Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia
COPPE UFRJ

ON DATA-SELECTIVE LEARNING

Hamed Yazdanpanah

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia Elétrica.

Orientadores: Paulo Sergio Ramirez Diniz
Markus Vinicius Santos Lima

Rio de Janeiro
Março de 2018

ON DATA-SELECTIVE LEARNING

Hamed Yazdanpanah

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Paulo Sergio Ramirez Diniz, Ph.D.

Prof. Markus Vinicius Santos Lima, D.Sc.

Prof. Marcello Luiz Rodrigues de Campos, Ph.D.

Prof. José Antonio Apolinário Jr., D.Sc.

Prof. Mário Sarcinelli Filho, D.Sc.

Prof. Cássio Guimarães Lopes, Ph.D.

RIO DE JANEIRO, RJ – BRASIL
MARÇO DE 2018

Yazdanpanah, Hamed

On Data-Selective Learning/Hamed Yazdanpanah. – Rio de Janeiro: UFRJ/COPPE, 2018.

XXVI, 149 p.: il.; 29, 7cm.

Orientadores: Paulo Sergio Ramirez Diniz

Markus Vinicius Santos Lima

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2018.

Referências Bibliográficas: p. 136 – 149.

1. Adaptive filtering. 2. Data-selective adaptive filtering. 3. Set-membership filtering. 4. Robustness. 5. Quaternion. 6. Trinion. 7. Partial-update. 8. Sparsity. 9. Feature LMS algorithm. 10. Computational complexity. I. Diniz, Paulo Sergio Ramirez *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

To my parents, Mohammad and Mina, and Ana Clara
for their love, attention, and support.

Acknowledgments

I would like to express my sincere gratitude to my advisor, Professor Paulo S. R. Diniz, for the continuous support, guidance, patience, motivation, and immense knowledge. Specially, I would like to thank him for his generous support and patience during my illness that lasted for about one year. Also, his extreme competence and friendly comprehension inspire me to be a better professional and friend. In fact, he is a remarkable example of a Brazilian. I could not have imagined having a better advisor for my Ph.D. study.

Also, I would like to thank Professor Markus V. S. Lima, my other advisor. He helped me for all details of my thesis. In fact, I am grateful for having his guidance during my study. He was always keen to know what I was doing and how I was proceeding. He always inspired me to be a serious and diligent researcher. I thank him for being not only my advisor, but also a friend.

Beside my advisors, I would like to thank my thesis committee: Prof. Marcello L. R. de Campos, Prof. José A. Apolinário Jr., Prof. Mário S. Filho, and Prof. Cássio G. Lopes for their encouragement, insightful comments and suggestions. My thesis benefited from their valuable comments. Moreover, I would like to express my sincere gratitude to Prof. José A. Apolinário Jr. for his invaluable comments on Chapter 7 of the text.

My sincere thanks also goes to Prof. Sergio L. Netto and Prof. Eduardo A. B. da Silva for offering me a research project in their group. I have learned a lot from them during the project.

I would like to thank the professors of the Programa de Engenharia Elétrica (PEE) who have contributed to my education. In particular, I am grateful to Prof. Wallace A. Martins for the courses he taught.

Also, I would like to thank the staff of the SMT Lab. I am particularly grateful to Michelle Nogueira for her support and assistance during my Ph.D. study. Moreover, I thank the university staff, in particular, Daniele C. O. da Silva and Mauricio de

Carvalho Machado for their help.

My sincere thanks also goes to Camila Gussen and all friends of the SMT Lab. They make the SMT Lab a pleasant and collaborative workplace. Also, I would like to thank Prof. Tadeu Ferreira for his special attention and help.

A very special gratitude goes out to Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), and Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ) for the financial support.

I am really grateful to my lovely girlfriend, Ana Clara, and her family for all their love, patience, and help. Her love motivates me to continue my studies in Brazil and to choose this beautiful country as my home. Her continuous encouragement, unfailing emotional support, and permanent attention played fundamental roles throughout my years of study.

I am deeply grateful to my parents for giving birth to me at the first place and supporting me spiritually throughout my life. I can never pay them back the sacrifice they made for me. My father, Mohammad Yazdanpanah, and my mother, Mina Alizadeh, have provided me through moral and emotional support during my education. Finally, I must express my very profound gratitude to my brother and my sister for providing me with support and continuous encouragement through the process of researching and writing this thesis. This accomplishment would not have been possible without my family. Thank you.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

APRENDIZADO SOB SELEÇÃO DE DADOS

Hamed Yazdanpanah

Março/2018

Orientadores: Paulo Sergio Ramirez Diniz
Markus Vinicius Santos Lima

Programa: Engenharia Elétrica

Filtros adaptativos são aplicados em diversos aparelhos eletrônicos e de comunicação, como *smartphones*, fone de ouvido avançados, DSP chips, antenas inteligentes e sistemas de teleconferência. Eles também têm aplicação em várias áreas como identificação de sistemas, equalização de canal, cancelamento de eco, cancelamento de interferência, previsão de sinal e mercado de ações. Desse modo, reduzir o consumo de energia de algoritmos adaptativos tem importância significativa, especialmente em tecnologias verdes e aparelhos que usam bateria.

Nesta tese, filtros adaptativos com seleção de dados, em particular filtros adaptativos da família *set-membership* (SM), são apresentados para cumprir essa missão. No presente trabalho objetivamos apresentar novos algoritmos, baseados nos clássicos, a fim de aperfeiçoar seus desempenhos e, ao mesmo tempo, reduzir o número de operações aritméticas exigidas. Dessa forma, primeiro analisamos a robustez dos filtros adaptativos SM clássicos. Segundo, estendemos o SM aos números trinions e quaternions. Terceiro, foram utilizadas também duas famílias de algoritmos, SM filtering e *partial-updating*, de uma maneira elegante, visando reduzir energia ao máximo possível e obter um desempenho competitivo em termos de estabilidade. Quarto, a tese propõe novos filtros adaptativos baseado em algoritmos *least-mean-square* (LMS) e mínimos quadrados recursivos com complexidade computacional baixa para espaços esparsos. Finalmente, derivamos alguns algoritmos *feature* LMS para explorar a esparsidade escondida nos parâmetros.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

ON DATA-SELECTIVE LEARNING

Hamed Yazdanpanah

March/2018

Advisors: Paulo Sergio Ramirez Diniz
Markus Vinicius Santos Lima

Department: Electrical Engineering

Adaptive filters are applied in several electronic and communication devices like smartphones, advanced headphones, DSP chips, smart antenna, and teleconference systems. Also, they have application in many areas such as system identification, channel equalization, noise reduction, echo cancellation, interference cancellation, signal prediction, and stock market. Therefore, reducing the energy consumption of the adaptive filtering algorithms has great importance, particularly in green technologies and in devices using battery.

In this thesis, data-selective adaptive filters, in particular the set-membership (SM) adaptive filters, are the tools to reach the goal. There are well known SM adaptive filters in literature. This work introduces new algorithms based on the classical ones in order to improve their performances and reduce the number of required arithmetic operations at the same time. Therefore, firstly, we analyze the robustness of the classical SM adaptive filtering algorithms. Secondly, we extend the SM technique to trinion and quaternion systems. Thirdly, by combining SM filtering and partial-updating, we introduce a new improved set-membership affine projection algorithm with constrained step size to improve its stability behavior. Fourthly, we propose some new least-mean-square (LMS) based and recursive least-squares based adaptive filtering algorithms with low computational complexity for sparse systems. Finally, we derive some feature LMS algorithms to exploit the hidden sparsity in the parameters.

Contents

List of Figures

List of Tables

List of Symblos

$(\cdot)^*$	Conjugation operator, p. 48
$(\cdot)^H$	Hermitian transposition of $(\cdot)$, p. 6
$(\cdot)^T$	Transposition of $(\cdot)$, p. 6
$B(\theta)$	The beam pattern of a beamformer, p. 67
G_β	Continuous and almost everywhere differentiable function that approximates the l_0 norm; β controls the quality of the approximation, p. 83
L	Data reuse factor, p. 14
N	Order of the FIR adaptive filter, p. 14
$S(k)$	The hypersphere in $\mathbb{R}^{N+1}$ centered at $\mathbf{w}(k)$ with the radius $\mu(k)$, p. 75
$\mathbf{A}(k)$	Auxiliary matrix $\mathbf{A}(k) \triangleq (\mathbf{X}^T(k)\mathbf{X}(k))^{-1}$, p. 15
$\mathbf{C}_{\mathcal{I}_M(k)}$	The diagonal matrix that identifies the coefficients to be updated at instant time k, if an update is required, p. 73
$\mathbf{F}(k)$	Feature matrix, p. 113
$\mathbf{F}_\epsilon(\mathbf{w})$	The Jacobian matrix of $\mathbf{f}_\epsilon(\mathbf{w})$, p. 88
$\mathbf{F}_h$	Feature matrix for systems with highpass narrowband spectrum, p. 117
$\mathbf{F}_l$	Feature matrix for systems with lowpass narrowband spectrum, p. 116

$\mathbf{I}$	Identity matrix, p. 6
$\mathbf{P}(k)$	The auxiliary matrix $\mathbf{P}(k) \triangleq (\mathbf{X}^T(k)\mathbf{C}_{\mathcal{I}_M(k)}\mathbf{X}(k) + \delta\mathbf{I})^{-1}$, p. 73
$\mathbf{R}$	Correlation matrix, p. 8
$\mathbf{R}_D(k)$	Deterministic correlation matrix of the input signal, p. 10
$\mathbf{R}_{D,\epsilon}(k)$	The deterministic correlation matrix of the input signal involved $\mathbf{F}_\epsilon(\mathbf{w}(k))$, p. 96
$\Re(\cdot)$	The real part of $(\cdot)$, p. 52
$\mathbf{S}_D(k)$	The inverse of $\mathbf{R}_D(k)$, p. 10
$\mathbf{S}_{D,\epsilon}(k)$	The inverse of $\mathbf{R}_{D,\epsilon}(k)$, p. 96
Θ	Feasibility set, p. 11
$\mathbf{X}(k)$	Input signal matrix, p. 14
$\bar{\imath}$	The first orthogonal unit imaginary axis vector in trinion numbers, p. 52
$\bar{\jmath}$	The second orthogonal unit imaginary axis vector in trinion numbers, p. 52
$\mathbf{d}(k)$	Desired signal vector, p. 14
δ	Regularization factor, p. 9
$\mathbf{e}(k)$	Error signal vector, p. 14
$\widetilde{\mathbf{e}}(k)$	Noiseless error signal vector, p. 31
$\emptyset$	Empty set, p. 25
$\widetilde{e}(k)$	Noiseless error signal, p. 16
$\mathbf{f}_\epsilon(\cdot)$	Discard vector function, p. 87
$\overline{\gamma}$	Upper bound for the magnitude of the error signal, p. 11
$\overline{\gamma}(k)$	Time-varying error bound, p. 30

$\boldsymbol{\gamma}(k)$	Constraint vector, p. 14
$\mathbf{g}_\beta(\mathbf{w})$	Gradient of $G_\beta(\mathbf{w})$ with respect to $\mathbf{w}$, p. 84
$\imath$	The first orthogonal unit imaginary axis vector in quaternion numbers, p. 48
$\jmath$	The second orthogonal unit imaginary axis vector in quaternion numbers, p. 48
κ	The third orthogonal unit imaginary axis vector in quaternion numbers, p. 48
$\mathbb{C}$	Set of complex numbers, p. 6
$\mathbb{E}$	Expected value operator, p. 6
$\mathbb{F}_\epsilon^a$	Alternative feature function, p. 122
$\mathbb{F}_\epsilon$	Feature function, p. 119
$\mathbb{H}$	Set of quaternion numbers, p. 6
$\mathbb{N}$	Set of natural numbers, p. 6
$\mathbb{P}$	Probability operator, p. 6
$\mathbb{R}$	Set of real numbers, p. 6
$\mathbb{R}_+$	Set of nonnegative real numbers, p. 6
$\mathbb{T}$	Set of trinion numbers, p. 6
$\mathbb{Z}$	Set of integer numbers, p. 6
μ	Convergence factor, p. 8
$\mathbf{n}(k)$	Additive noise signal vector, p. 14
$\mathbf{p}(k)$	Gradient of $\mathcal{P}\left(\mathbf{F}(k)\mathbf{w}(k)\right)$, p. 115
$\mathbf{p}_D(k)$	Deterministic cross-correlation vector between the input and the desired signals, p. 10

