dots, \varphi_0\}$ according to the model (D90), so it is *independent* of φ_k . It then remains to find $\mathcal{E}\{W_{opt,k-1} W_{opt,k-1}^H\}$ in the first term of (D101). That may be approximated by assuming that $W_{opt,k}$ is only weakly non-stationary and much slower than X_k , or W_k by itself. Otherwise W_k would not be able to track the optimal weight vector. In that case the incremental change in $W_{opt,k}$ in one iteration is negligible, so according to (D90) with the AR independence assumption,

$$\begin{aligned} \mathcal{E}\{W_{opt,k} W_{opt,k}^H\} &= \Phi^2 \mathcal{E}\{W_{opt,k-1} W_{opt,k-1}^H\} + \sigma_\varphi^2 I = \\ &\approx \mathcal{E}\{W_{opt,k-1} W_{opt,k-1}^H\} . \end{aligned} \quad \dots(D102)$$

whereupon it follows that

$$\boxed{\mathcal{E}\{W_{opt,k-1} W_{opt,k-1}^H\} \approx \left(\frac{\sigma_\varphi^2}{1 - \Phi^2} \right) I} . \quad \dots(D103)$$

Consequently (D101) reduces to

$$\boxed{\mathcal{E}\{\Delta W_{opt,k} \Delta W_{opt,k}^H\} \approx \left(\frac{\sigma_\varphi^2}{1 - \Phi^2} \right) (\Phi - 1)^2 I + \sigma_\varphi^2 I = \left(\frac{2\sigma_\varphi^2}{1 + \Phi} \right) I} . \quad \dots(D104)$$

Then, for the purposes of (D99) the $\text{diag}[I] = J$, i.e. a vector of ones, so the resulting steady-state mean square tracking error may be written as

$$tmse_{\infty} = \mathfrak{L}^H [(2\beta\Lambda - \beta^2\Lambda^2) - \beta^2\mathfrak{L}\mathfrak{L}^H]^{-1} J \left(\frac{2\sigma_{\Phi}^2}{1 + \Phi} \right). \quad \dots(D105)$$

By the same tedious process as (D37) was obtained from (D34) for the stationary $emse_{\infty}$, the tracking error $tmse_{\infty}$ is obtained as

$$tmse_{\infty} = \left[1 - \frac{\mathfrak{L}^H (2\beta\Lambda - \beta^2\Lambda^2)^{-1} \mathfrak{L}}{\mathfrak{L}^H (2\beta\Lambda - \beta^2\Lambda^2)^{-1} \mathfrak{L} - 1/\beta^2} \right] \mathfrak{L}^H (2\beta\Lambda - \beta^2\Lambda^2)^{-1} J \left(\frac{2\sigma_{\Phi}^2}{1 + \Phi} \right), \quad \dots(D106)$$

which simplifies with the help of

$$\mathfrak{L}^H (2\beta\Lambda - \beta^2\Lambda^2)^{-1} \mathfrak{L} = \sum_{n=1}^N \frac{\lambda_n^2}{\beta\lambda_n(2 - \beta\lambda_n)} \quad \dots(D107)$$

and

$$\mathfrak{L}^H (2\beta\Lambda - \beta^2\Lambda^2)^{-1} J = \sum_{n=1}^N \frac{\lambda_n}{\beta\lambda_n(2 - \beta\lambda_n)} \quad \dots(D108)$$

to yield equation (102) of the thesis, i.e.

$$tmse_{\infty} = \left(\frac{2\sigma_{\Phi}^2\beta^{-1}}{1 + \Phi} \right) \left(\sum_{n=1}^N \frac{1}{2 - \beta\lambda_n} \right) \left[\frac{1}{1 - \sum_{n=1}^N \frac{\beta\lambda_n}{2 - \beta\lambda_n}} \right]. \quad \dots(D109)$$

With the usual assumption that $\beta\lambda_n < 2$ and $1/\beta = \sum_{n=1}^N \lambda_n$ when $g \rightarrow \infty$, it reduces to equation (103) of the thesis:

$$tmse_{\beta} = \frac{2\sigma_{\Phi}^2 N}{(1 - \Phi)\beta} = \frac{2\sigma_{\Phi}^2 N \sum_{n=1}^N \lambda_n}{1 + \Phi} \quad \dots(D110)$$

Notice that $tmse_{\beta}$ is *inversely* proportional to the step-size parameter β , whereas $emse_{\beta}$ is *directly* proportional to it, so an optimal value of β can be found if the non-stationarity parameter and driving-process covariance are known.

FRQN Algorithm - Steady-State Weakly Non-Stationary Tracking Error, $tmse$:

The same system-identification model is used to investigate the tracking behaviour of the FRQN algorithm. Definition (D41) now includes a time-varying optimal weight vector, $W_{opt,k-1}$, and the weight vector is again substituted into equation (63). The more general form of the FRQN algorithm (95) is not used in this treatment for clarity, it having been decided in the stationary analysis that the condition $\rho = 1/2$ is essential for maximum robustness and stability margin. Including the "speed-up" factor q (also dubbed the Newton "quasity" factor) in equation (63) and substituting

$$W_{k-1} = Z_{k-1} + W_{opt,k-1} \quad \dots(D111)$$

into the preferred form requires defining a lag error as in (D95), as well as an *acceleration error*

$$\Delta^2 W_{opt,k} \doteq \Delta W_{opt,k} - \Delta W_{opt,k-1} \quad \dots(D112)$$

The resulting form of the difference equation is then obtained similarly as was (D43), i.e.

$$\begin{aligned}
& \tau_1 \tau_2 \Delta^2 Z_k + \tau_2 \left[\gamma I + \left(\frac{g}{q} \right) Y_k Y_k^H \right] \Delta Z_k = \\
& = g Y_k Y_k^H Z_k + g Y_k v_k^* - \tau_1 \tau_2 \Delta^2 W_{opt,k} - \tau_2 \left[\gamma I + \left(\frac{g}{q} \right) Y_k Y_k^H \right] \Delta W_{opt,k} .
\end{aligned} \tag{D113}$$

This also can be arranged into a form like (64), by applying the manipulations used in equations (76)-(80) of the thesis. That yields essentially equation (80) with W replaced by Z and τ_k by v_k , plus the last two terms of (D113), each premultiplied by

$$\frac{1}{a_0} \left[I - \frac{1}{p_k} Y_k Y_k^H \right] ,$$

where p_k is defined in Tables 5 and 6 of the thesis, but will be replaced by its time-average in the final form before averaging everything else, for tractability, i.e.

$$\begin{aligned}
p_k - \theta &= a_0/b_0 + \mathcal{E}\{Y_k^H Y_k\} = \\
&= \frac{\tau_2(\tau_1 + \gamma)}{g} \left(\frac{q}{q + \tau_2} \right) + Tr[R] .
\end{aligned} \tag{D114}$$

Explicitly, the result is

$$\begin{aligned}
Z_k &= \left[\left(\frac{a_1}{a_0} \right) I + \left(\frac{b_1/b_0 - a_1/a_0}{\theta} \right) Y_k Y_k^H \right] Z_{k-1} - \\
&- \left(\frac{\tau_1 \tau_2}{a_0} \right) \left[I - \frac{1}{\theta} Y_k Y_k^H \right] Z_{k-2} - \\
&+ \left(\frac{g}{b_0} \right) \frac{1}{\theta} Y_k v_k^* - \\
&- \left(\frac{\tau_1 \tau_2}{a_0} \right) \left[I - \frac{1}{\theta} Y_k Y_k^H \right] \Delta^2 W_{opt,k} - \left(\frac{\tau_2}{a_0} \right) \left[I - \frac{1}{p_k} Y_k Y_k^H \right] \left[\gamma I + \frac{g}{q} Y_k Y_k^H \right] \Delta W_{opt,k} .
\end{aligned} \tag{D115}$$

with $a_0 = \tau_2(\tau_1 + \gamma)$, $b_0 = g(1 + \tau_2/q)$, $a_1 = 2\tau_2(\tau_1 + \gamma/2)$, $b_1 = g\tau_2/q$. Using (D42) it is assumed that the lag error and acceleration error are independent of the stationary

emse, the data, noise and everything else. With some hindsight to the NLMS tracking error development, the only new terms that contribute to the *mse* and which need to be derived are those dependent on the last two terms of (D115) above, which simplify to what will be denoted by L_k as

$$L_k = - \left(\frac{\tau_1 \tau_2}{a_0} \right) \left[I - \frac{1}{\theta} Y_k Y_k^H \right] \Delta^2 W_{opt,k} - \left(\frac{\tau_2}{a_0} \right) \left[\gamma I + \frac{g}{q p_k} \left(p_k - \frac{\gamma q}{g} - Y_k^H Y_k \right) Y_k Y_k^H \right] \Delta W_{opt,k} . \quad \dots(D116)$$

where $p_k - Y_k^H Y_k = a_0/b_0$. With the understanding that $\rho = 1/2$ one may use equations (D44)-(D47) and define

$$C_k \doteq - \left(\frac{\tau_1 \tau_2}{a_0} \right) \left[I - \frac{1}{\theta} Y_k Y_k^H \right] \quad \dots(D117)$$

and

$$D_k \doteq - \left(\frac{\gamma \tau_2}{a_0} \right) \left[\gamma I + \frac{g}{\gamma q \theta} \left(\frac{a_0}{b_0} - \frac{\gamma q}{g} \right) Y_k Y_k^H \right] \quad \dots(D118)$$

to simplify the complete weight-vector error recursion to

$$Z_k = A_k Z_{k-1} + B_k Z_{k-2} + c_k Y_k v_k^* + C_k \Delta^2 W_{opt,k} + D_k \Delta W_{opt,k} \quad \dots(D119)$$

The same process as was used in reducing (D48) can now be applied to (D119) with the aforementioned assumptions. That leads to the weight-error covariance matrix (the FRQN counterpart of (D96))

$$\begin{aligned}
H_k = & \left(\frac{a_1}{a_0} \right)^2 \left[H_{k-1} + z(H_{k-1}R + RH_{k-1}) + z^2(RTr[RH_{k-1}] + RH_{k-1}R) \right] + \\
& + \left(\frac{a_2}{a_0} \right)^2 \left[H_{k-2} + x(H_{k-2}R + RH_{k-2}) + x^2(RTr[RH_{k-2}] + RH_{k-2}R) \right] + \\
& + \left(\frac{-a_1 a_2}{a_0^2} \right) 2Re[H_{k1,2} + (xH_{k1,2}R + zRH_{k1,2}) + zx(RTr[RH_{k1,2}] + RH_{k1,2}R)] + \\
& + \left(\frac{g}{\theta b_0} \right)^2 Rmmse + \\
& + \mathcal{E}\{L_k L_k^H\} .
\end{aligned} \tag{D120}$$

It remains to simplify the last term in (D120), where

$$L_k = C_k \Delta^2 W_{opt,k} + D_k \Delta W_{opt,k} . \tag{D121}$$

Proceeding,

$$\begin{aligned}
\mathcal{E}\{L_k L_k^H\} = & \mathcal{E}\{C_k \Delta^2 W_{opt,k} \Delta^2 W_{opt,k}^H C_k^H\} + \mathcal{E}\{D_k \Delta W_{opt,k} \Delta W_{opt,k}^H D_k^H\} + \\
& + \mathcal{E}\{C_k \Delta^2 W_{opt,k} \Delta W_{opt,k}^H D_k^H\} + \mathcal{E}\{D_k \Delta W_{opt,k} \Delta^2 W_{opt,k}^H C_k^H\} .
\end{aligned} \tag{D122}$$

The results in [31] will again be invoked similarly as they were in (D52)-(D53), but in addition it will be necessary to simplify the following , using the results (D100)-(D104) of the corresponding NLMS analysis:

$$\boxed{\mathcal{E}\{\Delta W_{opt,k} \Delta W_{opt,k}^H\} = \left(\frac{2\sigma_\varphi^2}{1 + \Phi} \right) I = f(\varphi, \Phi) I} \tag{D123}$$

Then, since

$$\Delta^2 W_{opt,k} = \Delta W_{opt,k} - \Delta W_{opt,k-1} = (\Phi - 1) \Delta W_{opt,k-1} + \varphi_k - \varphi_{k-1} \quad \dots (D124)$$

one may similarly approximate, for weakly non-stationary weights,

$$\begin{aligned} \mathcal{E}\{\Delta^2 W_{opt,k} \Delta^2 W_{opt,k}^H\} &= (\Phi - 1)^2 \mathcal{E}\{\Delta W_{opt,k-1} \Delta W_{opt,k-1}^H\} + \mathcal{E}\{(\varphi_k - \varphi_{k-1})(\varphi_k^H - \varphi_{k-1}^H)\} \\ &\quad + 2(\Phi - 1) \text{Re} \mathcal{E}\{\Delta W_{opt,k-1}(\varphi_k^H - \varphi_{k-1}^H)\} \approx \\ &\approx \left(\frac{(\Phi - 1)^2 2\sigma_\varphi^2}{1 + \Phi} \right) I + 2\sigma_\varphi^2 I + 2(\Phi - 1) \text{Re}[(1 - \Phi)\sigma_\varphi^2 I] \approx \\ &\approx \left(\frac{2\sigma_\varphi^2}{1 + \Phi} \right) (1 + 2\Phi^2 - \Phi^3) I \doteq g(\varphi, \Phi) I, \quad \dots (D125) \end{aligned}$$

since φ_k is a white-noise process in which $\mathcal{E}\{\varphi_k \varphi_{k-1}^H\} = 0$ and use was made of (D104) or (D123) above. The AR property discussed under (D101) was invoked to simplify the term

$$\begin{aligned} \mathcal{E}\{(W_{opt,k} - W_{opt,k-1})(\varphi_k^H - \varphi_{k-1}^H)\} &= \\ = \mathcal{E}\{W_{opt,k} \varphi_k^H\} - \mathcal{E}\{W_{opt,k} \varphi_{k-1}^H\} - [0] + \mathcal{E}\{W_{opt,k-1} \varphi_{k-1}^H\} &\approx \\ \approx 2\mathcal{E}\{W_{opt,k} \varphi_k^H\} - \mathcal{E}\{(\Phi W_{opt,k-1} + \varphi_{k-1}) \varphi_{k-1}^H\} &\approx \dots (D126) \\ \approx 2\mathcal{E}\{(\Phi W_{opt,k-1} + \varphi_k) \varphi_k^H\} - \mathcal{E}\{(\Phi W_{opt,k-1} \varphi_{k-1}^H + \varphi_{k-1} \varphi_{k-1}^H)\} &\approx \\ \approx 2\sigma_\varphi^2 I - \Phi(\sigma_\varphi^2 I) - \sigma_\varphi^2 I \approx (1 - \Phi)\sigma_\varphi^2 I. \end{aligned}$$