$\mathbf{p}_{D,\epsilon}(k)$	The deterministic cross-correlation vector between the input and the desired signals involved $\mathbf{F}_\epsilon(\mathbf{w}(k))$, p. 96
$\psi(k)$	Exact membership set, p. 11
$\psi^{L+1}(k)$	The intersection of the $L+1$ last constraint sets, p. 54
$\mathbf{s}_c(\theta, \phi)$	The spatial steering vector for a far-field incident signal in adaptive beamforming, p. 62
σ_e^2	Variance of the error signal, p. 18
σ_n^2	Variance of the noise signal, p. 17
$\triangleq$	Definition, p. 6
$\varepsilon(k)$	*A posteriori* error signal, p. 10
$\mathbf{w}(k)$	Coefficient vector, p. 8
$\mathbf{w}_o$	Impulse response of the unknown system, p. 14
$\widetilde{\mathbf{w}}(k)$	Auxiliary vector $\widetilde{\mathbf{w}}(k) \triangleq \mathbf{w}_o - \mathbf{w}(k)$, p. 23
$\mathbf{x}(k)$	Input signal vector, p. 8
$\mathbf{y}(k)$	Output signal vector, p. 14
$d(k)$	Desired signal, p. 8
$e(k)$	Error signal, p. 8
$f_\epsilon(\cdot)$	Discard function; ϵ defines what is considered as close to zero, p. 87
k	Iteration counter, p. 8
$n(k)$	Noise signal, p. 16
q_a	The real component of a quaternion q, p. 48
q_b	The first imaginary component of a quaternion q, p. 48
q_c	The second imaginary component of a quaternion q, p. 48

q_d	The third imaginary component of a quaternion q, p. 48
v_a	The real component of a trinion v, p. 52
v_b	The first imaginary component of a trinion v, p. 52
v_c	The third imaginary component of a trinion v, p. 52
$y(k)$	Output signal, p. 8
$\mathbf{0}$	Zero vector or zero matrix, p. 6
$\mathcal{H}(k)$	Constraint set at iteration k, p. 11
$\mathcal{I}_M(k)$	The set of M coefficients to be updated at time instant k, p. 72
$\mathcal{K}_{\rm up}$	Set containing the iteration indexes in which $\mathbf{w}(k)$ is updated, p. 25
$\mathcal{P}(\cdot)$	Sparsity-promoting penalty function, p. 114
$\mathcal{S}$	Set comprised of all possible pairs $(\mathbf{x}, d)$, p. 11
Var	Variance operator, p. 17
$\text{diag}(\mathbf{x})$	Diagonal matrix with $\mathbf{x}$ on its diagonal, p. 6
$\text{erfc}(\cdot)$	The complementary error function, p. 28
$\text{sgn}(\cdot)$	The sign function, p. 85
$\text{tr}(\cdot)$	Trace of matrix, p. 6

List of Abreviations

l_0-RLS	l_0 Norm RLS, p. 5, 82, 83, 99–101, 108–112
A-l_0-RLS	Alternative l_0-RLS, p. 101, 108
AI-LCF-LMS	Alternative I-LCF-LMS, p. 123
ALCF-LMS	Alternative Low-Complexity Feature LMS, p. 122, 123
AP	Affine Projection, p. 8, 9, 15, 18, 19, 41–43, 45, 58, 59, 71, 81, 92, 93, 103, 106
AR	Autoregressive, p. 104, 106
AS-RLS	Alternative S-RLS, p. 96, 98, 100, 108
ASVB-L	Adaptive Sparse Variational Bayes Iterative Scheme Based on Laplace Prior, p. 103, 108, 110–112
BPSK	Binary Phase-Shift Keying, p. 77, 103, 106
CV	Constraint Vector, p. 14, 20, 35, 36, 39–43, 45
D-SM-AP	Discard SM-AP, p. 94
DOA	Direction of Arrival, p. 62, 67
DS-l_0-RLS	Data-Selective l_0-RLS, p. 82, 101, 108–110
DS-ASVB-L	Data-Selective ASVB-L, p. 108, 110, 111
DS-S-RLS	Data-Selective S-RLS, p. 82, 94, 98, 108, 110, 111
EMSE	Excess of the Steady-State Mean-Square Error, p. 17
F-LMS	Feature LMS, p. 5, 6, 113–118, 126–128, 132

GMF	Geman-McClure Function, p. 84, 104, 109
I-LCF-LMS	Improved LCF-LMS, p. 121
I-SM-PUAP	Improved SM-PUAP, p. 5, 71, 74, 75, 77–80
IS-AP	Improved S-AP, p. 82, 87, 92, 106
IS-SM-AP	Improved S-SM-AP, p. 5, 82, 92–94, 103–105, 112
LCF-LMS	Low-Complexity Feature LMS, p. 118, 120, 122, 123
LF	Laplace Function, p. 84
LHS	Left-Hand Side, p. 34, 37
LMS	Least-Mean-Square, p. 5, 8–10, 18, 54, 59, 81, 93, 113–116, 126–128, 132
MSE	Mean-Squared Error, p. 19, 42, 43, 79, 108, 114, 126–128, 132
NLMS	Normalized LMS, p. 8, 9, 12, 13, 18–20, 27, 38, 39, 44, 47, 54, 59, 71, 81, 86, 103, 104
OSDR	Output Signal to Desired Plus Noise Ratio, p. 67
OSIR	Output Signal to Interference Plus Noise Ratio, p. 67
PAPA	Proportionate Affine Projection Algorithm, p. 86
PASTd	Projection Approximation Subspace Tracking with Deflation, p. 88, 100
PNLMS	Proportionate Normalized LMS, p. 86
PU	Partial-Update, p. 70, 71
QAP	Quaternion-Valued Affine Projection, p. 47, 60, 61, 64, 67
QLMS	Quaternion-Valued LMS, p. 47, 64, 67
QNLMS	Quaternion-Valued Normalized LMS, p. 60, 64, 67
RHS	Right-Hand Side, p. 31, 35, 37

RLS	Recursive Least-Squares, p. 5, 8–10, 18, 81–83, 94, 96, 98, 102, 103, 108, 110–112
S-AP	Simple AP, p. 82, 87, 92
S-RLS	RLS Algorithm for Sparse System, p. 5, 82, 83, 94–96, 98, 99, 108, 110–112
S-SM-AP	Simple SM-AP, p. 5, 82, 83, 87–94, 103, 112
SC-CV	Simple Choice CV, p. 35, 39–43
SIR	Signal-to-Interference Ratio, p. 67
SM-AP	Set-Membership Affine Projection, p. 4, 5, 10, 12, 14–16, 18–22, 30, 31, 33–36, 39–45, 54, 59, 71, 74, 79, 89–91
SM-NLMS	Set-Membership Normalized LMS, p. 4, 10, 12, 13, 16, 18–29, 34–36, 38, 39, 41, 44, 45, 59, 71
SM-PAPA	Set-Membership Proportionate AP Algorithm, p. 83, 86, 90, 104–106, 112
SM-PUAP	Set-Membership Partial-Update AP, p. 5, 71–75, 77–80
SMF	Set-Membership Filtering, p. 1, 2, 4, 7, 8, 10, 11, 18, 47, 53, 70, 71, 82, 87, 108, 112
SMQAP	Set-Membership Quaternion-Valued AP, p. 4, 47, 48, 59, 60, 67
SMQNLMS	Set-Membership Quaternion-Valued NLMS, p. 4, 47, 59, 60, 67
SMTAP	Set-Membership Trinion-Valued AP, p. 4, 47, 48, 54, 57, 64
SMTNLMS	Set-Membership Trinion-Valued NLMS, p. 4, 47, 58, 59, 64
SM	Set-Membership, p. 16, 19
SNR	Signal-to-Noise Ratio, p. 36, 67, 77, 103, 109, 126
SSM-AP	Sparsity-Aware SM-AP, p. 83, 85, 90, 104–106
TAP	Trinion-Valued Affine Projection, p. 58, 61, 64

TLMS	Trinion-Valued LMS, p. 47, 64
TNLMS	Trinion-Valued Normalized LMS, p. 59, 64
ULA	Uniform Linear Array, p. 62

Chapter 1

Introduction

In the last decades, the volume of data to be processed and kept for storage has been proliferated, mainly due to the increased availability of low-cost sensors and storage devices. As examples, we can mention the usage of multiple antennas in multiple-input and multiple-output wireless communication systems, the application of multiple audio devices in speech enhancement and audio signal processing, and the employment of echo cancellers in small or handheld communication devices. Moreover, these technological features are continuously spreading.

Our world is overwhelmed by data and to benefit from them in our daily life, we need to process the data correctly. A significant amount of data, however, brings about no new information in order that only part of it is particularly useful [4, 5]. Therefore, we are compelled to improve our ability to evaluate the importance of the received data. This capability is called *data selection*. It enables the derivation of *data-selective adaptive filters*, which can neglect undesired data in a smart way. These filters are designed to reject the redundant data and perform their modeling tasks utilizing a small fraction of the available data.

Data-selective adaptive filters evaluate, select, and process data at each iteration of their learning process. These filters assess the data and choose only the ones bringing about some innovation. This property of the data-selective adaptive filters distinguishes them from the family of classical adaptive filters, which consider all data. In particular, these data-selective adaptive filters improve the accuracy of the estimator and decrease the computational complexity at the same time [6–8].

In this thesis, to apply the data selection, we employ the *set-membership filtering* (SMF) approach [2, 9]. The set-membership (SM) adaptive filtering algorithm aims at estimating the system such that the magnitude of the estimation output error is upper bounded by a predetermined positive constant called the threshold. The threshold is usually chosen based on *a priori* information about the sources of uncertainty. A comparison between traditional and SM adaptive filters was performed in [1, 2], where the results had shown that the algorithms employing the SMF strategy require lower computational resources as compared to the conventional adaptive filters. The SMF algorithms, however, are not so widely used since there is some lack of analysis tools, and there is a limited number of set-membership adaptive filtering algorithms available. This thesis introduces new algorithms employing the SMF approach and provides some analysis tools.

This chapter is organized as follows. Section 1.1 contains the main motivations. The targets of this thesis are given in Section 1.2. Section 1.3 describes the contributions of this thesis. Finally, the notation is explained in Section 1.4.

1.1 Motivations

The area of *Digital Signal Processing* takes part in our daily lives for decades now, since it is at the core of virtually all electronic gadget we have been utilizing, ranging from medical equipment to mobile phones. If we have full information about the signals, we can apply the most suitable algorithm (a digital filter for instance) to process the signals. However, if we do not know the statistical properties of the signals, a possible solution is to utilize an adaptive filter that automatically modifies its characteristics to match the behavior of the observed data.

Adaptive filters [2, 10, 11] are utilized in several electronic and communication devices, such as smartphones, advanced headphones, DSP chips, smart antennas, and microphone arrays for teleconference systems. Also, they have application in many areas such as system identification [12], channel equalization [13], noise reduction [14], echo cancellation [15], interference cancellation [16], signal prediction [17], acoustic images [18], stock market [19], etc. Due to the diversity of applications of adaptive signal processing, traditional adaptive filters cannot meet the needs of every application. An ideal adaptive filter would have low processing time, high accuracy in the learning process, low energy consumption, low memory usage, etc. These properties, however, conflict with each other.

An adaptive filter uses an algorithm to adjust its coefficients. An algorithm is a procedure to modify the coefficients in order to minimize a prescribed criterion. The algorithm is characterized by defining the search method, the objective function, and the error signal nature. The traditional algorithms in adaptive filtering implement coefficient updates at each iteration. However, when the adaptive filter learns from the observed data and reaches its steady state, it is desirable that the adaptive filter has the ability to reduce its energy consumption since there is less information to be learned. Here appears the importance of data-selective adaptive filters since they assess the input data, then according to the innovation they decide to perform an update or not.

After defining the set-membership adaptive filtering algorithms as a subset of the data-selective adaptive filters, many works have shown how effective these algorithms are in reducing the energy consumption. In some environments they can decrease the number of updates by 80% [1, 2]. This thesis, however, shows that there is room for improvements regarding the reduction in the number of arithmetic operations and energy consumption, as discussed in Chapters 5 and 6.

1.2 Targets

The targets of this thesis are:

- To analyze the performance of some existing set-membership adaptive filtering algorithms to confirm their competitive performance as compared to the classical adaptive filtering approaches;
- To develop data-selective adaptive filtering algorithms beyond the real and complex numbers, and examine the advantage of the set-membership technique in different mathematical number systems;
- To improve some existing set-membership adaptive filtering algorithms to bring about improvements in performance and computational complexity;
- To introduce some new sparsity-aware set-membership adaptive filtering algorithms with low computational burden;
- To exploit the hidden sparsity in the linear combination of parameters of adaptive filters.

In a nutshell, in this thesis, we improve and analyze data-selective adaptive filtering algorithms.

1.3 Thesis Contributions

In this thesis, we analyze the robustness of classical set-membership adaptive filtering algorithms and extend these conventional algorithms for the trinion and the quaternion systems. In addition, we introduce an improved version of a set-membership adaptive filtering algorithm along with the partial updating strategy. Moreover, we develop some algorithms for sparse systems utilizing the SMF technique. Finally, we try to exploit the hidden sparsity in systems with lowpass and highpass frequencies. To address such topics, the text is hereinafter organized as follows.

Chapter 2 introduces some conventional adaptive filtering algorithms, such as the least-mean-square (LMS), the normalized LMS (NLMS), the affine projection (AP), and the recursive least-squares (RLS) ones. Then, we review the set estimation theory in adaptive signal processing and presents the set-membership filtering (SMF) strategy. Also, we describe a short review of the set-membership normalized least-mean-square (SM-NLMS) and the set-membership affine projection (SM-AP) algorithms.

In Chapter 3, we address the robustness, in the sense of l_2-stability, of the SM-NLMS and the SM-AP algorithms. For the SM-NLMS algorithm, we demonstrate that it is robust regardless the choice of its parameters and that the SM-NLMS enhances the parameter estimation in most of the iterations in which an update occurs, two advantages over the classical NLMS algorithm. Moreover, we also prove that if the noise bound is known, then we can set the SM-NLMS so that it never degrades the estimate. As for the SM-AP algorithm, we demonstrate that its robustness depends on a judicious choice of one of its parameters: the constraint vector (CV). We prove the existence of CVs satisfying the robustness condition, but practical choices remain unknown. We also demonstrate that both the SM-AP and the SM-NLMS algorithms do not diverge, even when their parameters are selected naively, provided the additional noise is bounded. Furthermore, numerical results that corroborate our analyses are presented.

In Chapter 4, we introduce new data-selective adaptive filtering algorithms for trinion and quaternion systems $\mathbb{T}$ and $\mathbb{H}$. The work advances the set-membership trinion- and quaternion-valued normalized least-mean-square (SMTNLMS and SMQNLMS) and the set-membership trinion- and quaternion-valued affine projection (SMTAP and SMQAP) algorithms. Also, as special cases, we obtain trinion- and quaternion-valued algorithms not employing the set-membership strategy. Prediction simulations based on recorded wind data are provided, showing the improved performance of the proposed algorithms regarding reduced computational load. Moreover, we study the application of quaternion-valued adaptive filtering algorithms to adaptive beamforming.

Usually, set-membership algorithms implement updates more regularly during the early iterations in stationary environments. Therefore, if these updates exhibit high computational complexity, an alternative solution is needed. A possible approach to partly control the computational complexity is to apply partial update technique, where only a subset of the adaptive filter coefficients is updated at each iteration. In Chapter 5, we present an improved set-membership partial-update affine projection (I-SM-PUAP) algorithm, aiming at accelerating the convergence rate, and decreasing the update rate of the set-membership partial-update affine projection (SM-PUAP) algorithm. To meet these targets, we constrain the weight vector perturbation to be bounded by a hypersphere instead of the threshold hyperplanes as in the standard algorithm. We use the distance between the present weight vector and the expected update in the standard SM-AP algorithm to construct the hypersphere. Through this strategy, the new algorithm shows better behavior in the early iterations. Simulation results verify the excellent performance of the proposed algorithm related to the convergence rate and the required number of updates.

In Chapter 6, we derive two LMS-based algorithms, namely the simple set-membership affine projection (S-SM-AP) and the improved S-SM-AP (IS-SM-AP), in order to exploit the sparsity of an unknown system while focusing on having low computational cost. To achieve this goal, the proposed algorithms apply a discard function on the weight vector to disregard the coefficients close to zero during the update process. In addition, the IS-SM-AP algorithm reduces the overall number of computations required by the adaptive filter even further by replacing small coefficients with zero. Moreover, we introduce the l_0 norm RLS (l_0-RLS) and the RLS algorithm for sparse models (S-RLS). Also, we derive the data-selective version of these RLS-based algorithms. Simulation results show similar performance when comparing the proposed algorithms with some existing state-of-the-art sparsity-aware algorithms while the proposed algorithms require lower computational complexity.

When our target is to detect and exploit sparsity in the model parameters, in many situations, the sparsity is hidden in the relations among these coefficients so that some suitable tools are required to reveal the potential sparsity. Chapter 7 proposes a set of least-mean-square (LMS) type algorithms, collectively called feature LMS (F-LMS) algorithms, setting forth a hidden feature of the unknown parameters, which ultimately would improve convergence speed and steady-state mean-squared error. The fundamental idea is to apply linear transformations, by means of the so-called feature matrices, to reveal the sparsity hidden in the coefficient vector, followed by a sparsity-promoting penalty function to exploit such sparsity. Some F-LMS algorithms for lowpass and highpass systems are also introduced by using simple feature matrices that require only trivial operations. Simulation results demonstrate that the proposed F-LMS algorithms bring about several performance improvements whenever the hidden sparsity of the parameters is exposed.

Finally, chapter 8 highlights the conclusions of the work, and gives some clues for future works regarding the topics addressed in the thesis.

1.4 Notation

In this section, we introduce most of the usual notation utilized in this thesis. However, in order to avoid confusing the reader, we evade presenting here the definition of the rare notation in this text, and we introduce them only at the vital moments.

Equalities are shown by $=$, and when they refer to a definition, we use $\triangleq$. The real, nonnegative real, nature, integer, complex, trinion, and quaternion numbers are denoted by $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{C}$, $\mathbb{T}$, and $\mathbb{H}$, respectively.

Moreover, scalars are represented by lowercase letters (e.g., x), vectors by lowercase boldface letters (e.g., $\mathbf{x}$), and matrices by uppercase boldface letters (e.g., $\mathbf{X}$). The symbols $(\cdot)^T$ and $(\cdot)^H$ stand for the transposition and Hermitian operators, respectively. Also, all vectors are column vectors in order that the inner product between two vectors $\mathbf{x}$ and $\mathbf{y}$ is defined as $\mathbf{x}^T\mathbf{y}$ or $\mathbf{x}^H\mathbf{y}$.

We represent the trace operator by $\text{tr}(\cdot)$. The identity matrix and zero vector (matrix) are denoted by $\mathbf{I}$ and $\mathbf{0}$, respectively. Also, $\text{diag}(\mathbf{x})$ stands for a diagonal matrix with vector $\mathbf{x}$ on its diagonal and zero outside it. Furthermore, $\mathbb{P}[\cdot]$ and $\mathbb{E}[\cdot]$ denote the probability and the expected value operators, respectively. Also, $\|\cdot\|$ denotes the l_2 norm (when the norm is not defined explicitly, we are referring to the l_2 norm).

Chapter 2

Conventional and Set-Membership Adaptive Filtering Algorithms

The *point estimation theory* [20] utilizes a sample data for computing a single solution as the best estimate of an unknown parameter. For decades, machine learning and adaptive filtering have been grounded in the point estimation theory [2, 10, 11, 21, 22]. Nowadays, the benefit of the set estimation approach, however, is becoming clearer by disclosing its advantages [23–25].

In contrast with the world of theoretical models, in the real-world we live with uncertainties originate from measurement noise, quantization, interference, modeling errors, etc. Therefore, searching the solution utilizing point estimation theory sometimes results in a waste of energy and time. An alternative is to address the problem from the *set estimation theory* [23] point of view. In fact, in this approach, we search for a set of acceptable solutions instead of a unique point as a solution.

The adaptive filtering algorithms presented in [10, 11] exhibit a trade-off between convergence rate and misadjustment after transient, particularly in stationary environments. In general, fast converging algorithms lead to high variance estimators after convergence. To tackle this problem, we can apply set-membership filtering (SMF) [1, 2] which is a representative of the set estimation theory. The SMF technique prevents unnecessary updates and reduces the computational complexity by updating the filter coefficients only when the estimation error is greater than a predetermined upper bound [9, 26, 27].

In set-membership adaptive filters, we try to find a *feasibility set* such that any member in this set has the output estimation error limited by a predetermined upper bound. For this purpose, the objective function of the algorithm is related to a bounded error constraint on the filter output, such that the updates are contained in a set of acceptable solutions. The inclusion of *a priori* information, such as the noise bound, into the objective function leads to some noticeable advantages. As compared with the normalized least-mean-square (NLMS) and the affine projection (AP) algorithms, their set-membership counterparts have lower computational cost, better accuracy, data selection, and robustness against noise [6, 7, 9, 28–31].

This chapter presents a brief review of some adaptive filtering algorithms. An interested reader should refer to [2] for more details. Section 2.1 describes the point estimation adaptive filtering algorithms. Section 2.2 reviews the SMF approach and the main set-membership algorithms. The estimation of the threshold parameter for big data applications is discussed in Section 2.3. Finally, Section 2.4 contains the conclusions.

2.1 Point Estimation Adaptive Filtering Algorithms

In this section, we introduce some LMS-based adaptive filtering algorithms and the recursive least-squares (RLS) algorithm.

2.1.1 Least-mean-square algorithm

The update equation of the least-mean-square (LMS) algorithm is given by [2]

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu e(k)\mathbf{x}(k), \tag{2.1}$$

where $\mathbf{x}(k) = [x_0(k)\ x_1(k)\ \cdots\ x_N(k)]^T$ and $\mathbf{w}(k) = [w_0(k)\ w_1(k)\ \cdots\ w_N(k)]^T$ are the input signal vector and the the weight vector, respectively. The output signal is defined by $y(k) \triangleq \mathbf{w}^T(k)\mathbf{x}(k) = \mathbf{x}^T(k)\mathbf{w}(k)$, and $e(k) \triangleq d(k) - y(k)$ denotes the error signal, where $d(k)$ is the desired signal. The convergence factor μ should be chosen in the range $0 < \mu < \frac{1}{\text{tr}[\mathbf{R}]}$ to guarantee the convergence, where $\mathbf{R} \triangleq \mathbb{E}[\mathbf{x}(k)\mathbf{x}^T(k)]$ is the correlation matrix.