Finally, the lag and acceleration error cross-covariance terms may be approximated using (D123) and (D100) as

$$\begin{aligned} \mathcal{E}\{\Delta W_{opt,k} \Delta^2 W_{opt,k}^H\} &= \mathcal{E}\{\Delta W_{opt,k} (\Delta W_{opt,k}^H - \Delta W_{opt,k-1}^H)\} \approx \\ &\approx \left(\frac{2\sigma_\varphi^2}{1 + \Phi} \right) I - \mathcal{E}\{\Delta W_{opt,k} \Delta W_{opt,k-1}^H\} \approx \dots (D127) \\ &\approx \left(\frac{2\sigma_\varphi^2}{1 + \Phi} \right) I - \mathcal{E}\{[(\Phi - 1) W_{opt,k-1} + \varphi_k][(\Phi - 1) W_{opt,k-2}^H + \varphi_{k-1}^H]\} \end{aligned}$$

The AR property of $W_{opt,k-1}$ is again invoked, since it depends on φ_{k-1} according to (90). Then

$$\begin{aligned}
& \mathcal{E}\{[(\Phi - 1)W_{opt,k-1} + \Phi_k][(\Phi - 1)W_{opt,k-2}^H + \Phi_{k-1}^H]\} = \\
& = (\Phi - 1)^2 \mathcal{E}\{W_{opt,k-1} W_{opt,k-2}^H\} + (\Phi - 1) \mathcal{E}\{W_{opt,k-1} \Phi_{k-1}^H\} = \\
& = (\Phi - 1)^2 \mathcal{E}\{\Phi W_{opt,k-2} W_{opt,k-2}^H + \Phi_{k-1} W_{opt,k-2}^H\} + (\Phi - 1) \sigma_\Phi^2 I = \dots (D128) \\
& = \frac{(\Phi - 1)^2 \sigma_\Phi^2}{1 - \Phi^2} I + [0] + (\Phi - 1) \sigma_\Phi^2 I = \\
& = \sigma_\Phi^2 \Phi \left(\frac{1 - \Phi}{1 + \Phi} \right) I.
\end{aligned}$$

Therefore

$$\mathcal{E}\{\Delta W_{opt,k} \Delta^2 W_{opt,k}^H\} = \left(\frac{\sigma_\Phi^2}{1 + \Phi} \right) (2 - \Phi + \Phi^2) I + h(\varphi, \Phi) I. \quad \dots (D129)$$

Now the results (D123), (D125) and (D129) are used in the result from [31] as before and applied to the terms in (D122) to obtain the following:

$$\begin{aligned}
& \mathcal{E}\{C_k \Delta^2 W_{opt,k} \Delta^2 W_{opt,k}^H C_k^H\} = \dots (D130) \\
& = \left(\frac{\tau_1 \tau_2}{a_0} \right)^2 \left[Ig(\varphi, \Phi) - \frac{g(\varphi, \Phi)R + Rg(\varphi, \Phi)}{\theta} + \frac{RTr[Rg(\varphi, \Phi)] + Rg(\varphi, \Phi)R}{\theta^2} \right],
\end{aligned}$$

$$\begin{aligned}
& \mathcal{E}\{D_k \Delta W_{opt,k} \Delta W_{opt,k}^H D_k^H\} = \dots (D131) \\
& = \left(\frac{\gamma \tau_2}{a_0} \right)^2 [If(\varphi, \Phi) + \Omega(f(\varphi, \Phi)R + Rf(\varphi, \Phi)) + \Omega^2(RTr[Rf(\varphi, \Phi)] + Rf(\varphi, \Phi)R)]
\end{aligned}$$

where

$$\Omega = \frac{g}{\gamma q \theta} \left(\frac{a_0}{b_0} - \frac{\gamma q}{g} \right), \quad \dots (D132)$$

and

$$\begin{aligned}
& \mathcal{E}\{D_k \Delta W_{opt,k} \Delta^2 W_{opt,k}^H C_k^H\} = \dots (D133) \\
& = \frac{\gamma \tau_1 \tau_2^2}{a_0^2} \left[Ih(\varphi, \Phi) + \left(\Omega h(\varphi, \Phi)R - \frac{Rh(\varphi, \Phi)}{\theta} \right) - \frac{\Omega(RTr[Rh(\varphi, \Phi)] + Rh(\varphi, \Phi)R)}{\theta} \right]
\end{aligned}$$

(The above terms were modelled after (D54), (D55) and (D59).) With the terms expanded as above, the "diagonalizing" procedure may be applied to (D122). Because the functions f, g, h of (φ, Φ) are scalars, the diagonalization concerns mostly matrix R . First the terms in the said three equations are collected and rearranged so the common factors are I, Λ and $\Lambda Tr[\Lambda] + \Lambda^2$, as

$$\begin{aligned} F_k &= E^H \mathcal{E} \{ L_k L_k^H \} E = \left(g(\varphi, \Phi) \left(\frac{\tau_1 \tau_2}{a_0} \right)^2 + f(\varphi, \Phi) \left(\frac{\gamma \tau_2}{a_0} \right)^2 - 2h(\varphi, \Phi) \left(\frac{\gamma \tau_1 \tau_2^2}{a_0^2} \right) \right) I + \\ &+ 2 \left(\frac{-g(\varphi, \Phi)}{\theta} \left(\frac{\tau_1 \tau_2}{a_0} \right)^2 + \Omega f(\varphi, \Phi) \left(\frac{\gamma \tau_2}{a_0} \right)^2 + \left(\frac{1}{\theta} - \Omega \right) h(\varphi, \Phi) \left(\frac{\gamma \tau_1 \tau_2^2}{a_0^2} \right) \right) \Lambda + \dots (D134) \\ &+ \left(g(\varphi, \Phi) \left(\frac{\tau_1 \tau_2}{a_0 \theta} \right)^2 + f(\varphi, \Phi) \left(\frac{\gamma \tau_2 \Omega}{a_0} \right)^2 + h(\varphi, \Phi) \left(\frac{\gamma \tau_1 \tau_2^2}{a_0^2} \right) \left(\frac{2\Omega}{\theta} \right) \right) [\Lambda Tr[\Lambda] + \Lambda^2]. \end{aligned}$$

Note that at this stage, the evolving tracking error is part of the $emse_k$ and is given by

$$tmse_k = Tr[\Lambda F_k] = \mathcal{E}^H \mathcal{F}_k \quad \dots (D135)$$

where $\mathcal{F}_k = diag[F_k]$ (a vector). In this development only the steady-state tracking error under the assumption of a weakly non-stationary model system in an adaptive system-identifier is of interest. Recall that it was assumed that the $tmse_{\infty}$ is totally independent of the $emse_{\infty}$; it is computed using (D135) for which purpose $\mathcal{F}_{\infty}$ is obtained using the same method as was used to obtain N_{∞} in (D68) for the stationary case, so that

$$tmse_{\infty} = \mathcal{E}^H [I - (hI + f\Lambda)^2 - f^2 \mathcal{E} \mathcal{E}^H]^{-1} \left[\lim_{k \rightarrow \infty} \mathcal{F}_k \right] \quad \dots (D136)$$

where h and f are the constants defined in equations (D66) and (D67). To relieve the notational fatigue, the scalar functions of φ, Φ and algorithm parameters (algo.) in each

of the 3 rows of (D134) above will be temporarily abbreviated so that (D135) may be "expanded" as

$$\begin{aligned} \mathcal{F}_k = \text{diag}[F_k] = & f_1(\varphi, \Phi, \text{algo.})J + f_2(\varphi, \Phi, \text{algo.})\mathcal{L} + \\ & + f_3(\varphi, \Phi, \text{algo.})[\mathcal{L}\mathcal{L}^H J + \Lambda^2 J], \end{aligned} \quad \dots(\text{D137})$$

where one will recall that J is a vector of "1"s. The RHS does not change as $k \rightarrow \infty$. The matrix inverse in (D136) is available from (D68)-(D70). With this equipment (D136) can be expanded to yield the tracking error in a "friendly" form as

$$\begin{aligned} \text{tmse}_\infty = & f_1(\varphi, \Phi, \text{algo.})\mathcal{L}^H [D - f^2 \mathcal{L} \mathcal{L}^H]^{-1} J + f_2(\varphi, \Phi, \text{algo.})\mathcal{L}^H [D - f^2 \mathcal{L} \mathcal{L}^H]^{-1} \mathcal{L} + \\ & + f_3(\varphi, \Phi, \text{algo.})\mathcal{L}^H [D - f^2 \mathcal{L} \mathcal{L}^H]^{-1} [\mathcal{L} \mathcal{L}^H J + \Lambda^2 J], \end{aligned} \quad \dots(\text{D138})$$

where f is defined in (D67) and ultimately in (D77), and D in (D69). Expanding (D138) along the lines of (D66)-(D79) produces

$$\begin{aligned} \text{tmse}_\infty = & f_1(\varphi, \Phi, \text{algo.}) \frac{\sum_{n=1}^N \frac{1/p}{2 - p\lambda_n}}{1 - \sum_{n=1}^N \frac{p\lambda_n}{2 - p\lambda_n}} + \\ & + f_2(\varphi, \Phi, \text{algo.}) \frac{\sum_{n=1}^N \frac{p\lambda_n}{2 - p\lambda_n}}{1 - \sum_{n=1}^N \frac{p\lambda_n}{2 - p\lambda_n}} + \\ & + f_3(\varphi, \Phi, \text{algo.}) \left[\left(\sum_{n=1}^N \lambda_n \right) \frac{\sum_{n=1}^N \frac{p\lambda_n}{2 - p\lambda_n}}{1 - \sum_{n=1}^N \frac{p\lambda_n}{2 - p\lambda_n}} + \frac{\frac{1}{p} \sum_{n=1}^N \frac{\lambda_n^2}{2 - p\lambda_n}}{1 - \sum_{n=1}^N \frac{p\lambda_n}{2 - p\lambda_n}} \right] \end{aligned} \quad \dots(\text{D139})$$

where one will recall that p has to be specialized to the case of $p = 1/2$ when making

use of the results from the stationary analysis. The scalar functions of φ, Φ and algorithm parameters (algo.) can be reduced by substitutions to the following forms:

$$\begin{aligned} f_1(\varphi, \Phi, \text{algo.}) &= g(\varphi, \Phi) \left(\frac{\tau_1 \tau_2}{a_0} \right)^2 + f(\varphi, \Phi) \left(\frac{\gamma \tau_2}{a_0} \right)^2 - 2h(\varphi, \Phi) \left(\frac{\gamma \tau_1 \tau_2^2}{a_0^2} \right) = \dots \\ &\dots = \frac{2\sigma_\varphi^2}{1+\Phi} \left(\frac{1}{\tau_1 + \gamma} \right)^2 \left[(\tau_1 - \gamma)^2 + \Phi \gamma \tau_1 + \Phi^2 \tau_1 (2\tau_1 - \gamma) + \Phi^3 \tau_1^2 \right], \end{aligned} \quad \dots(\text{D140})$$

$$\begin{aligned} f_2(\varphi, \Phi, \text{algo.}) &= 2 \left(\frac{-g(\varphi, \Phi)}{\theta} \left(\frac{\tau_1 \tau_2}{a_0} \right)^2 + \Omega f(\varphi, \Phi) \left(\frac{\gamma \tau_2}{a_0} \right)^2 + \left(\frac{1}{\theta} - \Omega \right) h(\varphi, \Phi) \left(\frac{\gamma \tau_1 \tau_2^2}{a_0^2} \right) \right) \\ &\dots = \frac{2\sigma_\varphi^2}{1+\Phi} \left(\frac{1}{\tau_1 + \gamma} \right)^2 \left[(\tau_1 - \gamma) \left[(\tau_1 - \gamma) \left(\Omega - \frac{1}{\theta} \right) - (\tau_1 + \gamma) \left(\Omega + \frac{1}{\theta} \right) \right] + \right. \\ &\quad \left. + \Phi \gamma \tau_1 \left(\Omega - \frac{1}{\theta} \right) - \Phi^2 \tau_1 \left(4 \frac{\tau_1}{\theta} + \gamma \left(\Omega - \frac{1}{\theta} \right) \right) + \Phi^3 \tau_1^2 \left(\frac{2}{\theta} \right) \right] \quad \dots(\text{D141}) \end{aligned}$$

where some of the remaining parameter expressions are later approximated as

$$P = \frac{g}{\tau_2(\tau_1 + \gamma) + g(1 + \tau_2/q)Tr[R]} \approx \frac{q}{q + \tau_2} \frac{1}{Tr[R]}, \quad \dots(\text{D142})$$

$$\Omega - \frac{1}{\theta} = \frac{1}{\theta} \left[\left(\frac{\tau_1 - \gamma}{\gamma} \right) \left(\frac{\tau_2}{\tau_2 + q} \right) - 2 \left(\frac{q}{\tau_2 + q} \right) \right] \approx \frac{\tau_2(\tau_1/\gamma - 1) - 2q}{(\tau_2 + q)Tr[R]}, \quad (\text{D143})$$

$$\frac{1}{\theta} = \frac{g}{\tau_2(\tau_1 + \gamma) \left(\frac{q}{\tau_2 + q} \right) + gTr[R]} \approx \frac{1}{Tr[R]}, \quad \dots(\text{D144})$$

and finally

$$\begin{aligned}
f_3(\varphi, \Phi, algo.) &= g(\varphi, \Phi) \left(\frac{\tau_1 \tau_2}{a_0 \theta} \right)^2 + f(\varphi, \Phi) \left(\frac{\gamma \tau_2 \Omega}{a_0} \right)^2 + h(\varphi, \Phi) \left(\frac{\gamma \tau_1 \tau_2^2}{a_0^2} \right) \left(\frac{2\Omega}{\theta} \right) = \dots \\
\dots &= \frac{2\sigma_\varphi^2}{1+\Phi} \left(\frac{1}{\tau_1 + \gamma} \right)^2 \left[\left(\frac{\tau_1}{\theta} + \gamma \Omega \right)^2 - \Phi \gamma \tau_1 \left(\frac{\Omega}{\theta} \right) + \Phi^2 \tau_1 \left(\frac{1}{\theta} \right) \left(\frac{2\tau_1}{\theta} + \Omega \gamma \right) - \Phi^3 \tau_1^2 \left(\frac{1}{\theta} \right)^2 \right] \\
&\dots (D145)
\end{aligned}$$

In the approximations it was assumed that g is sufficiently large to satisfy equation (116) of the thesis. When one makes the additional assumption that $p \ll 2/\lambda_{\max}$, as in the stationary analysis, the summations over n in (D139) simplify to expressions involving $Tr[R]$, all of which reduces equation (D139) to the form

$$\begin{aligned}
mse_{\infty} &= \frac{2\sigma_\varphi^2 p^{-1} \left(\frac{1}{\tau_1 + \gamma} \right)^2}{1 - p Tr[R]/2} \left[\left[(\tau_1 - \gamma)^2 + \Phi \gamma \tau_1 + \Phi^2 \tau_1 (2\tau_1 - \gamma) + \Phi^3 \tau_1^2 \right] \frac{N}{2} + \right. \\
&+ \left[(\tau_1 - \gamma) \left[(\tau_1 - \gamma) \left(\Omega - \frac{1}{\theta} \right) - (\tau_1 + \gamma) \left(\Omega + \frac{1}{\theta} \right) \right] + \right. \\
&+ \left. \Phi \gamma \tau_1 \left(\Omega - \frac{1}{\theta} \right) - \Phi^2 \tau_1 \left(4 \frac{\tau_1}{\theta} + \gamma \left(\Omega - \frac{1}{\theta} \right) \right) + \Phi^3 \tau_1^2 \left(\frac{2}{\theta} \right) \right] \frac{p^2}{2} Tr[R] + \\
&+ \left. \left[\left(\frac{\tau_1}{\theta} + \gamma \Omega \right)^2 - \frac{\Phi \gamma \tau_1 \Omega}{\theta} + \frac{\Phi^2 \tau_1}{\theta} \left(\frac{2\tau_1}{\theta} + \Omega \gamma \right) - \Phi^3 \tau_1^2 \left(\frac{1}{\theta} \right)^2 \right] \left[\frac{(p Tr[R])^2}{2} + Tr[R^2] \right] \right] \\
&\dots (D146)
\end{aligned}$$

This expression for the steady-state tracking error of the FRQN algorithm is yet too cumbersome for operational use, but is already showing characteristics similar to those of the NLMS tracking error (D109): Again p corresponds to β and the tracking error is inversely proportional to it. Equation (D142) allows simplification of the denominator in the common factor of the RHS of (D146) and, since $\Omega \pm 1/\theta$ and $1/\theta$ are all inversely proportional to $Tr[R]$, expression (D146) reduces still further to

$$\begin{aligned}
 mse = & \frac{2\sigma_p^2(q + \tau_2) \left(\frac{\tau_1}{\tau_1 + \gamma} \right)^2 \sum_{n=1}^N \lambda_n \left\{ \frac{N}{2} \left[\left(1 - \frac{\gamma}{\tau_1} \right)^2 + \frac{\Phi \gamma}{\tau_1} + \Phi^2 \left(2 - \frac{\gamma}{\tau_1} \right) + \Phi^3 \right] + \right. \\
 & \left. + \frac{p^2}{2} \left[\left(1 - \frac{\gamma}{\tau_1} + \frac{\tau_2}{\tau_2 + q} \right)^2 - 2 \left(1 - \frac{\gamma}{\tau_1} \right) - \frac{\Phi \gamma}{\tau_1} \left(\frac{q}{\tau_2 + q} \right) - \Phi^2 \left(2 - \frac{\gamma/\tau_1}{\tau_2 + q} \right) + \Phi^3 \right] + \right. \\
 & \left. + \frac{Tr[R^2]}{(Tr[R])^2} \left[\left(1 - \frac{\gamma}{\tau_1} + \frac{\tau_2}{\tau_2 + q} \right)^2 + \frac{\Phi \gamma}{\tau_1} \left(1 - \frac{\tau_2 \tau_1 / \gamma}{\tau_2 + q} \right) + \Phi^2 \left(2 - \frac{\gamma}{\tau_1} + \frac{\tau_2}{\tau_2 + q} \right) - \Phi^3 \right] \right\} \quad \dots (D147)
 \end{aligned}$$

where p was expanded in the common factor (the first factor on the RHS). To compare with the corresponding result (D109) for NLMS, note that p^{-1} is about τ_2/q times larger than β^{-1} . Given that $\tau_2 > q$ and $0 < \Phi < 1$, then for the (empirically-obtained) small value of the leakage term $\gamma = 1/\tau_2$, the FRQN term proportional to N is almost $2\tau_2/q$ times larger than the mse_{NLMS} of the NLMS algorithm for this kind of non-stationarity. As the other terms in (D147) are all > 0 , the FRQN tracking error should be considerably greater than that of NLMS, yet the BER results in Figures 22 and 23 for both algorithms in a fading mobile-radio scenario with an adaptive array show that the opposite is in fact happening. Equation (D147) is equation (108) of the thesis body. Figures 24 and 25 do however show that the FRQN tracking error eventually causes the total mse to exceed that of NLMS, though at unusually high fade rates for that application.

With the above-mentioned value of the leakage term $\gamma = 1/\tau_2$, the assumptions $\tau_2 \gg \{\tau_1, q\} > 1$, expression (D147) may be further approximated as

$$\begin{aligned}
mse_{\gamma} = & \frac{2\sigma_p^2 p^{-1}}{1+\Phi} \left\{ \frac{N}{2} [1 + 2\Phi^2 + \Phi^3] + \right. \\
& + \frac{p^2}{2} [2 - 2\Phi^2 + \Phi^3] + \\
& \left. + \frac{Tr\{R^2\}}{(Tr\{R\})^2} [4 - \Phi + 3\Phi^2 - \Phi^3] \right\}. \quad \dots (D148)
\end{aligned}$$

It may also seem that the FRQN tracking error as given by (D147) might be reduced by a different choice of the leakage term. For example, $\gamma = \tau_1$ reduces the FRQN term proportional to N by a factor of $3/16$, eliminates the term proportional to p (middle line of (D147)) and reduces the last term by $1/10$. Simulations showed a slight improvement of tracking error (BER in the ill-conditioned scenario with the highest SNR decreased slightly, from 0.0610 to 0.0467), the transient response in that scenario was corrupted to one that was worse-behaved than that of NLMS. Clearly, this is not a good model of the typical kind of non-stationarity encountered in the most useful type of application of the FRQN algorithm, namely an adaptive antenna array in a fading mobile-radio scenario with frequent occurrence of closely-spaced mobile emitters.

For completeness, it may be noted that expression (D147) gives the tracking error of the "ordinary second-order LMS" [26] when one substitutes in it $\gamma = 1$ and lets $q \rightarrow \infty$. That also mostly increases each line in magnitude (by factors of 2, -4, 4/5, for the first-, second- and third-line terms, respectively). Therefore its tracking error is not likely to be smaller even in the scenarios of interest herein. Simulations of that algorithm occasionally diverged even in stationary scenarios, so it was not pursued further in this work.

Appendix E: The Matrix-Inversion Lemma

Because many of the key results, including the FRQN algorithm itself, are derived with the help of this identity, a non-rigorous sketch of its derivation is included in this Appendix for completeness. The author is indebted to his brother, Dr. Ivo Klemes (Department of Mathematics, McGill University) for having pointed the way towards deriving the various versions of this identity some years ago.

It is required to derive what is also known as the Woodbury identity, from which the other forms follow as special cases, i.e.

$$[A + BCD]^{-1} = A^{-1} - A^{-1}B[DA^{-1}B + C^{-1}]^{-1}DA^{-1} \quad \dots(E1)$$

where matrix A is $N \times N$, C is $P \times P$, A^{-1} and C^{-1} exist, B is $N \times P$ and D is $P \times N$. Without loss of generality, assume that $P < N$. The usefulness of this identity is appreciated when for example A^{-1} is known (e.g. it is diagonal), so it is required only to compute the $P \times P$ inverse of C , which is considerably less laborious. In some fortuitous special cases C is a scalar and B and D are vectors.

It is useful to identify some generalized identity operators which apply in the subspace spanned by C (referred to as its rank-space herein) and associated pseudoinverses [E1]. The pseudoinverse of B is denoted by $^+$ and is given by

$$B^+ = B^H(BB^H)^{-1} \quad \dots(E2)$$

so that

$$BB^+ = I_{P \times P} \quad \dots(E3)$$

is the $P \times P$ identity and the generalized identity operator in the span of B is the projection operator that projects any vector onto that subspace (i.e. parallel to it), i. e.

$$P_B^1 = B^* B \quad \dots(E4)$$

which is also assumed to be the same subspace spanned by D and C . Similarly, the pseudoinverse of D is given by

$$D^* = (D^H D)^{-1} D^H \quad \dots(E5)$$

so that

$$D^* D = I_{P \times P} \quad \dots(E6)$$

and the generalized identity operator associated with it is the projection onto the span of D , i.e.

$$P_D^1 = D D^* \quad \dots(E7)$$

Generalized identity means that any vector within its associated subspace (i. e. span of B or D) will remain unchanged when multiplied by it, since it simply projects onto that subspace. Recall that all projection operators are also idempotent and symmetric.

With the above in mind, proceed to factor $[A + BCD]^{-1}$ as follows:

$$\begin{aligned} [A + BCD]^{-1} &= [A(I + A^{-1}BCD)]^{-1} = [A((BCD)^{-1} + A^{-1})BCD]^{-1} \quad \dots(E8) \\ &= (BCD)^{-1}[(BCD)^{-1} + A^{-1}]^{-1}A^{-1} \end{aligned}$$

Now add and subtract $A^{-1}[(BCD)^{-1} + A^{-1}]^{-1}A^{-1}$ on the RHS and rearrange as

$$[A + BCD]^{-1} = [(BCD)^{-1} + A^{-1}][(BCD)^{-1} + A^{-1}]^{-1}A^{-1} - A^{-1}[(BCD)^{-1} + A^{-1}]^{-1}A^{-1}$$

to obtain

$$[A + BCD]^{-1} = A^{-1} - A^{-1}[(BCD)^{-1} + A^{-1}]^{-1}A^{-1} \quad \dots(E9)$$

which may be one useful form. To generalize further, expand $(BCD)^{-1}$. Postulate that it is given by $D^*C^{-1}B^*$ because when multiplied together in either order, the generalized identities allow simplification of the results to the required $P \times P$ identity,

as follows:

$$\begin{aligned}
 (D^* C^{-1} B^*)(BCD) &= D^* C^{-1} (B^* B) CD \\
 &= D^* C^{-1} P_B^1 CD = D^* C^{-1} CD \quad \dots(E10) \\
 &= D^* D = I_{P \times P}
 \end{aligned}$$

because $P_B^1 C = C$ and similarly, because also $C P_D^1 = C$, one has

$$\begin{aligned}
 (BCD)(D^* C^{-1} B^*) &= B C P_D^1 C^{-1} B^* = B C C^{-1} B^* \quad \dots(E11) \\
 &= B B^* = I_{P \times P}
 \end{aligned}$$

Therefore one may factor out the pseudoinverses as

$$\begin{aligned}
 [A + BCD]^{-1} &= A^{-1} - A^{-1} [D^* C^{-1} B^* + A^{-1}]^{-1} A^{-1} \quad \dots(E12) \\
 &= A^{-1} - A^{-1} [D^* (D D^* C^{-1} B^* B + D A^{-1} B) B^*]^{-1} A^{-1} .
 \end{aligned}$$

The generalized identities also leave C^{-1} unaffected and inverses of the pseudoinverses are the original matrices B and D , so the above result simplifies as

$$\begin{aligned}
 [A + BCD]^{-1} &= A^{-1} - A^{-1} (D^* [C^{-1} + D A^{-1} B] B^*)^{-1} A^{-1} \\
 &= A^{-1} - A^{-1} B [C^{-1} + D A^{-1} B]^{-1} D A^{-1} . \quad \dots(E13)
 \end{aligned}$$

In the algorithm derivations C is a scalar and $B = Y_k^*$, $D = B^H$. A is either assumed known (RLS) or is diagonal; in either case its inverse is readily available.