2.1.2 Normalized LMS algorithm

To increase the convergence rate of the LMS algorithm without using matrix $\mathbf{R}$, we can utilize the NLMS algorithm. The recursion rule of the NLMS algorithm is described by [2]

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \frac{\mu_n}{\mathbf{x}^T(k)\mathbf{x}(k)+\delta} e(k)\mathbf{x}(k), \tag{2.2}$$

where δ is a small regularization factor, and the step size μ_n should be selected in the range $0 < \mu_n < 2$.

2.1.3 Affine projection algorithm

When the input signal is correlated, it is possible to use old data signal to improve the convergence speed of the algorithm. For this purpose, let us utilize the last $L+1$ input signal vector and form matrix $\mathbf{X}(k)$ as

$$\mathbf{X}(k) = [\mathbf{x}(k)\ \mathbf{x}(k-1)\ \cdots\ \mathbf{x}(k-L)] \in \mathbb{R}^{(N+1)\times(L+1)}. \tag{2.3}$$

Also, let us define the desired signal vector $\mathbf{d}(k)$, the output signal vector $\mathbf{y}(k)$, and the error signal vector $\mathbf{e}(k)$ as follows

$$\begin{aligned}
\mathbf{d}(k) &= [d(k)\ d(k-1)\ \cdots\ d(k-L)]^T, \\
\mathbf{y}(k) &\triangleq \mathbf{w}^T(k)\mathbf{X}(k) = \mathbf{X}^T(k)\mathbf{w}(k), \\
\mathbf{e}(k) &\triangleq \mathbf{d}(k) - \mathbf{y}(k).
\end{aligned} \tag{2.4}$$

Then, the update rule of the affine projection (AP) algorithm is described by [2]

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu\mathbf{X}(k)[\mathbf{X}^T(k)\mathbf{X}(k)]^{-1}\mathbf{e}(k), \tag{2.5}$$

where μ is the convergence factor.

2.1.4 Recursive least-squares algorithm

Here, we review the RLS algorithm. The goal of this algorithm is to match the output signal to the desired signal as much as possible.

The objective function of the RLS algorithm is given by

$$\zeta(k) = \sum_{i=0}^{k} \lambda^{k-i} \varepsilon^2(i) = \sum_{i=0}^{k} \lambda^{k-i} [d(i) - \mathbf{x}^T(i)\mathbf{w}(k)]^2, \tag{2.6}$$

where λ is a forgetting factor which should be adopted in the range $0 \ll \lambda \leq 1$, and $\varepsilon(i)$ is called the *a posteriori* error. Note that in the elaboration of the LMS-based algorithms we use the *a priori* error, whereas for the RLS algorithm we utilize the *a posteriori* error.

If we differentiate $\zeta(k)$ with respect to $\mathbf{w}(k)$ and equate the result to zero, we get the optimal coefficient vector $\mathbf{w}(k)$ [2]

$$\mathbf{w}(k) = \Big[\sum_{i=0}^{k} \lambda^{k-i} \mathbf{x}(i)\mathbf{x}^T(i)\Big]^{-1} \sum_{i=0}^{k} \lambda^{k-i} \mathbf{x}(i) d(i) = \mathbf{R}_D^{-1}(k)\mathbf{p}_D(k), \tag{2.7}$$

where $\mathbf{R}_D(k)$ and $\mathbf{p}_D(k)$ are named the deterministic correlation matrix of the input signal and the deterministic cross-correlation vector between the input and the desired signals, respectively. By using the matrix inversion lemma [32], the inverse of $\mathbf{R}_D(k)$ can be given by

$$\mathbf{S}_D(k) = \mathbf{R}_D^{-1}(k) = \frac{1}{\lambda}\Big[\mathbf{S}_D(k-1) - \frac{\mathbf{S}_D(k-1)\mathbf{x}(k)\mathbf{x}^T(k)\mathbf{S}_D(k-1)}{\lambda + \mathbf{x}^T(k)\mathbf{S}_D(k-1)\mathbf{x}(k)}\Big]. \tag{2.8}$$

2.2 Set-Membership Adaptive Filtering Algorithms

In this section, we firstly introduce the set-membership filtering (SMF) approach. Secondly, we present the SM-NLMS algorithm. Finally, we review the SM-AP algorithm.

2.2.1 Set-membership filtering

The SMF approach proposed in [9] is suitable for adaptive filtering problems that are linear in parameters. Thus, for a given input signal vector $\mathbf{x}(k) \in \mathbb{R}^{N+1}$ at iteration k and the filter coefficients $\mathbf{w} \in \mathbb{R}^{N+1}$, the output signal of the filter is obtained by

$$y(k) = \mathbf{w}^T\mathbf{x}(k), \tag{2.9}$$

where $\mathbf{x}(k) = [x_0(k)\ x_1(k)\ \cdots\ x_N(k)]^T$ and $\mathbf{w} = [w_0\ w_1\ \cdots\ w_N]^T$. For a desired signal sequence $d(k)$, the estimation error sequence $e(k)$ is computed as

$$e(k) = d(k) - y(k). \tag{2.10}$$

The SMF criterion aims at estimating the parameter $\mathbf{w}$ such that the magnitude of the estimation output error is upper bounded by a constant $\overline{\gamma} \in \mathbb{R}_+$, for all possible pairs $(\mathbf{x}, d)$. If the value of $\overline{\gamma}$ is suitably selected, there are various valid estimates for $\mathbf{w}$. The threshold is usually chosen based on *a priori* information about the sources of uncertainty. Note that any $\mathbf{w}$ leading to an output estimation error with magnitude smaller than $\overline{\gamma}$ is an acceptable solution. Hence, we obtain a set of filters rather than a single estimate.

Let us denote by $\mathcal{S}$ the set comprised of all possible pairs $(\mathbf{x}, d)$. We want to find $\mathbf{w}$ such that $|e| = |d - \mathbf{w}^T\mathbf{x}| \leq \overline{\gamma}$ for all $(\mathbf{x}, d) \in \mathcal{S}$. Therefore, the *feasibility set* Θ will be defined as

$$\Theta \triangleq \bigcap_{(\mathbf{x},d)\in\mathcal{S}} \{\mathbf{w} \in \mathbb{R}^{N+1} : |d - \mathbf{w}^T\mathbf{x}| \leq \overline{\gamma}\}, \tag{2.11}$$

so that the SMF criterion can be stated as finding $\mathbf{w} \in \Theta$.

In the case of online applications, we do not have access to all members of $\mathcal{S}$. Thus, we consider the practical case in which only measured data are available and develop iterative techniques. Suppose that a set of data pairs $\{(\mathbf{x}(0), d(0)), (\mathbf{x}(1), d(1)), \cdots, (\mathbf{x}(k), d(k))\}$ is available, and define the *constraint set* $\mathcal{H}(k)$ at time instant k as

$$\mathcal{H}(k) \triangleq \{\mathbf{w} \in \mathbb{R}^{N+1} : |d(k) - \mathbf{w}^T\mathbf{x}(k)| \leq \overline{\gamma}\}. \tag{2.12}$$

Also, define the *exact membership set* $\psi(k)$ as the intersection of the constraint sets from the beginning, i.e. the first iteration, to iteration k, or

$$\psi(k) \triangleq \bigcap_{i=0}^{k} \mathcal{H}(i). \tag{2.13}$$

Then, Θ can be iteratively estimated via the exact membership set since $\lim_{k\to\infty} \psi(k) = \Theta$.

Figure 2.1 shows the geometrical interpretation of the SMF principle. The bound-

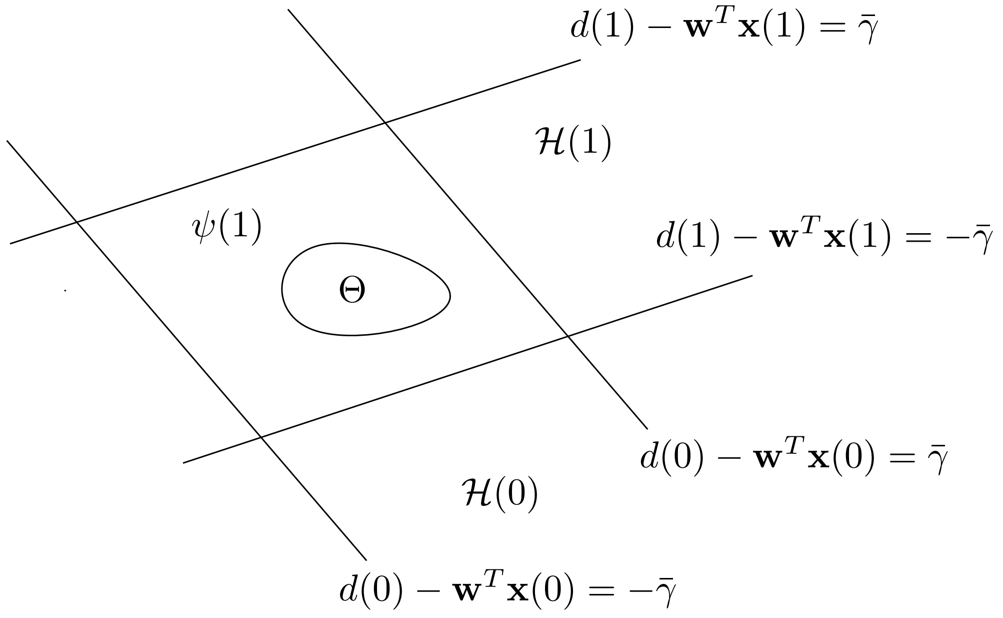

$d(1) - \mathbf{w}^T\mathbf{x}(1) = \bar{\gamma}$
$\mathcal{H}(1)$
$\psi(1)$
$d(1) - \mathbf{w}^T\mathbf{x}(1) = -\bar{\gamma}$
Θ
$d(0) - \mathbf{w}^T\mathbf{x}(0) = \bar{\gamma}$
$\mathcal{H}(0)$
$d(0) - \mathbf{w}^T\mathbf{x}(0) = -\bar{\gamma}$

Figure 2.1: SMF geometrical interpretation in the parameter space $\psi(1)$ (redrawn from [1]).

aries of the constraint sets are hyperplanes, and $\mathcal{H}(k)$ corresponds to region between the parallel hyperplanes in the parameter space. The exact membership set represents a polytope in the parameter space. The volume of $\psi(k)$ decreases for each k in which the pairs $(\mathbf{x}(k), d(k))$ bring about some innovation. Note that $\Theta \subset \psi(k)$ for all k, since Θ is the intersection of all possible constraint sets.

The target of set-membership adaptive filtering is to obtain adaptively an estimate that belongs to the feasibility set. The simplest method is to calculate a point estimate using, for example, the information provided by $\mathcal{H}(k)$ similar to the set-membership NLMS algorithm described in the following subsection, or several previous $\mathcal{H}(k)$ like in the SM-AP algorithm discussed in Subsection 2.2.3.

2.2.2 Set-membership normalized LMS algorithm

The set-membership NLMS algorithm, first proposed in [9], implements a test to check if the previous estimate $\mathbf{w}(k)$ lies outside the constraint set $\mathcal{H}(k)$. If $|d(k) - \mathbf{w}^T(k)\mathbf{x}(k)| > \overline{\gamma}$, then $\mathbf{w}(k+1)$ will be updated to the closest boundary of $\mathcal{H}(k)$ at a minimum distance. Figure 2.2 depicts the updating procedure of the SM-NLMS algorithm.