Appendix F: Simulated Scenarios:

Generalized Side-Lobe Canceller with FRQN and NLMS Algorithms:

Initial simulations were carried out using a simple generalized sidelobe canceller with desired signal absent. Three independent interferers' signals were incident on a 4-element array in two extreme scenarios as shown in Figures F1 and F2 below. All sources were stationary; multipath and mutual coupling were ignored. All signals were modelled in their baseband equivalent forms using the narrowband assumption. The two scenarios were designed to represent the extremes of numerical difficulty for the algorithms in the sense of condition number of the array-covariance matrix, or equivalently, the disparity of its eigenvalues. The well-conditioned scenario had an eigenvalue spread of about 10 whereas the ill-conditioned one had about 10 000. Each scenario also demanded each algorithm to use all of its degrees of freedom to suppress the interference (although fewer than three nulls might suffice to spatially filter out the three interferences in the ill-conditioned scenario).

Both the FRQN and NLMS algorithms were used to adapt the array weights in these scenarios; their parameters and relative performance are discussed in section 3.3. The desired signal was actually absent from the simulations so as not to obscure the output power which served as an indicator of convergence in the (mean) square. Its bearing would normally be known and correspond to 0 degrees azimuth in this kind of application. To a first-order approximation, its presence in practice would not affect the system performance because each auxiliary beam of the beamformer is designed to have a null in its direction. To keep these initial simulations simple, pointing errors, non-ideal propagation conditions and imperfect array and beamformer construction were ignored, only about -60 dB of uncorrelated receiver noise was added to the received array signals.

These scenarios were considered to be reasonably representative of practical situations in applications such as land mobile-radio telephony, in that they covered the variations in mathematical condition, yet were not unduly contrived. (For example, the well-conditioned scenario might represent direct-path co-channel interference from frequency re-use cells which are relatively widely-separated in azimuth, being embedded in a regular hexagonal grid. The ill-conditioned scenario might represent co-channel interference arriving indirectly from three re-use cells via a common large scatterer such as a group of high-rise buildings in the general direction of 20 degrees azimuth.)

Figure F1: Well-conditioned scenario for simulating generalized sidelobe-canceller performance in Figures 3(a), 3(b), 3(c) and 4(a), 4(b), 4(c).

VGSLCWELL.HPG

Figure F2: Ill-conditioned scenario for simulating generalized sidelobe-canceller performance in Figures 5(a), 5(b), 5(c) and 6(a), 6(b), 6(c).

\GSLCILL.HPG

Reference-Based FRQN and NLMS Algorithms in Stationary Scenarios:

In these simulations the same basic geometry of the far-field sources was used but the reference-direction was changed to correspond to the line of the array, and interferer #3 was rotated up by 10 degrees. The signal environment was somewhat more complex in detail, as it was intended to model a simple land mobile-radio application. A desired signal was actually present all of the time at the 90 degrees azimuth bearing (i.e. directly in front of the array), and an exact copy of it served as the reference signal for adapting the four array weights with each algorithm. Again the signals were modelled in their baseband equivalent forms and time-dispersion was neglected. Each source emitted an independent pseudo-random QPSK symbol stream synchronously, at 30 ks/s. No beam-former was used with the 4-element array, which used 4 complex weights applied to the elements. Roughly the same amount of uncorrelated thermal noise was modelled at the receivers as complex Gaussian (pseudo)random processes. Each receiver was also band-limited to 25 kHz, although these filters were not used when bit-error-rates (BER) were being counted, as they effectively caused inter-symbol interference which raised the BER, thus masking other effects of interest. In most simulations of stationary scenarios, the output a-priori errors were used as indicators of convergence in the (mean) square; they were always normalized to start at "1", corresponding to the zero initial conditions of the weights.

The only propagation effects that were modelled were the radial distance of 3 km of the sources from the array and 2-ray elevation multipath from each source, which was responsible for the inverse-fourth-power propagation loss with distance, as typically assumed in land mobile-radio environments [48]. The ground's reflection coefficient had a small component with randomized phase. The mobile sources were 2 metres high above ground and the receiving adaptive array antenna which represented the base-station was

50 metres above ground. Figures F3 and F4 depict those scenarios. Since mainly the adaptation transients were of interest, Doppler effects were not modelled; the scenarios were stationary. Eigenvalue disparity was about 28 in the well-conditioned scenario and about 1 000 000 in the ill-conditioned one.

Figure F3: Well-conditioned scenario for simulating performance of reference-based Adaptive array algorithms in Figures 7(a), 7(b), 8(a), 8(b), 11(a), and 11(b).

REFSWEL.DXF

Figure F4: Ill-conditioned scenario for simulating performance of reference-based adaptive array algorithms in Figures 9(a), 9(b), 10(a), 10(b), 12(a) and 12(b).

\\REPSFILL.DXF

Comparisons of FRQN, NLMS and RLS Adaptation Transients in Stationary Scenarios:

Transient performance of the FRQN algorithm in stationary well-conditioned and ill-conditioned scenarios was compared with that of NLMS and RLS algorithms using the same basic scenarios as before, but without filtering the received signals. Figures F5 and F6 illustrate the simulation setup pictorially.

Theoretical minimum mean squared errors were also computed for the three algorithms in these simulations. For the FRQN and NLMS algorithms, the evolving excess mean squared errors were also computed in these simulations, but not for the RLS algorithm, which optimizes a different penalty function. Again, all other features are the same as in the previous set of simulations. Although source velocities are shown, they were stationary. In these scenarios, the ground reflection coefficient ρ did not have a random component; it too was fixed.

Figure F5: Well-conditioned stationary scenario for comparing adaptation transients of FRQN, NLMS and RLS algorithms in Figures 13(a), 13(b), 14(a), 14(b), 15(a), 15(b), and 19, 20, 21, respectively.

REFSUWEL.DXF

Comparisons of FRQN, NLMS and RLS Tracking Performance in Non-Stationary Scenarios:

In these simulations the Bit Error Rate (BER) relative to the desired source was used as a measure of the tracking performance of the FRQN, NLMS and RLS algorithm in the same two scenarios, but with all sources moving as indicated in Figures F7 and F8. Appropriate Doppler fading modulations were applied to the signals at their sources. Their speeds were used to calculate the Doppler frequencies, which became the cut-off frequencies of the 5-th order Chebyshev low-pass filters that were used to filter independent pseudorandom Gaussian white noise sequences to create the fading modulations. The same sets of complex pseudorandom sequences were used in each run, with each algorithm.

Tracking performance versus Signal to Noise Ratio (BER vs. SNR) was simulated by varying the noise powers added in front of the receivers to cause the SNR to take on values of approximately 60, 30, 18 and 11 dB (calculated on the desired received signal), with all source velocities held fixed at the values and directions indicated in the Figures.

Tracking performance versus fade rate (BER vs. Doppler) was simulated by varying the speed of *only* the desired-signal source over the values 30, 60, 120 and 240 km/h, while keeping its SNR fixed at the 30 dB value. (Therefore the set of points corresponding to desired-source velocity of 60 km/h and SNR of 30 dB is common among Figures 22 and 24, and among Figures 23 and 25).

Figure F7: Well-conditioned non-stationary scenario to simulate tracking performance of NLMS, FRQN, RLS algorithms versus SNR in Figures 22, 23 and Doppler in Figures 24, 25.

REFMUNWHL.DXF

Figure F8: III-conditioned non-stationary scenario to simulate tracking performance of NLMS, FRQN, RLS algorithms versus SNR in Figures 22, 23 and Doppler in Figures 24, 25.

REFMULL.DXF

Appendix G: Requirements of Reference-Generating Subsystems Operating with NLMS and FRQN Algorithms

This Appendix deals with so-called internal reference-generating techniques and how they operate with feedback (recursive) adaptive algorithms so that both the algorithm and reference-generator are stable. The reference signal is assumed to be generated from the output of the adaptive array processor and fed back to compute the *a-priori* error for adaptation. Therefore the reference-generating path perturbs the original feedback structure of the adaptive algorithm. Although it is generally a non-linear, signal-selective subsystem, the reference-generator will be treated only in its linear region of operation which affects reference acquisition. In its steady state, it does not affect the stability of the adaptive algorithm.

All the internal reference-generating techniques may be modelled by a nonlinear system which provides some gain, r , to the component of its input, y_k , which is 100% correlated with the SOI and some smaller gain, ε , to those which are not. Consequently its output, r_k , is a better estimate of the SOI than its input. In addition, such a system provides a means to limit the amplitude of its output when operated with a (quasi-linear) adaptive system which also strives to enhance the SOI in its own output, y_k . In that sense it is similar to Bussgang techniques for blind adaptation [45, 46], which provide nonlinear processing of the SOI component a_{1k} of their input to yield $\xi[a_{1k}]$ at their output at interval k , such that $\mathcal{E}\{a_{1k}a_{1k}^*\} = \mathcal{E}\{\xi[a_{1k}]\xi[a_{1k}]^*\}$ where a_{1k} is the estimate of the SOI and $r_k = \xi[a_{1k}]$ is the part of the reference signal which correlates with it.

In the sequel, all internal reference-generators will be modeled by a linear processor which provides gain r to the component of y_k correlated with the SOI and a smaller gain ϵ to the remainder of y_k . The conditions on these gains which make the SOI component grow and the remainder to diminish with k will be derived for the NLMS and FRQN algorithms. At some predetermined amplitude of r_k the incremental gain r will approach zero to limit the growth, and ϵ will tend to 0. At such steady state, the adaptive array will behave as assumed in other parts of the thesis; the issue of tracking by the reference generator will not be explored herein. Because all components of y_k are affected by the adapting weights, W_{k-1} , r_k is also a function of these weights and therefore affects the characteristic convergence behaviour of each algorithm. The following treatment examines its effects on the mean convergence properties of each algorithm, particularly the effects of r_k on their stability and time-constants.

Reference-Generating Subsystem with NLMS Algorithm:

Let r_k be the output-derived reference signal given by linear functions of the output signal components during the initial transient stage of adaptation. In this stage the conditions for acquisition of the SOI and at least partial suppression of interfering signals in r_k will be derived. Accordingly, the reference signal at this time is modelled by

$$r_k = r W_{k-1}^H a_{1k} s_1 + \epsilon W_{k-1}^H X_k + \delta W_{k-1}^H \eta_k + r_{0k} , \quad \dots(G1)$$

which is a signal-selective linear operation on the output signal $y_k = W_{k-1}^H Y_k$ with gain r for the SOI component, a much smaller gain ε for the interference and δ for the noise component. These components are identified respectively in the incident-signal vector sample as

$$Y_k = a_{1k} S_1 + X_k + \eta_k \quad \dots(G2)$$

with η_k being the noise component and the interference component being

$$X_k = \sum_{m=2}^M a_{mk} S_m \quad \dots(G3)$$

The "extra" component r_{0k} is not correlated with any signal and will contribute only to the *mmse* and *emse* in the mean-square analysis, which will not be pursued in detail in this treatment. It is assumed that the gains r , ε and δ are real and > 0 , and r_k is substituted directly into a convenient form of the NLMS algorithm such as (55), which after regrouping yields

$$W_k = [I - \beta_k Y_k Y_k^H + \beta_k r Y_k S_1^H a_{1k}^* + \beta_k \varepsilon Y_k X_k^H + \beta_k \delta Y_k \eta_k^H] W_{k-1} + \beta_k Y_k r_{0k}^* \quad \dots(G4)$$

With the usual independence assumptions of Y , W and β applied in the averaging process, the above difference equation becomes

$$W_k = [I - \beta(p_1(1-r)S_1 S_1^H + (1-\varepsilon)(R_I - \sigma^2 I) + (1-\delta)\sigma^2 I)] W_{k-1} \quad \dots(G5)$$

where the relations (147) - (151) of the thesis body are implicit, since $R_I = \mathcal{E}\{Y_k Y_k^H\}$ and $\mathcal{E}\{Y_k \eta_k^H\} = \sigma^2 I$, because the signals are uncorrelated (i.e. $\mathcal{E}\{a_{1k} a_{mk}^*\} = p_1 \delta_{1m}$), $R_S = \mathcal{E}\{Y_k S_1^H a_{1k}^*\} = p_1 S_1 S_1^H$ and $\mathcal{E}\{Y_k r_{0k}^*\} = 0$.

Now in order to determine the range of parameters of the reference-generating subsystem in Figure 26 of the thesis body which will cause it to acquire the SOI in the presence of noise and interference in the adaptive-array system, consider the corresponding subspaces of the array-space, $\mathbb{C}^N$, in terms of the structure of the array covariance matrix, $\mathbf{R}$, in Figure G1. By projecting both sides of (G5) onto any one of them and forming an inner product with the result and a steering vector that is entirely within that subspace, the transient (recursive) behaviour of the corresponding component of $\mathbf{r}_k$ may be analyzed, to see which values of parameters will make it grow or decay, as required.

It is analytically more convenient to define projections which project a vector orthogonally to the defined subspaces (i.e. so the inner product of a vector within the subspace and the resultant, projected vector is 0). Accordingly, let $\mathbf{Q}_I = [\mathbf{S}_2 | \dots | \mathbf{S}_M]$ be the matrix whose columns are the steering vectors of *only* the interference signals and write the projection that projects orthogonally to the interference subspace (I) as

$$\begin{aligned} \mathbf{P}_I^\perp &= \mathbf{I} - \mathbf{Q}_I (\mathbf{Q}_I^H \mathbf{Q}_I)^{-1} \mathbf{Q}_I^H \\ &= \mathbf{I} - \mathbf{P}_I^I \end{aligned} \quad \dots(\text{G6})$$

where the generalized identity operator on I is recognized from Appendix E as e.g. (E4). Note that $\mathbf{P}_I^\perp \mathbf{S}_m = \mathbf{0}$, $m = 2, \dots, M$ in particular. Similarly, the projection which projects any vector so the result is orthogonal to the steering vector of the SOI is simply given by

$$\begin{aligned} \mathbf{P}_S^\perp &= \mathbf{I} - \mathbf{S}_1 (\mathbf{S}_1^H \mathbf{S}_1)^{-1} \mathbf{S}_1^H \\ &= \mathbf{I} - \mathbf{P}_S^I \end{aligned} \quad \dots(\text{G7})$$

which makes $\mathbf{P}_S^\perp \mathbf{S}_1 = \mathbf{0}$. Finally, the projection which projects a vector so the result is orthogonal to *all* the signals' steering vectors is simply

$$\begin{aligned} P_{s,l}^\perp &= I - Q(Q^H Q)^{-1} Q^H \\ &= I - P_{s,l}^I \end{aligned} \quad \dots(G8)$$

with Q defined as in (148). All projection matrices are Hermitian ($P^H = P$), idempotent ($P^N = P$ for any positive integer N), but not invertible.

Starting with the simplest component of the reference signal, that component of the weight vector which is orthogonal to all signal vectors evolves according to the relation $P_{s,l}^\perp W_k = P_{s,l}^\perp [I - \beta(1 - \delta)\sigma^2 I] W_{k-1}$, or in simpler notation wherein $W_{s,l,k}^\perp = P_{s,l}^\perp W_k$, one has the recursion

$$W_{s,l,k}^\perp = (1 - \beta(1 - \delta)\sigma^2) W_{s,l,k-1}^\perp \quad \dots(G9)$$

Clearly, since no steering vector will form a non-zero inner product with the above, it governs the evolution of noise in the reference-acquisition transient and should decay to 0, as noise is not desired in r_k . Assuming the initial condition is generally not 0, it will decay provided that $|1 - \beta(1 - \delta)\sigma^2| < 1$, which ultimately implies that

$$\boxed{\delta < 1} \quad \dots(G10)$$

Next, premultiplying (G5) by (G6) yields the component affecting the evolution of the SOI in the reference signal, as no other steering vector but S_1 will form a non-zero inner product with the result, which is

$$P_I^\perp W_k = W_{I,k}^\perp = W_{I,k-1}^\perp - \beta(1 - r)P_I^\perp P_1 S_1 S^H (P_I^\perp + P_I^I) W_{k-1} \quad \dots(G11)$$

where the other arrangement of (G6) was borrowed to replace an implicit I , so that the recursion could be rearranged as

$$\begin{aligned} \mathbf{W}_{I,k}^\perp = & (1 - \beta(1 - \delta)\sigma^2) \mathbf{W}_{I,k-1}^\perp - \beta(1 - r) \mathbf{P}_1 \mathbf{P}_I^\perp \mathbf{S}_1 \mathbf{S}_1^H \mathbf{W}_{I,k-1}^\perp - \dots \\ & - \beta(1 - r) \mathbf{P}_1 \mathbf{P}_I^\perp \mathbf{S}_1 (\mathbf{S}_1^H \mathbf{P}_I^\perp \mathbf{W}_{k-1}) \end{aligned} \quad \dots(\text{G12})$$

Denoting the inner product of the steering vector of the SOI (i.e. $\mathbf{S}_1$) with $\mathbf{W}_{I,k-1}^\perp$ as $\mathbf{S}_1^H \mathbf{W}_{I,k-1}^\perp = t_{1,k-1}^\perp$ and with its complement as $\mathbf{S}_1^H \mathbf{P}_I^\perp \mathbf{W}_{k-1} = t_{1,k-1}^I$ allows one to write a scalar valued recursion for the part which determines the SOI component of the transient reference signal as

$$t_{1,k}^\perp = (1 - \beta(1 - \delta)\sigma^2 - \beta(1 - r) \mathbf{P}_1 (\mathbf{S}_1^H \mathbf{P}_I^\perp \mathbf{S}_1)) t_{1,k-1}^\perp - \beta(1 - r) \mathbf{P}_1 (\mathbf{S}_1^H \mathbf{P}_I^\perp \mathbf{S}_1) t_{1,k-1}^I \quad \dots(\text{G13})$$

With the last term above serving as a forcing function in the difference equation, it is straightforward to see from a Z-domain analysis that the scale-factor $t_{1,k-1}^\perp$, which determines the SOI contribution to r_k will grow with k providing that

$$1 - \beta(1 - \delta)\sigma^2 - \beta(1 - r) \mathbf{P}_1 \mathbf{g}_{1,opt} > 1 \quad \dots(\text{G14})$$

where it was noted that the optimum gain of the array toward the SOI is $\mathbf{g}_{1,opt} = \mathbf{S}_1^H \mathbf{P}_I^\perp \mathbf{S}_1$. This ultimately reduces to the condition on r that

$$r > 1 + \frac{1 - \delta}{\text{SNR}_{opt}} \quad \dots(\text{G15})$$

Note that if the S and I subspaces were disjoint (i.e. $\mathbf{S}_1 \perp \mathbf{S}_m$, $m = 2, \dots, M$) then the forcing function above would be zero and one would assume a non-zero initial condition for the recursion, and come to the same conclusion for r. Also note that the

forcing function comes from a part of the intersection of S and I subspaces which is not subject to growth or decay¹.

Finally, the condition for the factor which determines the interference contribution to the reference signal will be derived. Premultiplying both sides of (G5) by $P_S^\perp$ gets rid of the SOI contributions, leaving

$$W_{S,k}^\perp = W_{S,k-1}^\perp - \beta(1-\epsilon)P_S^\perp(R_I - \sigma^2 I)W_{k-1} - \beta(1-\delta)\sigma^2 W_{S,k-1}^\perp \dots (G16)$$

which can again be rearranged into standard form by replacing the implicit identity factor of IW_{k-1} with $I = P_S^\perp + P_S^I$, thereby creating a forcing-function in the intersection of the I and S subspaces on the RHS as

$$W_{S,k}^\perp = [I - \beta(1-\delta)\sigma^2 I]W_{S,k-1}^\perp - \beta(1-\epsilon)P_S^\perp[R_I - \sigma^2 I]P_S^\perp W_{S,k-1}^\perp - \beta(1-\epsilon)P_S^\perp(R_I - \sigma^2 I)W_{S,k-1}^\perp \dots (G17)$$

where the idempotence of $P_S^\perp$ was used in the second term. Because that projection matrix is also Hermitian, the matrix $P_S^\perp[R_I - \sigma^2 I]P_S^\perp$ is also Hermitian and therefore has positive eigenvalues. Recalling the definition of R_I , one recognizes this as a system of linear equations for the interference-scaling factors $\{t_m^\perp\}$ ($m=2, \dots, M < N$) when both sides are premultiplied by an interference-steering vector and the inner products $t_{m,k-1}^\perp = S_m^H W_{S,k-1}^\perp$ are identified. It is required that all these scaling factors do not grow during the acquisition transient. That is to say, that the system of equations whose m-th member is

¹ Recall that an adaptive array basically minimizes all signals at its output in the absence of a reference signal or constraint. Therefore $P_I^\perp W_{k-1}$ in (G12) which is proportional to the array gain of those signals is being minimized and not growing. Thus (G13) succeeds in growing the desired part of the reference signal. As that happens, $W_{S,k-1}^\perp$ in (G17) will tend toward the optimal weight for which $R_I W_{opt} = (\text{scalar}) W_{opt}$ holds. Then $P_S^\perp$ will suppress it in the last term (forcing function) in (G17).

Figure G1: Interference (I, rank M) and SOI (S, rank 1) subspaces of complex N-dimensional arrayspace, containing steering vectors (and components of weight vector).

$$\begin{aligned} t_{m,k}^{\perp} = & (1 - \beta(1 - \delta)\sigma^2)t_{m,k-1}^{\perp} - \beta(1 - \epsilon)S_m^H P_S^{\perp} \left(\sum_{m=2}^M p_m S_m t_{m,k-1}^{\perp} \right) - \\ & - \beta(1 - \epsilon)S_m^H P_S^{\perp} \left(\sum_{m=2}^M p_m S_m t_{m,k-1}^{\perp} \right) \end{aligned} \quad \dots(G18)$$

should be stable. The conditions for this stability may be determined by eigen-analysis of the modes of the system. Let $\lambda_m^{\perp}$ be the m-th eigenvalue of $P_S^{\perp} [R_I - \sigma^2 I] P_S^{\perp}$. It is relatively straightforward to see that this system will be stable providing that

$$1 - \beta(1 - \delta)\sigma^2 - \beta(1 - \epsilon)\lambda_m^{\perp} < 1. \quad \dots(G19)$$

Since all quantities in (G19) are > 0 , it follows that the required condition is

$$\epsilon < 1.$$

$\dots(G20)$

In the steady state after the acquisition transients have expired, one would like the NLMS algorithm to operate as assumed elsewhere in this thesis, i.e. with a constant level of reference signal which contains primarily the signal that correlates with the SOI. The growth of that component should therefore cease. This cessation may be achieved through the use of a limiter in the reference-generating subsystem, which will effectively reduce the gain "r" to unity at some predetermined amplitude of $\tau_{1,k-1}^+ a_{1,k}$. Actually, this gain should be maintained slightly larger than unity according to (G15), due to the presence of noise. This condition is reminiscent of the gain-control within an oscillator, where effective unity gain is maintained by a nonlinearity (amplitude-dependency) of the loop gain at the selected frequency in order to maintain oscillation. Since it is the *total* amplitude which is limited, the non-growing components decay.

Reference-Generating Subsystem with FRQN Algorithm:

It will be shown that the same reference-generating subsystem can be equally well applied to the FRQN algorithm. Accordingly, the same reference signal (G1) is substituted into a convenient form of the FRQN algorithm such as (77) to yield

$$[a_0 I + b_0 Y_k Y_k^H] W_k = [a_1 I + b_1 Y_k Y_k^H] W_{k-1} - \tau_1 \tau_2 W_{k-2} + \dots (G21) \\ + g Y_k (r a_{1,k}^* S_1^H + \epsilon X_k^H + \delta \eta_k^H) W_{k-1} + g Y_k r_{0,k}^*$$

where the parameters of the FRQN algorithm are defined as in equations (90) and (92), and the data vector Y_k as in (G2) and (G3). By regrouping the W_{k-1} terms and applying the same independence assumptions whereby

$$\mathcal{E}\{Y_k a_{1,k}^* S_1^H W_{k-1}\} = p_1 S_1 S_1^H W_{k-1} \triangleq R_S W_{k-1}, \quad \dots(\text{G22})$$

$$\mathcal{E}\{Y_k X_k^H W_{k-1}\} = (R_I - \sigma^2 I) W_{k-1} \triangleq R_X W_{k-1}, \quad \dots(\text{G23})$$

$$\mathcal{E}\{Y_k \eta_k^H W_{k-1}\} = \sigma^2 W_{k-1}, \quad \dots(\text{G24})$$

during averaging, the following second-order difference equation is obtained:

$$[a_0 I + b_0 R] W_k = [a_1 I + (b_1 + g r) R_S + (b_1 + g \epsilon) R_X + (b_1 + g \delta) \sigma^2 I] W_{k-1} - \tau_1 \tau_2 W_{k-2} + Q, \quad \dots(\text{G25})$$

where $R = R_S + R_X + \sigma^2 I$ and $\mathcal{E}\{Y_k r_{0,k}^*\} = Q$ were used. The same treatment with projections (G6) - (G8) may now be applied. Firstly, premultiplying both sides of (G25) by (G8) yields the evolution of the portion of the weight vector which has a non-zero inner-product only with noise vectors as

$$a_0 W_{S,I}^+ = (a_1 + (b_1 + g \delta) \sigma^2) W_{S,I}^+ - \tau_1 \tau_2 W_{S,I}^+ \quad \dots(\text{G26})$$

Clearly again, one wants this to decay to 0 during the linear acquisition transient, so that the reference signal will contain negligible noise. The Z-domain transfer function for each component of this projection of the weight vector must therefore be stable. That may be seen by writing the Z-transform of (G26) time-advanced by 2 sample intervals with the initial conditions $W_{S,I}^+ 0$ and $W_{S,I}^+ 1$ as

$$[z^2(a_0 + b_0 \sigma^2) - z(a_1 + (b_1 + g \delta) \sigma^2) + \tau_1 \tau_2] W_{S,I}^+(z) = -z^2 W_{S,I}^+ 0 - z W_{S,I}^+ 1 \quad \dots(\text{G27})$$

where the poles are given by

$$\tilde{z}_{p,1,2} = \frac{(a_1 + (b_1 + g\delta)\sigma^2) \left[1 \pm \sqrt{1 - \frac{4(a_0 + b_0\sigma^2)\tau_1\tau_2}{(a_1 + (b_1 + g\delta)\sigma^2)^2}} \right]}{2(a_0 + b_0\sigma^2)} \quad \dots(\text{G28})$$

which will be solved for the case $g \rightarrow \infty$ to make the simplifying approximations more obvious. It also allows the square root to be approximated according to $\sqrt{1-\epsilon} \approx 1 - \epsilon/2$ for $\epsilon \ll 1$ which gives the two poles as

$$\tilde{z}_{p,1} \approx \frac{\tau_1\tau_2}{a_1 + (b_1 + g\delta)\sigma^2} \rightarrow 0 \quad \dots(\text{G29})$$

and

$$\tilde{z}_{p,2} \approx \frac{a_1 + (b_1 + g\delta)\sigma^2}{a_0 + b_0\sigma^2} = \tilde{z}_{p,1} \quad \dots(\text{G30})$$

where the latter is clearly the dominant one for each component of the projected weight vector. Upon substitution of the parameters it simplifies to

$$\tilde{z}_{p,2} \rightarrow \frac{\tau_2/q + \delta}{1 + \tau_2/q} \quad \dots(\text{G31})$$

Clearly, in order for $|\tilde{z}_{p,2}| < 1$ which ensures that this projected weight vector does not diverge, one must have the condition that

$$\boxed{\delta < 1} \quad \dots(\text{G32})$$

which is the same as for the NLMS algorithm. Note also that this transient settling time may be designed to be fixed, because the dominant pole does not depend on the noise power.