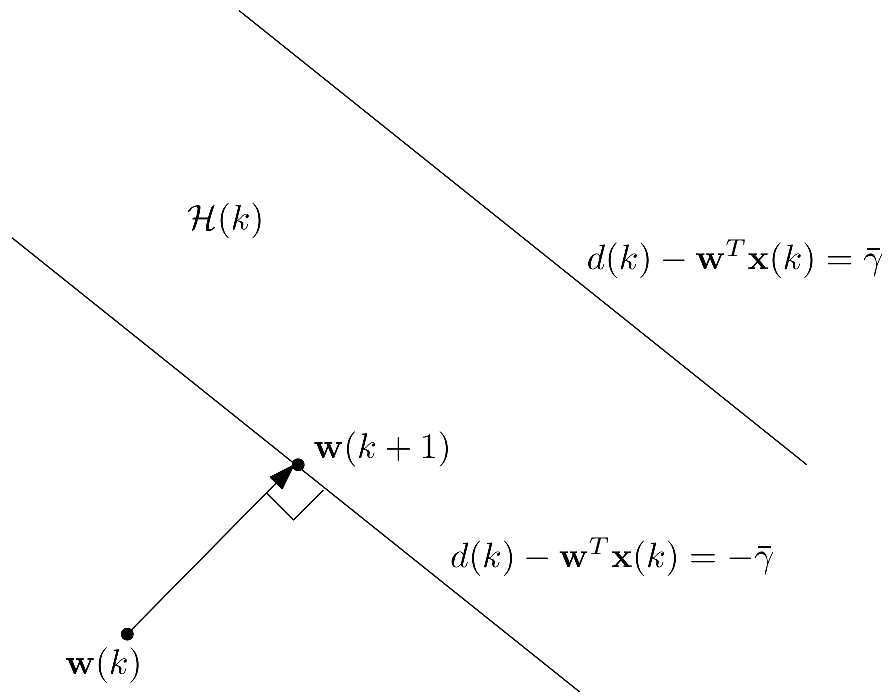

$\mathcal{H}(k)$
$d(k) - \mathbf{w}^T\mathbf{x}(k) = \bar{\gamma}$
$\mathbf{w}(k+1)$
$d(k) - \mathbf{w}^T\mathbf{x}(k) = -\bar{\gamma}$
$\mathbf{w}(k)$

Figure 2.2: Coefficient vector updating for the SM-NLMS algorithm (redrawn from [2]).

The SM-NLMS algorithm has the updating rule

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \frac{\mu(k)}{\mathbf{x}^T(k)\mathbf{x}(k) + \delta} e(k)\mathbf{x}(k), \tag{2.14}$$

where the variable step size $\mu(k)$ is given by

$$\mu(k) = \begin{cases} 1 - \frac{\overline{\gamma}}{|e(k)|} & \text{if } |e(k)| > \overline{\gamma}, \\ 0 & \text{otherwise,} \end{cases} \tag{2.15}$$

and δ is a small regularization factor. As a rule of thumb, the value of $\overline{\gamma}$ is selected about $\sqrt{\tau\sigma_n^2}$, where σ_n^2 is the variance of the additional noise [9, 33], and $1 \leq \tau \leq 5$.

Note that we can introduce the NLMS algorithm through the SM-NLMS algorithm. Indeed, the NLMS algorithm with unit step size is a particular case of the SM-NLMS algorithm by adopting $\overline{\gamma} = 0$.

2.2.3 Set-membership affine projection algorithm

The exact membership set $\psi(k)$ suggests the use of more constraint sets in the update [31]. Moreover, it is widely known that data-reusing algorithms can increase convergence speed significantly for correlated-input signals [2, 11, 34]. This section introduces the SM-AP algorithm whose updates belong to the last $L+1$ constraint sets. For this purpose, let us define the input signal matrix $\mathbf{X}(k)$, the output signal vector $\mathbf{y}(k)$, the error signal vector $\mathbf{e}(k)$, the desired signal vector $\mathbf{d}(k)$, the additive noise signal vector $\mathbf{n}(k)$, and the constraint vector (CV) $\boldsymbol{\gamma}(k)$ as

$$\begin{aligned}
\mathbf{X}(k) &= [\mathbf{x}(k)\ \mathbf{x}(k-1)\ \cdots\ \mathbf{x}(k-L)] \in \mathbb{R}^{(N+1)\times(L+1)},\\
\mathbf{x}(k) &= [x(k)\ x(k-1)\ \cdots\ x(k-N)]^T \in \mathbb{R}^{N+1},\\
\mathbf{y}(k) &= [y(k)\ y(k-1)\ \cdots\ y(k-L)]^T \in \mathbb{R}^{L+1},\\
\mathbf{e}(k) &= [e(k)\ \epsilon(k-1)\ \cdots\ \epsilon(k-L)]^T \in \mathbb{R}^{L+1},\\
\mathbf{d}(k) &= [d(k)\ d(k-1)\ \cdots\ d(k-L)]^T \in \mathbb{R}^{L+1},\\
\mathbf{n}(k) &= [n(k)\ n(k-1)\ \cdots\ n(k-L)]^T \in \mathbb{R}^{L+1},\\
\boldsymbol{\gamma}(k) &= [\gamma_0(k)\ \gamma_1(k)\ \cdots\ \gamma_L(k)]^T \in \mathbb{R}^{L+1},
\end{aligned} \tag{2.16}$$

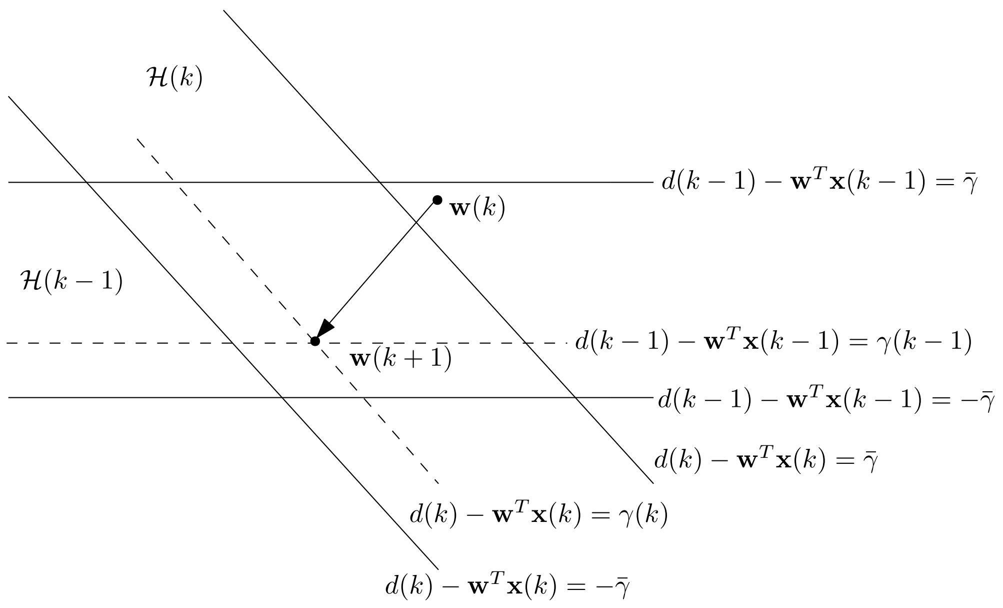
where N is the order of the adaptive filter, and L is the data-reusing factor, i.e., L previous data are used together with the data from the current iteration k. The output signal vector is defined as $\mathbf{y}(k) \triangleq \mathbf{w}^T(k)\mathbf{X}(k) = \mathbf{X}^T(k)\mathbf{w}(k)$, the desired signal vector is given by $\mathbf{d}(k) \triangleq \mathbf{w}_o^T\mathbf{X}(k) + \mathbf{n}(k)$, where $\mathbf{w}_o$ is the optimal solution (unknown system), and the error signal vector is given by $\mathbf{e}(k) \triangleq \mathbf{d}(k) - \mathbf{y}(k)$. The entries of the constraint vector should satisfy $|\gamma_i(k)| \leq \overline{\gamma}$, for $i = 0, \ldots, L$, where $\overline{\gamma} \in \mathbb{R}_+$ is the upper bound for the magnitude of the error signal $e(k)$.

The objective function to be minimized in the SM-AP algorithm can be stated as follows: a coefficient update is implemented whenever $\mathbf{w}(k) \notin \psi^{L+1}(k)$ in such a way that

$$\begin{aligned}
&\min \frac{1}{2}\|\mathbf{w}(k+1) - \mathbf{w}(k)\|^2\\
&\text{subject to:}\\
&\mathbf{d}(k) - \mathbf{X}^T(k)\mathbf{w}(k+1) = \boldsymbol{\gamma}(k),
\end{aligned} \tag{2.17}$$

where $\psi^{L+1}(k)$ is the intersection of the $L+1$ last constraint sets.

Figure 2.3 shows a usual coefficient update related to the SM-AP algorithm in $\mathbb{R}^2$,



$\mathcal{H}(k)$
$d(k-1) - \mathbf{w}^T\mathbf{x}(k-1) = \bar{\gamma}$
$\mathbf{w}(k)$
$\mathcal{H}(k-1)$
$d(k-1) - \mathbf{w}^T\mathbf{x}(k-1) = \gamma(k-1)$
$\mathbf{w}(k+1)$
$d(k-1) - \mathbf{w}^T\mathbf{x}(k-1) = -\bar{\gamma}$
$d(k) - \mathbf{w}^T\mathbf{x}(k) = \bar{\gamma}$
$d(k) - \mathbf{w}^T\mathbf{x}(k) = \gamma(k)$
$d(k) - \mathbf{w}^T\mathbf{x}(k) = -\bar{\gamma}$

Figure 2.3: Coefficient vector updating for the SM-AP algorithm (redrawn from [2]).

$L = 1$ and $|\gamma_i(k)| \leq \overline{\gamma}$ such that $\mathbf{w}(k+1)$ is not placed at the border of $\mathcal{H}(k)$.

By using the method of Lagrange multipliers, after some manipulations, the recursion rule of the SM-AP algorithm will be described as

$$\mathbf{w}(k+1) = \begin{cases} \mathbf{w}(k) + \mathbf{X}(k)\mathbf{A}(k)(\mathbf{e}(k) - \boldsymbol{\gamma}(k)) & \text{if } |e(k)| > \overline{\gamma}, \\ \mathbf{w}(k) & \text{otherwise,} \end{cases} \tag{2.18}$$

where we assume that $\mathbf{A}(k) \triangleq (\mathbf{X}^T(k)\mathbf{X}(k))^{-1} \in \mathbb{R}^{L+1\times L+1}$ exists, i.e., $\mathbf{X}^T(k)\mathbf{X}(k)$ is a full-rank matrix. Otherwise, we could add a regularization parameter as explained in [2].

Note that we can propose the AP algorithm through the SM-AP algorithm. In other words, the AP algorithm with unity step-size, aiming at improving the convergence speed of stochastic gradient algorithms, is a particular case of the SM-AP algorithm by selecting $\overline{\gamma} = 0$.

It is worthwhile to mention that when $L = 0$ and $\gamma_0(k) = \frac{\overline{\gamma}e(k)}{|e(k)|}$, the SM-AP algorithm has the SM-NLMS algorithm as special case.

2.3 Estimating $\overline{\gamma}$ in the Set-Membership Algorithm for Big Data Application

In big data applications, initially, it could be practical to prescribe a percentage of the amount of data we intend to utilize to achieve the desired performance. This percentage will be defined in accordance with our ability to analyze the data, taking into consideration the constraints on energy, computational time, and memory restrictions. After adopting a percentage of the update, our goal is to select the most informative data to be part of the corresponding selected percentage. Here, by taking the probability of updating into consideration, we will estimate the threshold in the SM-NLMS and the SM-AP algorithms, which is responsible for censoring the data in accordance with the adopted percentage of the update. The content of this section is published in [35].