Next, analyze the evolution of the SOI-contributing projection of the weight vector by premultiplying (G25) with (G6). That operation yields

$$\begin{aligned} P_I^+ [a_0 I + b_0 R] W_k &= (a_0 + b_0 \sigma^2) W_{I,k}^+ + b_0 P_I^+ S_1 P_1 S_1^H W_k = \\ &= a_1 W_{I,k-1}^+ + (b_1 + g r) P_I^+ S_1 P_1 S_1^H W_{k-1} + \dots (G33) \\ &+ (b_1 + g \delta) \sigma^2 W_{I,k-1}^+ - \tau_1 \tau_2 W_{I,k-2}^+ . \end{aligned}$$

Again, $W_k = [P_I^+ + P_I^1] W_k$ is substituted for W_k and similarly for W_{k-1} , then an inner product is formed on both sides of the resultant equation with S_1^H . A scalar recursive equation is thereby obtained for $S_1^H W_{I,k}^+ = t_{1,k}^+$ as

$$\begin{aligned} t_{1,k}^+ (a_0 + b_0 \sigma^2) &= -b_0 p_1 (S_1^H P_I^+ S_1) t_{1,k}^+ + a_1 t_{1,k-1}^+ + (b_1 + g r) p_1 (S_1^H P_I^+ S_1) t_{1,k-1}^+ + \\ &+ (b_1 + g \delta) \sigma^2 t_{1,k-1}^+ - \tau_1 \tau_2 t_{1,k-2}^+ + \dots (G34) \\ &+ (b_1 + g r) p_1 (S_1^H P_I^+ S_1) t_{1,k-1}^+ - b_0 p_1 (S_1^H P_I^+ S_1) t_{1,k}^+ . \end{aligned}$$

In the Z-domain, the corresponding transfer function for the SOI component coefficient, defined as before by $h_1^+(z) \triangleq t_1^+(z) / t_1^1(z)$, is found to be

$$h_1^+(z) = \frac{g_{1,opt} p_1 (b_1 + g r - z b_0)}{z^2 (a_0 + b_0 \lambda_1^+) - z (a_1 + b_1 \lambda_1^+ + g (r p_1 g_{1,opt} + \delta \sigma^2)) + \tau_1 \tau_2} , \dots (G35)$$

where the notation was simplified with the use of $g_{1,opt} = S_1^H P_I^+ S_1$, $\lambda_1^+ = \sigma^2 + p_1 g_{1,opt}$. Again, the poles are computed using the quadratic formula as

$$\begin{aligned} \tilde{z}_{p1,2} = \frac{(a_1 + b_1 \lambda_1^+ + g (r p_1 g_{1,opt} + \delta \sigma^2)) \left[1 \pm \sqrt{1 - \frac{4(a_0 + b_0 \lambda_1^+) \tau_1 \tau_2}{(a_1 + b_1 \lambda_1^+ + g (r p_1 g_{1,opt} + \delta \sigma^2))^2}} \right]}{2(a_0 + b_0 \lambda_1^+)} \end{aligned} \dots (G36)$$

With the same approximations, the two poles become approximately

$$z_{p1} = \frac{\tau_1 \tau_2}{a_1 + b_1 \lambda_1^+ + g(rp_1 g_{1,opt} + \delta \sigma^2)} - 0, \quad \dots(G37)$$

and

$$z_{p2} = \frac{a_1 + b_1 \lambda_1^+ + g(rp_1 g_{1,opt} + \delta \sigma^2)}{a_0 + b_0 \lambda_1^+} - z_{p1} - \frac{\tau_2/q + r}{1 + \tau_2/q}. \quad \dots(G38)$$

Again, for the SOI contribution $t_{1,k-1}^+ a_{1,k}$ to grow during the linear acquisition transient, it is required to have the dominant pole $|z_{p2}| > 1$, which by inspection requires that $r > 1$. More precisely, owing to the presence of thermal noise σ^2 , it is required to have slightly higher gain as $r > (SNR_{opt} - \delta)/(SNR_{opt} - 1)$, or approximately

$$r > 1 + \frac{1 - \delta}{SNR_{opt}} \quad \dots(G39)$$

when the $SNR_{opt} > 1$ and $r \ll 1$. This is the same condition as for the reference-loop used with the NLMS algorithm.

Finally, the condition which determines the interference-contribution coefficients' evolution during the linear acquisition transient will be derived. This is again a matrix version of the above treatment. Premultiplying both sides of (G25) by P_S^+ leaves

$$\begin{aligned} P_S^+ [a_0 I + b_0 R] W_k &= (a_0 + b_0 \sigma^2) W_{S,k}^+ + b_0 P_S^+ R_X W_k = \\ &= a_1 W_{S,k-1}^+ + (b_1 + g\epsilon) P_S^+ R_X W_{k-1} + \dots \\ &+ (b_1 + g\delta) \sigma^2 W_{S,k-1}^+ - \tau_1 \tau_2 W_{S,k-2}^+ \end{aligned} \quad \dots(G40)$$

Once again, the W_k and W_{k-1} terms may be separated into W_S^+ and W_S^I components. Then, premultiplying both sides of the result by the matrix of interference steering vectors Q_I (from (G6)) and by $\sqrt{P_I}$ yields the vector equation

for $\sqrt{P_I} T_{S,k}^\perp = \sqrt{P_I} Q_I^H P_S^\perp W_k = \sqrt{P_I} Q_I^H W_{S,k}^\perp$ (where $T_{S,k}^\perp \triangleq [t_{2,k}^\perp, \dots, t_{m,k}^\perp, \dots, t_{M,k}^\perp]^T$ is the vector of interference-contribution coefficients) as

$$\begin{aligned} (a_0 + b_0 \sigma^2) \sqrt{P_I} T_{S,k}^\perp = & -b_0 \left(\sqrt{P_I} Q_I^H P_S^\perp Q_I \sqrt{P_I} \right) \sqrt{P_I} T_{S,k}^\perp + (a_1 + (b_1 + g\delta) \sigma^2) \sqrt{P_I} T_{S,k-1}^\perp + \\ & + (b_1 + g\epsilon) \left(\sqrt{P_I} Q_I^H P_S^\perp Q_I \sqrt{P_I} \right) \sqrt{P_I} T_{S,k-1}^\perp - \tau_1 \tau_2 \sqrt{P_I} T_{S,k-2}^\perp + \\ & + (b_1 + g\epsilon) \left(\sqrt{P_I} Q_I^H P_S^\perp Q_I \sqrt{P_I} \right) \sqrt{P_I} T_{S,k-1}^\perp - b_0 \left(\sqrt{P_I} Q_I^H P_S^\perp Q_I \sqrt{P_I} \right) \sqrt{P_I} T_{S,k}^\perp, \end{aligned} \quad \dots(G41)$$

in which P_I is not to be confused with that in equation (G6), but rather taken from $R_X = Q_I \sqrt{P_I} \sqrt{P_I} Q_I^H$. The forcing-function vectors are similarly defined according to $T_{s,k}^\perp \triangleq Q_I^H P_S^\perp W_k = Q_I^H W_{s,k}^\perp$ and similarly for $T_{s,k-1}^\perp$. It is noticed that the $M \times M$ matrix

$$\left(\sqrt{P_I} Q_I^H P_S^\perp Q_I \sqrt{P_I} \right) = \sqrt{P_I} Q_I^H P_S^\perp P_S^{\perp H} Q_I \sqrt{P_I} \quad \dots(G42)$$

is positive-definite and Hermitian so it has M positive eigenvalues $\{\lambda_m^\perp\}$ and M corresponding orthonormal eigenvectors $\{U_m^\perp\}$. At the risk of protracting the tedium of creative notation to include

$$E_S^\perp = [U_2^\perp, \dots, U_m^\perp, \dots, U_M^\perp], \quad \dots(G43)$$

$$\Lambda_S^\perp = E_S^{\perp H} \left(\sqrt{P_I} Q_I^H P_S^\perp P_S^{\perp H} Q_I \sqrt{P_I} \right) E_S^\perp \quad \dots(G44)$$

and

$$\Gamma_{S,k}^\perp = E_S^{\perp H} \left(\sqrt{P_I} T_{S,k}^\perp \right) \quad \dots(G45)$$

the diagonalized equation can then be written as

$$\begin{aligned}
(a_0 + b_0 \sigma^2) \Gamma_{S,k}^+ &= -b_0 \Lambda_S^+ \Gamma_{S,k}^+ + (a_1 + (b_1 + g \delta) \sigma^2) \Gamma_{S,k-1}^+ + \\
&+ (b_1 + g \varepsilon) \Lambda_S^+ \Gamma_{S,k-1}^+ - \tau_1 \tau_2 \Gamma_{S,k-2}^+ + \dots \quad \text{...(G46)} \\
&+ (b_1 + g \varepsilon) \Lambda_S^+ \Gamma_{S,k-1}^+ - b_0 \Lambda_S^+ \Gamma_{S,k}^+ .
\end{aligned}$$

The m-th component of the above is an uncoupled scalar difference equation which must be stable in order for the interference-contribution coefficients not to grow during the linear reference-acquisition transient. Therefore the Z-domain poles of

$$\begin{aligned}
\gamma_{m,k}^+ (a_0 + b_0 (\sigma^2 + \lambda_m^+)) &= \gamma_{m,k-1}^+ (a_1 + \lambda_m^+ (b_1 + g \varepsilon) + \sigma^2 (b_1 + g \delta)) - \gamma_{m,k-2}^+ \tau_1 \tau_2 + \\
&+ (b_1 + g \varepsilon) \lambda_m^+ \gamma_{m,k-1}^+ - b_0 \lambda_m^+ \gamma_{m,k}^+ \\
&\dots \quad \text{...(G47)}
\end{aligned}$$

must lie within the unit circle $|z| < 1$. The poles of that transfer function (defined as $\eta_m^+(z) \triangleq \gamma_m^+(z) / \gamma_m^+(z)$) are computed as

$$z_{p1,2} = \frac{(a_1 + \lambda_m^+ (b_1 + g \varepsilon) + \sigma^2 (b_1 + g \delta)) \left[1 \pm \sqrt{1 - \frac{4(a_0 + b_0 (\sigma^2 + \lambda_m^+)) \tau_1 \tau_2}{(a_1 + \lambda_m^+ (b_1 + g \varepsilon) + \sigma^2 (b_1 + g \delta))^2}} \right]}{2(a_0 + b_0 (\sigma^2 + \lambda_m^+))} \quad \text{...(G48)}$$

With the same approximations as in the treatments of all the other contributions to the reference signal, the two poles for the m-th mode of these interference contributions

become

$$z_{p,1} \approx \frac{\tau_1 \tau_2}{a_1 + \lambda_m^+(b_1 + g\varepsilon) + \sigma^2(b_1 + g\delta)} - 0, \quad \dots(G49)$$

and

$$z_{p,2} = \frac{a_1 + \lambda_m^+(b_1 + g\varepsilon) + \sigma^2(b_1 + g\delta)}{a_0 + b_0(\sigma^2 + \lambda_m^+)} - z_{p,1} - \frac{\tau_2/q + \varepsilon}{1 + \tau_2/q}. \quad \dots(G50)$$

More precisely, $z_{p,2} \approx (\tau_2/q + \varepsilon)/(1 + \tau_2/q) + (\delta - \varepsilon)/[(1 + \tau_2/q)INR]$ but since $\varepsilon < 1$ and $\delta < 1$ are similar quantities, the approximate result in (G50) suffices to determine that, as in the case of δ and the noise contribution, it is required to have

$$\varepsilon < 1.$$

...(G51)

The above treatments confirm that exactly the same internal reference-generating loops as may be used with the NLMS algorithm, will work just as well and stably with the new FRQN algorithm. Moreover, the acquisition transient response times of the latter will be independent of the array-covariance-matrix eigenvalues, unlike in the case of LMS type of algorithms.

The above results in boxes are the same conditions as stated in [42], but were not derived therein. It can also be deduced that the result of [47], namely that equal powers be given to an additive reference signal as to the information signal, corresponds to the condition $r = 1$ in the above treatments, which means that no action takes place on the desired signal. The other cases therein predictably lead to faster nulling of the desired signal as the proportion of reference power is decreased, corresponding to $r < 1$. The reference loop therein should contain a limiter to maintain the condition $r = 1$ after acquisition, as in most other such loops.

Appendix H: Toward Implementation

In order to present the new FRQN algorithm in a more user-friendly form which can be adapted for implementation in any application, it was encoded using the block-diagram-based simulation software SIMULINK[®] (Version 1.3) which is available as an extension of the MATLAB software (version 4.2c.1 with Signal-Processing Toolbox version 3.0b). The algorithm was constructed using readily-available signal-processing blocks in SIMULINK[®] and the additional DSP Blockset (version 1.0a), in a relatively straightforward manner according to Tables 5 and 6. It was then inserted into a modified copy of the Noise Cancellation Model taken from the Demos group in the DSP Blockset. The original copy of this demonstration file contains the same model with NLMS and RLS algorithms for adapting the coefficients of the Adaptive FIR filter block, for comparison. It is described on pages 2-57 to 2-60 of the DSP Blockset Users' Guide. It is a purely real-valued formulation of a relatively well-conditioned, time-domain system-identification type of problem. With the help of the additional software package Real-Time Workshop, such systems created with SIMULINK[®] and DSP Blockset "building-blocks" can be turned into real-time executable C-code for rapid prototyping on floating-point DSP chips on evaluation-boards.

In this section, the "packaging" of the FRQN algorithm and its testing in the Noise-Cancellation Model of the DSP Blockset will be described. Real-time prototyping was not performed; it is felt that further steps toward implementation become too application-specific in their details and can be adapted from the block-diagram formulation presented herein.

Figure H1 shows the block diagram of the Adaptive Noise-Cancellation Demos as it appears after double-clicking on the Noise-Cancellation using Adaptive FIR Filter button in the Demos group of the DSP Blockset. Note that it is a discrete-time-domain filter and uses purely real arithmetic. Briefly, a signal of interest (SOI) is corrupted by low-pass filtered noise and processed by the adaptive system. Also available to the system is the same noise but unfiltered, which is highly correlated with the noise corrupting the SOI. The Adaptive FIR block must adapt itself as best it can to behave like the (unknown) Noise Filter block in order to minimize the feedback error power by cancelling the noise from the corrupted SOI through coherent subtraction.

Figure H1: SIMULINK[®] block diagram of Adaptive Noise Cancellation Model using a real-valued Normalized LMS algorithm.

In terms of the notation of the preceding sections, the top input of the Adaptive FIR block generates the data vector Y_k internally; the top output is $y_k = W_{k-1}^T Y_k$ which is subtracted from the current sample of the corrupted SOI that serves as r_k . The so-called Output Signal in this demo is actually $-\alpha_k = r_k - y_k$, which is fed back to the Adaptive FIR filter at its bottom input. Wide lines designate vector-valued quantities, e.g. the filter-tap coefficient vector W_{k-1} .

By selecting Unmask from the Options menu and double-clicking on the Adaptive FIR block, the block diagram of the DSP blockset's real-valued normalized LMS algorithm is revealed as in Figure H2. (Output #2 is mislabelled; it should be $W[k-1]$.) The buffer is a serial-to-parallel converter which generates the data vector of length N from the input time-series $\{x[k]\}$. For an FIR time-domain filter structure such as an adaptive T-spaced equalizer, buffer overlap is $N-1$; for an adaptive sensor array it should be 0. Sample time is set to 1 in the buffer as well as in the $1/z$ (delay) block. The buffer length N should be selected the same as that of the FIR Noise Filter block, i.e. 32 in this application.

Figure H2: Block diagram of SIMULINK® model of the Normalized LMS algorithm used in Figure H1.

In terms of the notation used in the preceding sections, $e[k]$ corresponds to $-\alpha_k$ and $X[k]$ to Y_k^T , with superscript T used in place of H because the algorithm deals only with real numbers. This version of the NLMS algorithm is governed by the difference equation $W_k = W_{k-1} - g_k Y_k \alpha_k$ with $g_k = \mu N / (Y_k^T Y_k + \text{eps})$, $\alpha_k = y_k - r_k$ and $\text{eps} = 10^{-10}$ when $Y_k^T Y_k = 0$. This is somewhat different than the form of the NLMS algorithm developed in the preceding chapters herein (e.g. equations (54) - (57)) because it can be made unstable by choice of μ . Specifically, the SIMULINK[®] NLMS algorithm will converge stably in the mean only if $0 < \mu N \lambda_{\max} / (\sum_{n=1}^N \lambda_n) < 2$, so if all N eigenvalues are roughly equal, one requires $\mu < 2$ and if N-1 of them are negligible compared to one dominant eigenvalue, then $\mu < 2/N$ is required for stability. Also note that when $1 < g_k \lambda_n < 2$, it will converge in an oscillating series, although it may be up to twice as fast as the NLMS algorithm developed in the preceding chapters herein. By contrast, the latter NLMS algorithm is unconditionally stable.

Again, the wide lines denote vector quantities. The block diagram of this NLMS algorithm may be edited and when done, selecting Close from its File menu collapses it back into a subsystem block. Then, selecting Mask from the Noise-Cancellation block-diagram's Options menu converts it back into the Adaptive FIR block in Figure H1; its mask which defines parameters such as the normalization gain "mu*n" can now be edited to reflect any changes made in the algorithm block diagram. By double-clicking the Adaptive FIR block, the parameter values may be edited in the mask dialogue box which appears; default values are N = 32, mu = 0.02 and Sample Time = 1.

Figure H3 is the block diagram of the same Adaptive Noise Cancellation Demo model as Figure H1, but uses a real-valued RLS algorithm to adapt the same FIR adaptive filter. The corresponding RLS FIR block cannot be exploded into a block

diagram of the RLS algorithm; it exists only in the form of lines of code evaluated as a function at each iteration. When double-clicked, its mask allows editing of the exponential "forgetting factor" λ , whose default value is 1 (as for stationary scenarios) and Sample Time, which is also set to 1 by default. (It is probably not the same as the RLS algorithm used elsewhere in this thesis, as it was invariably found to suffer from numerical instability when $\lambda < 1$.)

Figure H3: SIMULINK[®] block diagram of Adaptive Noise Cancellation Model using a real-valued RLS algorithm.

Figure H4 shows the same Adaptive Noise Cancellation Demo model incorporating the newly-developed FRQN algorithm to adapt the FIR filter contained in the "frqn" block. Because the frqn block already contains the subtracting junction and feedback connection for $-\alpha_k$, Figure H4 looks different from Figures H1 and H3. Also, a clock (the round one) was added to count the iterations. The top input to the frqn block is r_k and the bottom one is for the time-series that generates Y_k internally. Its top output is y_k , the middle one is $-\alpha_k$ (noise-canceller output, i.e. the SOI, with inverted phase, after successful adaptation) and the bottom (vector) output is the filter coefficient vector, W_{k-1} .

Figure H4: SIMULINK[®] block diagram of Adaptive Noise Cancellation Model using a real-valued FRQN algorithm, including the noise-cancellation junction.

When unmasked and double-clicked, the frqn block appears as in Figure H5. Each "building-block" is labelled with the term it generates at its output according to Tables 5 and 6 of the theisi body; prefix "R" denotes the use of real-only arithmetic. The Data Input buffer's mask is edited to make Buffer Length = N, Buffer Overlap = lap and Sample Time = dt. The delay-blocks' masks are edited to make Sample Time = dt and initial conditions = 0, as per Table 5. Parameters N, lap, dt and other FRQN algorithm parameters are given values in the mask for the whole frqn subsystem block which encapsulates Figure H5.

Figure H5: Block diagram of SIMULINK[®] model of the real-valued FRQN algorithm used in Figure H4.

The frqn mask was filled in as follows:

```

Length of Input Signal Vector, N:
32
Buffer Overlap (N-1 for FIR filter, 0 for sensor
array), lap:
31
Sample Time, dt:
1
Maximum Loop-Gain Parameter, g:
10^6
Maximum Input SNR, dB:
60
Speed-up Factor, q:
1

```

Because the minimum eigenvalue is typically the noise power in input signal, the loop-gain parameter may be set according to (116) as $g > qSNR/(\tau_2 \lambda_{\text{signal-power}})$. Since $\tau_2/q > 1$, one option in the mask is to use a gain-parameter $G=10^{(SNR \text{ dB}/10)}$ when the input signal power is roughly equal to 1. The only other parameter required by the FRQN algorithm is the length, N , of the input signal vector Y_k , which should be the same as the length of the FIR Noise Filter in the demo model. In the Simulation Parameters menu, one should select Minimum Step-Size = Maximum Step-Size = 1 with the Runge-Kutta 5 method, and $\text{tol} = 1.0\text{e-}006$.

The mask for setting and initializing the parameters of the frqn subsystem block itself was set up as in the following text box.

Block Name:
FRQN

Initialization Parameters:
FRQN\Length of Input Signal Vector, N:\Buffer Overlap, (N-1 for FIR time-domain filter, 0 for sensor array),
lap:\Sample Time, dt:\Maximum Loop-Gain Parameter,
g:\Maximum Input SNR, dB:\Speed-up Factor, q:

Initialization Commands:
N=@1;lap=@2;dt=@3;g=@4;q=@6;
tau1=N;tau2=4*tau1;gama=1/tau2;G=q*10^(@5/10)/tau2;
p0=1/(1+gama/tau1);
p1=1/(1+tau2/q);
p2=p1*tau2*(tau1+gama)/G; % (or g)

Drawing Commands:
frqn

Help Instructions:
Fast Robust Quasi-Newton Adaptive Algorithm for General Array Processing.
(c) M. Klemes, 1996. Top input is reference, $r[k]$; bottom input is series for $Y[k]$ (an N-vector). Top output is $y[k]=W[k-1]^T Y[k]$; middle output is $-\alpha[k]=r[k]-y[k]$ and bottom output is $W[k-1]$.

Thus the FRQN algorithm is encapsulated in an accessible, user-friendly form which can be readily adapted for a specific real-time application. It has been verified to converge and run stably in the Adaptive Noise-Cancellation Demo system for more than 3.46×10^5 iterations (about 18 hours on a 66 MHz '486 PC). It converged in about the same number of iterations as the SIMULINK[®] NLMS algorithm with $\mu = 0.02$ (which is roughly equivalent to the NLMS of equations (54) - (57) with $g \rightarrow \infty$), which agrees with theoretical expectations since the system corresponds to a relatively well-conditioned scenario when noise is used to generate the input signal Y_k . The FRQN algorithm required 194 multiplications per iteration in this system.

Also as expected, the misadjustment error of the system with FRQN was less than that of the system using the SIMULINK[®] NLMS algorithm with $\mu = 0.02$. When μ is increased, the misadjustment increases. (It corresponds to the thesis version of NLMS

having a step-size larger than that required to make the *a-posteriori* error $e_k = W_k^H Y_k - r_k$ identically 0 and causes the coefficients to overcompensate, resulting in an oscillatory converging geometric series. If μ was made larger than about $2/N$, the SIMULINK[®] NLMS algorithm would diverge.) As evident from Figure H2, this NLMS algorithm required 96 multiplications per iteration.

The real-valued RLS algorithm in this demonstration system converged the fastest, rather suddenly after about $2N$ iterations, as would be expected. However, when its exponential forgetting factor was decreased from $\lambda = 1$ to that used in the RLS of preceding sections, e.g. to $\lambda = 1 - 1/\tau_2$ (or even closer to 1), it invariably diverged after a few minutes of run time, presumably owing to numerical instability. This did not occur in the adaptive-array application of the RLS algorithm used elsewhere in this thesis. That RLS algorithm would require 4226 multiplications per iteration.

Thus it has been demonstrated that the FRQN algorithm compares favourably with other versions of the NLMS and RLS algorithms in applications other than adaptive arrays and is equally simple to use. In this form, the algorithm can be used to gather vast amounts of simulation data with ease, but that would serve no new purpose, especially since a truly Fast RLS algorithm already exists [6] for adapting time-domain FIR structures.

Appendix I: Guide to Abbreviations and Notation:

This appendix contains explanations of abbreviations and meanings of symbols used in the notation in the thesis. Standard mathematical symbols are not listed. Some symbols are used in intermediate developments and may be the same as those used elsewhere for other purposes; their meanings are evident from their respective contexts.

A	An arbitrary matrix in Appendix E. Also a partial-fraction coefficient in Appendix B.
A_k	Matrix short form in equation (...D45).
$A(t)$	M-Vector of M complex envelopes of M received signals.
$a_m(t)$	Complex envelope of m-th received signal at continuous time, t.
a_{1k}	Complex envelope of 1-st received signal at discrete-time $t = k\Delta t$.
AR	Auto Regressive
AMPS	Advanced Mobile Phone System.
a_0, a_1, a_2	Parameters of the FRQN algorithm.
$\alpha_k; \alpha_{opt,k}$	A-priori output error at k-th discrete-time interval; minimum a-priori (with optimal weights) output error at the same time.
B	Occupied signal bandwidth. Also number of bits of precision.
B	An arbitrary matrix in Appendix E. Also a partial-fraction coefficient in Appendix B and transformed steering vector S .
B_k	Matrix short form in equation (...D46).
BER	Bit Error Rate (i.e. probability of bit error).
b_0, b_1, b_2	Parameters of the FRQN algorithm.
b_n	n-th component of transformed steering vector.
β, β_0	Step-size parameter of the LMS algorithm.
$\beta(t)$	Step-size parameter of continuous-time Normalized LMS algorithm.

β_k	Step-size parameter of discrete NLMS algorithm at k-th iteration ¹ .
C	An arbitrary matrix in Appendix E.
c, c_k	A parameter in the excess mean squared error of the FRQN algorithm, at the k-th iteration.
C_k	Covariance matrix of transformed weight-error vectors at the k-th iteration, of the NLMS algorithm. Also a matrix short form in equation (...D117).
C_k	Vector of diagonal elements of covariance matrix C_k .
$\mathbb{C}^N$	N-dimensional complex vector space (array-space of an N-element array).
D	An arbitrary matrix in Appendix E. Also a matrix short form, equation (...D69).
D_k	A matrix short form, equation (...D118). Also a vector short form in Table 6.
$diag[\]$	A vector of the diagonal elements of matrix $[\]$.
Δ	The finite difference between a discrete-time quantity at iteration k and at iteration k-1 (e.g. $\Delta W_k = W_k - W_{k-1}$, $\Delta t = k\Delta t - (k-1)\Delta t$).
Δ^2	Second-order difference, e.g. $\Delta^2 W_k = \Delta(\Delta W_k) = W_k - 2W_{k-1} + W_{k-2}$.
δ	A parameter in the continuous-time predecessor of the FRQN algorithm. Also the gain for uncorrelated array noise through a reference-generator.
$\delta(t - k\Delta t); \delta_{1,m}$	Unit impulse (Dirac delta) function; Kronecker delta.
E	Eigenvector transformation matrix. Also an arbitrary matrix in Appendix E.

¹One iteration takes place in one time interval Δt in discrete time.