We want to obtain $\overline{\gamma}$ such that the algorithm considers the desired percentage of data to update its recursion rule. In fact, if the magnitude of the output estimation error is greater than $\overline{\gamma}$, the set-membership (SM) algorithm will update since the current input and the desired signals carry enough innovation.

In general, for the desired update rate, p, we require computing $\overline{\gamma}$ such that

$$\mathbb{P}[|e(k)| > \overline{\gamma}] = p, \tag{2.19}$$

where $\mathbb{P}[\cdot]$ denotes the probability operator. Note that p represents the update rate of the algorithm, i.e., the percentage of the data which we consider most informative data.

Given the probability density function of the error signal, then it is possible to compute $\overline{\gamma}$. Note that the error signal is the difference between the desired and the output signals, i.e.,

$$\begin{aligned} e(k) &\triangleq d(k) - y(k) \triangleq \mathbf{w}_o^T\mathbf{x}(k) + n(k) - \mathbf{w}^T(k)\mathbf{x}(k) \\ &= [\mathbf{w}_o - \mathbf{w}(k)]^T\mathbf{x}(k) + n(k) = \widetilde{e}(k) + n(k), \end{aligned} \tag{2.20}$$

where $\widetilde{e}(k)$ is the noiseless error signal, and $n(k)$ is the noise signal. In the steady-state environment $\|\mathbb{E}[\mathbf{w}_o - \mathbf{w}(k)]\|_2^2 < \infty$ [7], where $\mathbb{E}[\cdot]$ is the expected value operator and, in general, $\mathbb{E}[\mathbf{w}_o - \mathbf{w}(k)] \approx \mathbf{0}$. Therefore, if you have sufficient order for the adaptive system, then in the steady-state environment the distribution of the error signal and the additive noise signal are the same. Thus, we can use the distribution of the additive noise signal in Equation (2.19) to calculate the desired value of $\overline{\gamma}$.

Assuming the distribution of the noise signal is Gaussian with zero mean and variance σ_n^2, an important case, we can provide a solution for the threshold for this special case. If the noiseless error signal is uncorrelated with the additional noise signal, by Equation (2.20), we have $\mathbb{E}[e(k)] = \mathbb{E}[\widetilde{e}(k)] + \mathbb{E}[n(k)] = 0$ and $\text{Var}[e(k)] = \mathbb{E}[\widetilde{e}^2(k)] + \sigma_n^2$, where $\text{Var}[\cdot]$ is the variance operator. $\mathbb{E}[\widetilde{e}^2(k)]$ is the excess of the steady-state mean-square error (EMSE) that in the steady-state environment is given by [36, 37]

$$\mathbb{E}[\widetilde{e}^2(k)] = \frac{(L+1)[\sigma_n^2 + \overline{\gamma}^2 - 2\overline{\gamma}\sigma_n^2\rho_0(k)]p}{[(2-p) - 2(1-p)\overline{\gamma}\rho_o(k)]}\Big(\frac{1-a}{1-a^{L+1}}\Big), \tag{2.21}$$

where

$$\rho_0(k) = \sqrt{\frac{2}{\pi(2\sigma_n^2 + \frac{1}{L+1}\overline{\gamma}^2)}}, \tag{2.22}$$

$$a = [1 - p + 2p\overline{\gamma}\rho_0(k)](1-p). \tag{2.23}$$

To calculate $\mathbb{E}[\widetilde{e}^2(k)]$ in Equation (2.21), we require the value of $\overline{\gamma}$, while estimating $\overline{\gamma}$ is our purpose. To address this problem, the natural approach is estimate it using numerical integration or Monte-Carlo methods. However, aiming at gaining some insight, at the first moment we can assume that in the steady-state environment $\mathbb{E}[\widetilde{e}^2(k)] = 0$, and the distribution of $e(k)$ is the same as $n(k)$, in order to calculate the estimation of $\overline{\gamma}$ using Equation (2.19). Then, we substitute the obtained value of $\overline{\gamma}$ in Equation (2.21) to compute $\mathbb{E}[\widetilde{e}^2(k)]$. Finally, by obtaining $\mathbb{E}[\widetilde{e}^2(k)]$, we can have a better estimation for the distribution of $e(k)$.

Therefore, since the distribution of $e(k)$ is the same as the distribution of $n(k)$, for the first estimation of $\overline{\gamma}$ we have

$$\mathbb{P}[|e(k)| > \overline{\gamma}] = \mathbb{P}[|n(k)| > \overline{\gamma}] = \mathbb{P}[n(k) < -\overline{\gamma}] + \mathbb{P}[n(k) > \overline{\gamma}] = p. \tag{2.24}$$

Then because of the symmetry in Gaussian distribution we have $\mathbb{P}[n(k) > \overline{\gamma}] = \frac{p}{2}$. Since $n(k)$ has Gaussian distribution, we need to obtain $\overline{\gamma}$ from

$$\int_{\overline{\gamma}}^{\infty} \frac{1}{\sqrt{2\pi\sigma_n^2}} \exp(-\frac{r^2}{2\sigma_n^2})dr = \frac{p}{2}. \tag{2.25}$$

Hence, given an update rate $0 \leq p \leq 1$, we may use the standard normal distribution table and find the desired $\overline{\gamma}$. As the second step, for getting a better estimation of $\overline{\gamma}$, we substitute $\overline{\gamma}$ in Equations (2.21)-(2.23) to obtain $\mathbb{E}[\widetilde{e}^2(k)]$. We can now use the zero mean Gaussian distribution with variance $\sigma_e^2 = \mathbb{E}[\widetilde{e}^2(k)] + \sigma_n^2$ as the distribution of the error signal. Applying this distribution to Equation (2.19), we can obtain a better estimation for $\overline{\gamma}$ through the equation

$$\int_{\overline{\gamma}}^{\infty} \frac{1}{\sqrt{2\pi\sigma_e^2}} \exp(-\frac{r^2}{2\sigma_e^2})dr = \frac{p}{2}. \tag{2.26}$$

By using the standard normal distribution table, from where we can find the new estimation of $\overline{\gamma}$. It is worth mentioning that the chosen desired update rate determines a loose relative importance of the innovation brought about by the new incoming data set.

2.4 Conclusions

In this chapter, we have reviewed some adaptive filtering algorithms which play an essential role in the following chapters. First, we have introduced the LMS, the NLMS, the AP, and the RLS algorithms. Then, we have described the SMF approach. By incorporating this strategy into the conventional algorithms, we implement an update when the magnitude of the output estimation error is greater than the predetermined positive constant. For this purpose, we have defined some of the involved sets such as the feasibility set, the constraint set, and the exact membership set. Then, we have described the SM-NLMS and the SM-AP algorithms. Finally, for the SM-NLMS and the SM-AP algorithms, we have discussed how to estimate the threshold parameter in big data applications to obtain the desired update rate.

Chapter 3

On the Robustness of the Set-Membership Algorithms

Online learning algorithms are a substantial part of Adaptive Signal Processing, thus the efficiency of the algorithms has to be assessed. The classical adaptive filtering algorithms are iterative estimation methods based on the *point estimation theory* [20]. This theory focuses on searching for a unique solution that minimizes (or maximizes) some objective function. Two widely used classical algorithms are the normalized least-mean-square (NLMS) and the affine projection (AP) algorithms. These algorithms present a trade-off between convergence rate and steady-state misadjustment, and their properties have been extensively studied [2, 10].

Two important set-membership (SM) algorithms are the set-membership NLMS (SM-NLMS) and the set-membership AP (SM-AP) algorithms, proposed in [9, 31], respectively. These algorithms keep the advantages of their classical counterparts, but they are more accurate, more robust against noise, and also reduce the computational complexities due to the data selection strategy previously explained [2, 37–39]. Various applications of SM algorithms and their advantages over the classical algorithms have been discussed in the literature [28, 29, 40–45].

Despite the recognized advantages of the SM algorithms, they are not broadly used, probably due to the limited analysis of the properties of these algorithms. The steady-state mean-squared error (MSE) analysis of the SM-NLMS algorithm has been discussed in [46, 47]. Also, the steady-state MSE performance of the SM-AP algorithm has been analyzed in [36, 37, 48].

The content of this chapter was published in [6, 7]. In this chapter, the robustness of the SM-NLMS and the SM-AP algorithms are discussed in the sense of l_2 stability [10, 49]. For the SM-NLMS algorithm, we demonstrate that it is robust regardless the choice of its parameters and that the SM-NLMS enhances the parameter estimation in most of the iterations in which an update occurs, two advantages over the classical NLMS algorithm. Moreover, we also prove that if the noise bound is known, then we can set the SM-NLMS so that it never degrades the estimate. As for the SM-AP algorithm, we demonstrate that its robustness depends on a judicious choice of one of its parameters: the constraint vector (CV). We prove the existence of CVs satisfying the robustness condition, but practical choices remain unknown. We also demonstrate that both the SM-AP and the SM-NLMS algorithms do not diverge, even when their parameters are selected naively, provided that the additional noise is bounded. Section 3.1 describes the robustness criterion. Section 3.2 presents the algorithms discussed in this chapter. The robustness of the SM-NLMS algorithm is studied in Section 3.3, where we also discuss the cases in which the noise bound is assumed known and unknown. Section 3.4 presents the local and the global robustness properties of the SM-AP algorithm. Section 3.5 contains the simulations and numerical results. Finally, concluding remarks are drawn in Section 3.6.

3.1 Robustness Criterion

At every iteration k, assume that the desired signal $d(k)$ is related to the unknown system $\mathbf{w}_o$ by

$$d(k) \triangleq \underbrace{\mathbf{w}_o^T \mathbf{x}(k)}_{\triangleq y_o(k)} + n(k), \tag{3.1}$$

where $n(k)$ denotes the unknown noise and accounts for both measurement noise and modeling uncertainties or errors. Also, we assume that the unknown noise sequence $\{n(k)\}$ has finite energy [10], i.e.,

$$\sum_{k=0}^{j} |n(k)|^2 < \infty, \qquad \text{for all } j. \tag{3.2}$$

Suppose that we have a sequence of desired signals $\{d(k)\}$ and we intend to estimate $y_o(k) = \mathbf{w}_o^T \mathbf{x}(k)$. For this purpose, assume that $\hat{y}_{k|k}$ is an estimate of $y_o(k)$ and it is only dependent on $d(j)$ for $j = 0, \cdots, k$. For a given positive number η, we aim at calculating the following estimates $\hat{y}_{k|k} \in \{\hat{y}_{0|0}, \hat{y}_{1|1}, \cdots, \hat{y}_{M|M}\}$, such that for any $n(k)$

satisfying (3.2) and any $\mathbf{w}_o$, the following criterion is satisfied:

$$\frac{\sum\limits_{k=0}^{j} \|\hat{y}_{k|k} - y_o(k)\|^2}{\widetilde{\mathbf{w}}^T(0)\widetilde{\mathbf{w}}(0) + \sum_{k=0}^{j} |n(k)|^2} < \eta^2, \qquad \text{for all } j = 0, \cdots, M \tag{3.3}$$

where $\widetilde{\mathbf{w}}(0) \triangleq \mathbf{w}_o - \mathbf{w}(0)$ and $\mathbf{w}(0)$ is our initial guess about $\mathbf{w}_o$. Note that the numerator is a measure of estimation-error energy up to iteration j and the denominator includes the energy of disturbance up to iteration j and the energy of the error $\widetilde{\mathbf{w}}(0)$ that is due to the initial guess.

So, the criterion given in (3.3) requires that we adjust estimates $\{\hat{y}_{k|k}\}$ such that the ratio of the estimation-error energy (numerator) to the energy of the uncertainties (denominator) does not exceed η^2. When this criterion is satisfied, we say that bounded disturbance energies induce bounded estimation-error energies and, therefore, the obtained estimates are robust. The smaller value of η results in the more robust solution, but the value of η cannot be decreased freely. The interested reader can refer to [10], pages 719 and 720, for more details about this robustness criterion.

3.2 The Set-Membership Algorithms

In this section, we remind the SM-NLMS and the SM-AP algorithms, and in the following sections we deal with their robustness.

3.2.1 The SM-NLMS Algorithm

The SM-NLMS algorithm is characterized by the updating rule [2]

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \frac{\mu(k)}{\|\mathbf{x}(k)\|^2 + \delta} e(k)\mathbf{x}(k), \tag{3.4}$$

where

$$\mu(k) \triangleq \begin{cases} 1 - \frac{\overline{\gamma}}{|e(k)|} & \text{if } |e(k)| > \overline{\gamma}, \\ 0 & \text{otherwise,} \end{cases} \tag{3.5}$$

and $\overline{\gamma} \in \mathbb{R}_+$ is the upper bound for the magnitude of the error signal that is acceptable and it is usually chosen as a multiple of the noise standard deviation σ_n [2, 37]. The parameter $\delta \in \mathbb{R}_+$ is a regularization factor, usually chosen as a small constant, used to avoid singularity (divisions by 0).

3.2.2 The SM-AP Algorithm

The SM-AP algorithm is described by the recursion [31]

$$\mathbf{w}(k+1) = \begin{cases} \mathbf{w}(k) + \mathbf{X}(k)\mathbf{A}(k)(\mathbf{e}(k) - \boldsymbol{\gamma}(k)) & \text{if } |e(k)| > \overline{\gamma}, \\ \mathbf{w}(k) & \text{otherwise,} \end{cases} \tag{3.6}$$

where we assume that $\mathbf{A}(k) \triangleq (\mathbf{X}^T(k)\mathbf{X}(k))^{-1} \in \mathbb{R}^{(L+1)\times(L+1)}$ exists, i.e., $\mathbf{X}^T(k)\mathbf{X}(k)$ is a full-rank matrix. Otherwise, we could add a regularization parameter as explained in [2].

3.3 Robustness of the SM-NLMS Algorithm

In this section, we discuss the robustness of the set-membership NLMS (SM-NLMS) algorithm. In Subsection 3.3.1, we present some robustness properties. We address the robustness of the SM-NLMS algorithm for the cases of unknown noise bound and known noise bound in Subsections 3.3.2 and 3.3.3, respectively. Then, in Subsection 3.3.4, we introduce a time-varying error bound aiming at achieving simultaneously fast convergence, low computational burden, and efficient use of the input data.

3.3.1 Robustness of the SM-NLMS algorithm

Let us consider a system identification scenario in which the unknown system is denoted by $\mathbf{w}_o \in \mathbb{R}^{N+1}$ and the desired (reference) signal $d(k)$ is defined as

$$d(k) \triangleq \mathbf{w}_o^T \mathbf{x}(k) + n(k), \tag{3.7}$$

where $n(k) \in \mathbb{R}$ represents the additive measurement noise.

One of the main difficulties of analyzing the SM-NLMS algorithm is its conditional statement in (3.5). We can overcome such difficulty by defining

$$\overline{\mu}(k) \triangleq 1 - \frac{\overline{\gamma}}{|e(k)|}, \tag{3.8}$$

and the indicator function $f : \mathbb{R} \times \mathbb{R}_+ \rightarrow \{0, 1\}$ as

$$f(e(k), \overline{\gamma}) \triangleq \begin{cases} 1 & \text{if } |e(k)| > \overline{\gamma}, \\ 0 & \text{otherwise.} \end{cases} \tag{3.9}$$

In this way, the SM-NLMS updating rule can be rewritten as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \frac{\overline{\mu}(k)}{\alpha(k)} e(k)\mathbf{x}(k) f(e(k), \overline{\gamma}), \tag{3.10}$$

where

$$\alpha(k) \triangleq \|\mathbf{x}(k)\|^2 + \delta. \tag{3.11}$$

Since we are interested in robustness properties, it is useful to define $\widetilde{\mathbf{w}}(k) \in \mathbb{R}^{N+1}$ as

$$\widetilde{\mathbf{w}}(k) \triangleq \mathbf{w}_o - \mathbf{w}(k), \tag{3.12}$$

i.e., $\widetilde{\mathbf{w}}(k)$ is a vector representing the discrepancy between the quantity we aim to estimate $\mathbf{w}_o$ and our current estimate $\mathbf{w}(k)$. Thus, the error signal can be rewritten as

$$\begin{aligned} e(k) = d(k) - \mathbf{w}^T(k)\mathbf{x}(k) &= \mathbf{w}_o^T \mathbf{x}(k) + n(k) - \mathbf{w}^T(k)\mathbf{x}(k) \\ &= \underbrace{\widetilde{\mathbf{w}}^T(k)\mathbf{x}(k)}_{\triangleq \widetilde{e}(k)} + n(k), \end{aligned} \tag{3.13}$$

where $\widetilde{e}(k)$ denotes the noiseless error, i.e., the error due to a mismatch between $\mathbf{w}(k)$ and $\mathbf{w}_o$.

By using (3.12) in (3.10) we obtain

$$\widetilde{\mathbf{w}}(k+1) = \widetilde{\mathbf{w}}(k) - \frac{\overline{\mu}(k)}{\alpha(k)} e(k)\mathbf{x}(k) f(e(k), \overline{\gamma}), \tag{3.14}$$

which can be further expanded by decomposing $e(k)$ as in Equation (3.13) yielding

$$\widetilde{\mathbf{w}}(k+1) = \widetilde{\mathbf{w}}(k) - \frac{\overline{\mu}(k)}{\alpha(k)} \widetilde{e}(k)\mathbf{x}(k) f(e(k), \overline{\gamma}) - \frac{\overline{\mu}(k)}{\alpha(k)} n(k)\mathbf{x}(k) f(e(k), \overline{\gamma}). \tag{3.15}$$

By computing the energy of (3.15), the robustness property given in Theorem 1 can be derived after some mathematical manipulations.

Theorem 1 (Local Robustness of SM-NLMS)**.** *For the SM-NLMS algorithm, it always holds that*

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 = \|\widetilde{\mathbf{w}}(k)\|^2, \text{ \textit{if} } f(e(k),\overline{\gamma}) = 0 \tag{3.16}$$

or

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 + \frac{\overline{\mu}(k)}{\alpha(k)}\widetilde{e}^2(k) < \|\widetilde{\mathbf{w}}(k)\|^2 + \frac{\overline{\mu}(k)}{\alpha(k)}n^2(k)\ , \tag{3.17}$$

if $f(e(k),\overline{\gamma}) = 1$.

Proof. We start by repeating Equation (3.15), but to simplify the notation we will omit both the index k and the arguments of function f that appear on the right-hand side of that equation to obtain

$$\widetilde{\mathbf{w}}(k+1) = \widetilde{\mathbf{w}} - \frac{\overline{\mu}}{\alpha}\widetilde{e}\mathbf{x}f - \frac{\overline{\mu}}{\alpha}n\mathbf{x}f. \tag{3.18}$$

By computing the Euclidean norm of the above equation we get

$$\begin{aligned}
\|\widetilde{\mathbf{w}}(k+1)\|^2 =& \widetilde{\mathbf{w}}^T\widetilde{\mathbf{w}} - \frac{\overline{\mu}}{\alpha}\widetilde{e}\widetilde{\mathbf{w}}^T\mathbf{x}f - \frac{\overline{\mu}}{\alpha}n\widetilde{\mathbf{w}}^T\mathbf{x}f - \frac{\overline{\mu}}{\alpha}\widetilde{e}\mathbf{x}^T\widetilde{\mathbf{w}}f + \frac{\overline{\mu}^2}{\alpha^2}\widetilde{e}^2\mathbf{x}^T\mathbf{x}f^2 \\
&+ \frac{\overline{\mu}^2}{\alpha^2}\widetilde{e}n\mathbf{x}^T\mathbf{x}f^2 - \frac{\overline{\mu}}{\alpha}n\mathbf{x}^T\widetilde{\mathbf{w}}f + \frac{\overline{\mu}^2}{\alpha^2}n\widetilde{e}\mathbf{x}^T\mathbf{x}f^2 + \frac{\overline{\mu}^2}{\alpha^2}n^2\mathbf{x}^T\mathbf{x}f^2 \\
=& \|\widetilde{\mathbf{w}}\|^2 - \frac{\overline{\mu}}{\alpha}\widetilde{e}^2f - \frac{\overline{\mu}}{\alpha}n\widetilde{e}f - \frac{\overline{\mu}}{\alpha}\widetilde{e}^2f + \frac{\overline{\mu}^2}{\alpha^2}\widetilde{e}^2\|\mathbf{x}\|^2f^2 + \frac{\overline{\mu}^2}{\alpha^2}\widetilde{e}n\|\mathbf{x}\|^2f^2 \\
&- \frac{\overline{\mu}}{\alpha}n\widetilde{e}f + \frac{\overline{\mu}^2}{\alpha^2}n\widetilde{e}\|\mathbf{x}\|^2f^2 + \frac{\overline{\mu}^2}{\alpha^2}n^2\|\mathbf{x}\|^2f^2 \\
=& \|\widetilde{\mathbf{w}}\|^2 - 2\frac{\overline{\mu}}{\alpha}\widetilde{e}^2f - 2\frac{\overline{\mu}}{\alpha}n\widetilde{e}f + (\widetilde{e}+n)^2\frac{\overline{\mu}^2}{\alpha^2}\|\mathbf{x}\|^2f^2 \\
=& \|\widetilde{\mathbf{w}}\|^2 + (\widetilde{e}+n)^2\frac{\overline{\mu}^2}{\alpha^2}\|\mathbf{x}\|^2f^2 - 2\frac{\overline{\mu}}{\alpha}\widetilde{e}^2f - 2\frac{\overline{\mu}}{\alpha}n\widetilde{e}f - \frac{\overline{\mu}}{\alpha}n^2f + \frac{\overline{\mu}}{\alpha}n^2f \\
=& \|\widetilde{\mathbf{w}}\|^2 + (\widetilde{e}+n)^2\frac{\overline{\mu}^2}{\alpha^2}\|\mathbf{x}\|^2f^2 + \frac{\overline{\mu}}{\alpha}n^2f - (\widetilde{e}+n)^2\frac{\overline{\mu}}{\alpha}f - \frac{\overline{\mu}}{\alpha}\widetilde{e}^2f,
\end{aligned} \tag{3.19}$$

where the second equality is due to the relation $\widetilde{e} = \widetilde{\mathbf{w}}^T\mathbf{x} = \mathbf{x}^T\widetilde{\mathbf{w}}$, as given in Equation (3.13). Rearranging the terms in (3.19) yields

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 + \frac{\overline{\mu}f}{\alpha}\widetilde{e}^2 = \|\widetilde{\mathbf{w}}\|^2 + \frac{\overline{\mu}f}{\alpha}n^2 + c_1c_2, \tag{3.20}$$

where

$$c_1 \triangleq \frac{\overline{\mu} f}{\alpha}(\widetilde{e}+n)^2, \qquad c_2 \triangleq \frac{\overline{\mu} f}{\alpha}\|\mathbf{x}\|^2 - 1. \tag{3.21}$$

From (3.20), we observe that when $f = 0$ we have

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 = \|\widetilde{\mathbf{w}}(k)\|^2 \tag{3.22}$$

as expected, since $f = 0$ means that no update was performed. However, when $f = 1$ we have $0 < \overline{\mu} < 1$ and $(\widetilde{e} + n)^2 = e^2 > \overline{\gamma}^2 > 0$. In addition, observe that $0 \leq \|\mathbf{x}\|^2/\alpha < 1$ due to Equation (3.11) and the fact that $\delta > 0$. Combining these inequalities leads to $c_2 < 0$ and $c_1 > 0$. Thus, when $f = 1$ the product $c_1 c_2 < 0$, which leads to the inequality

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 + \frac{\overline{\mu}}{\alpha}\widetilde{e}^2 < \|\widetilde{\mathbf{w}}\|^2 + \frac{\overline{\mu}}{\alpha}n^2. \tag{3.23}$$

Returning with the omitted index k, for $f(e(k), \overline{\gamma}) = 1$ we have

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 + \frac{\overline{\mu}(k)}{\alpha(k)}\widetilde{e}^2(k) < \|\widetilde{\mathbf{w}}(k)\|^2 + \frac{\overline{\mu}(k)}{\alpha(k)}n^2(k). \tag{3.24}$$

$\square$

Theorem 1 presents local bounds for the energy of the coefficient deviation when running from an iteration to the next one. Indeed, (3.16) states that the coefficient deviation does not change when no coefficient update is actually implemented, whereas (3.17) provides a bound for $\|\widetilde{\mathbf{w}}(k+1)\|^2$ based on $\|\widetilde{\mathbf{w}}(k)\|^2$, $\widetilde{e}^2(k)$, and $n^2(k)$, when an update occurs. In addition, the global robustness result in Corollary 1 can readily be derived from Theorem 1.