$E_S^\perp$	Matrix of eigenvectors orthogonal to the desired-signal array vector (SOI subspace).
$e_k; e_{opt}$	Output error at k-th iteration; minimum output error (with optimal weights).
$emse$	Excess mean squared error.
$\mathcal{E}\{\cdot\}; \mathcal{E}_Y\{\cdot\}$	Expected value of $\{\cdot\}$; expectation with respect to ensemble of Y.
ϵ	An arbitrarily small scalar constant.
ε	Gain of reference signal generator for interference component of array output signal.
e_k	Intermediate scalar in computation of the k-th iteration of FRQN algorithm.
$\eta_{\Delta w_n}$	Quantization error in the n-th component of the weight-step vector.
η_Y	Quantization error vector in the input-signal vector Y_k .
$\eta(t)$	Noise vector component of array input signal vector in continuous time.
$\eta_m^\perp(z)$	Transfer function of m-th transformed gain coefficient for interference components of the reference signal.
η_k	Discrete-time noise vector component at the k-th iteration.
F_k	Covariance matrix of transformed weight-lag-error vectors at k-th iteration of the FRQN algorithm.
$\mathcal{F}_k$	Vector of the diagonal elements of covariance matrix F_k .
FRQN	Fast Robust Quasi-Newton (algorithm).
f	An intermediate scalar short form in the analysis of FRQN algorithm convergence in the mean square. Also frequency in Hz in Appendix A.

$f(\varphi, \Phi)$	An intermediate scalar function of φ and Φ in the tracking analysis of the FRQN algorithm.
$f_{1,2,3}(\varphi, \Phi, algo)$	Three (subscripts 1, 2, 3) scalar function short forms in tracking analysis of the FRQN algorithm, including dependencies on algorithm parameters.
φ_k	The zero-mean Gaussian random vector assumed to be driving the weight-vector non-stationarity Markov process.
Φ	Non-stationarity parameter related to time-variability of "optimal" weights.
ϕ_m	Elevation angle of the m -th signal source.
G_k	Intermediate vector in the computation of the k -th iteration of the FRQN algorithm.
$G_n(\theta_m, \phi_m)$	Voltage gain of the n -th array element in the direction of the m -th source.
g	Loop-gain in the continuous-time LMS algorithm. Also a gain parameter in the FRQN algorithm.
g_d	A gain parameter in the FRQN algorithm, $=g/q$.
$g_{1,opt}$	Optimum gain in the direction of the SOI.
$g(\varphi, \Phi)$	An intermediate scalar function of φ and Φ in the tracking analysis of the FRQN algorithm.
$\Gamma_{S,k}^{\perp}; \Gamma_{S,k}^{\parallel}$	Transformed vectors of gains for signals whose array-vectors are $\perp$ to that of the SOI; $\parallel$ to that of the SOI.
γ	A leakage parameter in the FRQN algorithm.
$\gamma_{m,k}^{\perp}; \gamma_{m,k}^{\parallel}$	The m -th components of $\Gamma_{S,k}^{\perp}; \Gamma_{S,k}^{\parallel}$, respectively.
$[]^H$	The complex conjugate transpose of a vector or matrix in $[]$.
H_{k-1}	Covariance matrix of weight error vectors at the k -th iteration of the FRQN algorithm.

$H_{k1,2}$	Cross-covariance matrix of weight error vectors at the k-1st and k-2nd iterations of the FRQN algorithm.
h	An intermediate short form scalar function in the analysis of FRQN convergence in the mean square.
$h(\varphi, \Phi)$	An intermediate scalar function of φ and Φ in the tracking analysis of the FRQN algorithm.
$h_1^+(z)$	Z-domain transfer function of the SOI component of the reference signal.
I	Identity matrix. Also a subscript pertaining to interference.
IF	Intermediate Frequency.
$Im[\]$	Imaginary part of complex quantity in [].
INR	Interference to Noise Ratio.
J	Vector of the diagonal elements of I (i.e. a vector of 1s).
K_k	An intermediate vector in the computation of the k-th iteration of the RLS algorithm.
K_{k-1}	Also the covariance matrix of the weight error vectors at the k-1st iteration of the NLMS algorithm.
L_k	An intermediate vector in the computation of the k-th iteration of the RLS algorithm.
L_{k-1}	Also the weight vector error due to lag and acceleration at the k-1st iteration of the FRQN algorithm.
LHS	Left-Hand Side.
LMS	Least Mean Squares (algorithm).
LSB	Least Significant Bit.
Λ	Diagonalized array covariance matrix.
λ_n	n-th eigenvalue of array covariance matrix (diagonal entry of Λ).
$\underline{\lambda}$	Vector of eigenvalues of the array covariance matrix.

$\mathcal{L}[\]$	Laplace transform of the function (of t) in $[\]$.
$\lambda_1^\perp$	Eigenvalue in the subspace orthogonal to that spanned by the interference array vectors.
$\Lambda_S^\perp$	Eigenvalue matrix for subspace spanned by array vectors orthogonal to that of the SOI.
$\lambda_m^\perp; \lambda_m$	m-th diagonal entry of $\Lambda_S^\perp$, ($m=2,\dots,M$); wavelength of m-th signal.
M	The number of spatially discrete signal sources in the far field, or linear combinations thereof.
M_-	Steady-state misadjustment ratio.
$[\]_m$	Subscript pertaining to the m-th signal source or m-th linear combination thereof.
μ	Ratio of loop gain and time-constant in the FRQN and LMS algorithms.
N	The number of array elements and adaptive weights, or linear combinations thereof.
NLMS	Normalized Least Mean Squares (algorithm).
$N_k; N_{k1,2}$	Transformed covariance matrix H_k of weight error vectors at the k-th iteration of the FRQN algorithm; transformed covariance matrix $H_{k1,2}$.
$\mathbf{N}_k; \mathbf{N}_{k1,2}$	Vector of the diagonal elements of covariance matrix N ; that of $N_{k1,2}$.
v	Imaginary part of the independent complex variable z in the Z-domain.
$v_{n,k}$	n-th element of the transformed weight vector at the k-th iteration.
v_k	Model noise at k-th iteration in system-identification tracking model.

$[\]_{opt}$	Subscript pertaining to the optimal adapted state of the array weights.
Ω	An intermediate scalar function of FRQN algorithm parameters in its tracking analysis.
ω	Imaginary part of the independent complex variable s in the S (Laplace transform) domain (radian frequency).
ω_m	Radian frequency of the m -th signal.
ω_p	Pole radian frequency corresponding to the pole at s_p in the S -plane.
$P(W)$	Penalty function (scalar) of vector W .
p	A scalar function of parameters of the FRQN algorithm in its mean square convergence analysis, analogous to β for NLMS.
p_k	An intermediate scalar in the computation of the k -th iteration of the FRQN algorithm.
p_0, p_1, p_2	Scalar functions of parameters of the FRQN algorithm.
P	Diagonal matrix of all M signal powers (signal covariance matrix).
p_m	m -th signal's power (diagonal entry of P), $m=2, \dots, M$.
p_1	Signal power of the SOI ($m=1$).
P_I	Diagonal matrix of interference signals' powers (their covariance matrix).
$P_I^{\perp}; P_I^{\parallel}$	Projection matrix which projects a vector so the resultant vector is orthogonal to the interference subspace; complementary projection (resultant is within interference subspace).
$P_S^{\perp}; P_S^{\parallel}$	Projection matrix "orthogonal to" the SOI subspace; complementary projection matrix "parallel to" the SOI subspace.
$P_{S,I}^{\perp}; P_{S,I}^{\parallel}$	Projection matrix "orthogonal to" the whole signal subspace; complementary projection matrix "parallel to" the signal subspace.

$P_B^I; P_D^I$	Projection matrix whose resultant is in subspace spanned by rows of B ; similar projection matrix with respect to columns of D (generalized identity operators in those respective subspaces).
$[]_p$	Subscript denoting a pole.
ψ_k	An intermediate scalar variable in the computation of the k-th iteration of the FRQN algorithm.
Q	Matrix whose columns are the array vectors of the M signal sources.
Q_I	Matrix of the array vectors of only the interference sources.
q	A scalar gain parameter of the FRQN algorithm.
$[]_q$	Dummy index of summation in the formulation of the RLS algorithm.
$R; R_k$	Array covariance matrix; its estimate at the k-th iteration of the
RLS	algorithm.
$R_I; R_S; R_X$	Interference-plus-noise array covariance matrix; SOI array covariance matrix; interference-only array covariance matrix.
RHS	Right-Hand Side.
RLS	Recursive Least Squares (algorithm).
$\text{Re}[]$	Real part of the complex quantity in $[]$.
$\Re_w$	Range of weight coefficient voltages in vector W .
$r; r_k; r_{0,k}$	Gain of reference-generating loop for reference-signal component correlated with the SOI; reference signal at the k-th iteration; component of reference signal uncorrelated with the SOI, interference or array noise.
ρ	A parameter in the generalized s-to-z transformation and the corresponding version of the FRQN algorithm.

$S; S_k$	Array or steering vector of the SOI; its estimate at the k-th iteration of the RLS algorithm.
$S_1; S_m$	Array vector of the SOI; array vector of the m-th interfering signal ($m=2,\dots,M$).
$s_{m,n}$	n-th entry of the array vector of the m-th signal.
SINR	Signal to Interference plus Noise Ratio.
SNR	Signal to Noise Ratio.
SOI	Signal Of Interest.
$[]_s$	Subscript pertaining to the SOI.
s	The independent complex variable in the Laplace transform domain (S-plane).
σ	Real part of the independent variable s .
σ^2	Variance (normalized power) of array noise.
σ_r^2	Variance (normalized power) of the reference signal r_k .
σ_φ^2	Variance of vector noise process driving the weight nonstationarity φ_k .
$T; T_1, T_2$	Open-loop time-constant of the continuous-time LMS algorithm; time-constants of the continuous-time predecessor of the FRQN algorithm.
$T_{S,k}^i; T_{S,k-1}^1$	Vector of interference contribution factors to the transient reference signal at the k-th iteration; vector of complementary forcing terms at the k-1st iteration.
t	Independent variable in continuous-time domain.
$tmse$	Tracking mean squared error (due to weight lag in NLMS and weight lag plus acceleration in FRQN algorithms).

$t_{1,k}^{\perp}; t_{1,k-1}^{\perp}$	Scalar factor determining the SOI contribution to the transient reference signal at k-th iteration; its complement and forcing term from intersection of SOI and interference subspaces at iteration k-1.
$t_{m,k}^{\perp}; t_{m,k-1}^{\perp}$	Scalar factor determining the interference contribution to the transient reference signal at k-th iteration; its complement and forcing term from the intersection of SOI and interference subspaces at iteration k-1. (m-th entries of vectors $T_{S,k}^{\perp}; T_{S,k-1}^{\perp}$, respectively, $m=2,\dots,M$).
$\tau_n; \tau_{1,n}, \tau_{2,n}$	n-th mode normalized closed-loop time-constant of NLMS algorithm; "secondary" and dominant closed-loop time-constants of FRQN algorithm.
$Tr[\]$	Trace of matrix in $[\]$.
TRAAAP	Transmit-Receive Adaptive Antenna Array Processing.
θ	Average value of p_k at the k-th iteration of the FRQN algorithm.
θ_m	Azimuth angle of the m-th signal source.
U_n	n-th column of E , (i.e. the n-th orthonormal eigenvector of covariance matrix R , $n=1,\dots,N$).
$U_m^{\perp}$	m-th column of $E_S^{\perp}$, (i.e. an orthonormal eigenvector in the interference subspace, $m=2,\dots,M$).
u	Real part of independent variable z in the complex Z -plane.
V	Weight vector W transformed by eigenvector matrix E .
V_k	An intermediate vector in the computation of the k-th iteration of the RLS algorithm.
$V(s)$	Laplace transform of vector V .
$W_k; W(t)$	Vector of N complex weights of an adaptive array (spatial filter) at the k-th iteration; continuous-time weight vector.
W_{opt}	Vector of optimal steady-state adaptive array weights.

$W_{opt,k}$	Vector of optimal non-stationary adaptive array weights at k-th iteration.
$W_{I,k}^+$; $W_{I,k-1}^I$	Projected weight vector for excluding the interference contribution from the reference signal; the complementary-projected weight vector serving as the forcing function.
$W_{S,k}^+$; $W_{S,k-1}^I$	Projected weight vector for excluding the SOI contribution from the reference signal; the complementary-projected weight vector serving as the forcing function.
$W_{s,k}^+$	Projected weight vector for isolating the array-noise contribution to the reference signal at the k-th iteration.
$w_{Re\ n}$; $w_{Im\ n}$	Real part of the n-th complex weight; corresponding imaginary part.
X_k	Weight error vector at k-th iteration the mean square convergence of the NLMS algorithm. Also composite incident interference signal vector.
x_k	A scalar parameter in the mean square convergence at the k-th iteration of the FRQN algorithm.
$[]_x$	Subscript pertaining to interference-only aspects (no array noise).
Y_k	Composite incident SOI, interference + array noise vector at k-th iteration.
$y_{n,k}$	n-th entry of vector Y_k .
y_k	Available output signal of the adaptive array at the k-th iteration.
$Z[]$	Z-transform of discrete-time function (of k) in $[]$.
Z_k	Weight error vector at k-th iteration of the mean square convergence of the FRQN algorithm.
z	The independent complex variable in the Z-transform domain (Z-plane).

z_k	A scalar parameter in the mean square convergence at the k-th iteration of the FRQN algorithm.
$z_{p,1}; z_{1,n}$	Pole in the Z-plane corresponding to the first root of a quadratic characteristic polynomial; the same, pertaining to the n-th such polynomial.
z_1, z_2, z_3, z_4	Arbitrary complex random variables in a statement of the Gaussian moment-factoring theorem.
ζ_k	An intermediate scalar variable in the computation of the k-th iteration of the FRQN algorithm.
ξ	A dummy variable of integration in the continuous-time solution for the modes of the averaged LMS algorithm.
$\xi[\]$	A non-linear function of the signal in $[\]$.