Corollary 1 (Global Robustness of SM-NLMS)**.** *Consider the SM-NLMS algorithm running from iteration* 0 *(initialization) to a given iteration K. The relation*

$$\frac{\|\widetilde{\mathbf{w}}(K)\|^2 + \sum_{k\in\mathcal{K}_{\rm up}} \frac{\overline{\mu}(k)}{\alpha(k)}\widetilde{e}^2(k)}{\|\widetilde{\mathbf{w}}(0)\|^2 + \sum_{k\in\mathcal{K}_{\rm up}} \frac{\overline{\mu}(k)}{\alpha(k)}n^2(k)} < 1 \tag{3.25}$$

holds, where $\mathcal{K}_{\rm up} \neq \emptyset$ is the set containing the iteration indexes k in which $\mathbf{w}(k)$ is indeed updated. If $\mathcal{K}_{\rm up} = \emptyset$, then $\|\widetilde{\mathbf{w}}(K)\|^2 = \|\widetilde{\mathbf{w}}(0)\|^2$ due to (3.16)*, but this case is* *not of practical interest since $\mathcal{K}_{\text{up}} = \emptyset$ means that no update is performed at all.*

Proof. Define the set of all iterations under analysis $\mathcal{K} \triangleq \{0, 1, 2, \ldots, K-1\}$. Denote as $\mathcal{K}_{\text{up}}$ the subset of $\mathcal{K}$ comprised only of the iterations in which an update occurs, whereas $\mathcal{K}_{\text{up}}^c \triangleq \mathcal{K} \setminus \mathcal{K}_{\text{up}}$ contains the iteration indexes in which the filter coefficients are not updated.

From Theorem 1, (3.17) holds when $\mathbf{w}(k)$ is updated. By summing such inequality for all $k \in \mathcal{K}_{\text{up}}$ we obtain

$$\sum_{k\in\mathcal{K}_{\text{up}}} \left(\|\widetilde{\mathbf{w}}(k+1)\|^2 + \frac{\overline{\mu}(k)}{\alpha(k)}\widetilde{e}^2(k)\right) < \sum_{k\in\mathcal{K}_{\text{up}}} \left(\|\widetilde{\mathbf{w}}(k)\|^2 + \frac{\overline{\mu}(k)}{\alpha(k)}n^2(k)\right). \tag{3.26}$$

Similarly, we can use (3.16) to write, for all $k \in \mathcal{K}_{\text{up}}^c$, the equality

$$\sum_{k\in\mathcal{K}_{\text{up}}^c} \|\mathbf{w}(k+1)\|^2 = \sum_{k\in\mathcal{K}_{\text{up}}^c} \|\mathbf{w}(k)\|^2. \tag{3.27}$$

Combining (3.26) and (3.27) leads to

$$\sum_{k\in\mathcal{K}} \|\widetilde{\mathbf{w}}(k+1)\|^2 + \sum_{k\in\mathcal{K}_{\text{up}}} \frac{\overline{\mu}(k)}{\alpha(k)}\widetilde{e}^2(k) < \sum_{k\in\mathcal{K}} \|\widetilde{\mathbf{w}}(k)\|^2 + \sum_{k\in\mathcal{K}_{\text{up}}} \frac{\overline{\mu}(k)}{\alpha(k)}n^2(k). \tag{3.28}$$

But since several of the terms $\|\widetilde{\mathbf{w}}(k)\|^2$ get canceled from both sides of the inequality (3.28), we find that it simplifies to

$$\|\widetilde{\mathbf{w}}(K)\|^2 + \sum_{k\in\mathcal{K}_{\text{up}}} \frac{\overline{\mu}(k)}{\alpha(k)}\widetilde{e}^2(k) < \|\widetilde{\mathbf{w}}(0)\|^2 + \sum_{k\in\mathcal{K}_{\text{up}}} \frac{\overline{\mu}(k)}{\alpha(k)}n^2(k) \tag{3.29}$$

or, assuming a nonzero denominator,

$$\frac{\|\widetilde{\mathbf{w}}(K)\|^2 + \sum_{k\in\mathcal{K}_{\text{up}}} \frac{\overline{\mu}(k)}{\alpha(k)}\widetilde{e}^2(k)}{\|\widetilde{\mathbf{w}}(0)\|^2 + \sum_{k\in\mathcal{K}_{\text{up}}} \frac{\overline{\mu}(k)}{\alpha(k)}n^2(k)} < 1. \tag{3.30}$$

This relation holds for all K. The only assumption used in the derivation is that $\mathcal{K}_{\text{up}}$ is a nonempty set. Otherwise, we would have $\|\widetilde{\mathbf{w}}(K)\|^2 = \|\widetilde{\mathbf{w}}(0)\|^2$, which would happen only if $\mathbf{w}(k)$ is never updated, which has no practical interest. □

Corollary 1 shows that, for the SM-NLMS algorithm, l_2-stability from its uncertainties $\{\widetilde{\mathbf{w}}(0), \{n(k)\}_{0\leq k\leq K}\}$ to its errors $\{\widetilde{\mathbf{w}}(K), \{\widetilde{e}(k)\}_{0\leq k\leq K}\}$ is always guaranteed.

Unlike the NLMS algorithm, in which the step-size parameter must be chosen appropriately to guarantee such l_2-stability, for the SM-NLMS algorithm it is taken for granted (i.e., no restriction is imposed on $\overline{\gamma}$).

3.3.2 Convergence of $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ with unknown noise bound

The robustness results mentioned in Subsection 3.3.1 provide bounds for the evolution of $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ in terms of other variables. However, we have experimentally observed that the SM-NLMS algorithm presents a well-behaved convergence of the sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$, i.e., for most iterations we have $\|\widetilde{\mathbf{w}}(k+1)\|^2 \leq \|\widetilde{\mathbf{w}}(k)\|^2$. Therefore, in this subsection, we investigate under which conditions the sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ is (and is not) decreasing.

Corollary 2. *When an update occurs (i.e., $f(e(k),\overline{\gamma}) = 1$), $\widetilde{e}^2(k) \geq n^2(k)$ implies $\|\widetilde{\mathbf{w}}(k+1)\|^2 < \|\widetilde{\mathbf{w}}(k)\|^2$.*

Proof. By rearranging the terms in (3.17) we obtain

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 + \frac{\overline{\mu}(k)}{\alpha(k)}\left(\widetilde{e}^2(k) - n^2(k)\right) < \|\widetilde{\mathbf{w}}(k)\|^2, \tag{3.31}$$

which is valid for $f(e(k),\overline{\gamma}) = 1$. Observe that $\frac{\overline{\mu}(k)}{\alpha(k)} > 0$ since $\alpha(k) \in \mathbb{R}_+$ and $\overline{\mu}(k) \in (0,1)$ when $f(e(k),\overline{\gamma}) = 1$. Thus $\frac{\overline{\mu}(k)}{\alpha(k)}\left(\widetilde{e}^2(k) - n^2(k)\right) \geq 0$ when $f(e(k),\overline{\gamma}) = 1$ and $\widetilde{e}^2(k) \geq n^2(k)$. Therefore, when an update occurs, $\widetilde{e}^2(k) \geq n^2(k) \Rightarrow \|\widetilde{\mathbf{w}}(k+1)\|^2 < \|\widetilde{\mathbf{w}}(k)\|^2$. □

In words, Corollary 2 states that the SM-NLMS algorithm improves its estimate $\mathbf{w}(k+1)$ every time an update is required and the energy of the error signal $e^2(k)$ is dominated by $\widetilde{e}^2(k)$, the component of the error which is due to the mismatch between $\mathbf{w}(k)$ and $\mathbf{w}_o$.

Corollary 2 also explains why the SM-NLMS algorithm usually presents a *monotonic decreasing sequence* $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ during its transient period. Indeed, in the early iterations, the absolute value of the error is generally large, thus $|e(k)| > \overline{\gamma}$ and $\widetilde{e}^2(k) > n^2(k)$, implying that $\|\widetilde{\mathbf{w}}(k+1)\|^2 < \|\widetilde{\mathbf{w}}(k)\|^2$. In addition, there are a few iterations during the transient period in which the input data do not bring enough innovation so that no update is performed, which means that $\|\widetilde{\mathbf{w}}(k+1)\|^2 = \|\widetilde{\mathbf{w}}(k)\|^2$ for these few iterations. As a conclusion, it is very likely to have $\|\widetilde{\mathbf{w}}(k+1)\|^2 \leq \|\widetilde{\mathbf{w}}(k)\|^2$ for all iterations k belonging to the transient period.

After the transient period, however, the SM-NLMS algorithm may yield $\|\widetilde{\mathbf{w}}(k+1)\|^2 > \|\widetilde{\mathbf{w}}(k)\|^2$ in a few iterations. Although it is hard to compute how often such an event occurs, we can provide an upper bound for the probability of this event as follows:

$$\begin{aligned}\mathbb{P}[\|\widetilde{\mathbf{w}}(k+1)\|^2 > \|\widetilde{\mathbf{w}}(k)\|^2] &\leq \mathbb{P}[\{|e(k)| > \overline{\gamma}\} \cap \{\widetilde{e}^2(k) < n^2(k)\}] \\ &< \mathbb{P}[|e(k)| > \overline{\gamma}] = \operatorname{erfc}\left(\sqrt{\frac{\tau}{2}}\right), \end{aligned} \tag{3.32}$$

where $\mathbb{P}[\cdot]$ and $\operatorname{erfc}(\cdot)$ are the probability operator and the complementary error function [50], respectively. The first inequality follows from the fact that we do not know exactly what will happen with $\|\widetilde{\mathbf{w}}(k+1)\|^2$ when an update occurs and $\widetilde{e}^2(k) < n^2(k)$ at the same time[1] and, therefore, it corresponds to a *pessimistic bound*. The second inequality is trivial and the subsequent equality follows from [33] by parameterizing $\overline{\gamma}$ as $\overline{\gamma} = \sqrt{\tau\sigma_n^2}$, where $\tau \in \mathbb{R}_+$ (typically $\tau = 5$) and by modeling the error $e(k)$ as a zero-mean Gaussian random variable with variance σ_n^2.

From (3.32), one can observe that the probability of obtaining $\|\widetilde{\mathbf{w}}(k+1)\|^2 > \|\widetilde{\mathbf{w}}(k)\|^2$ is small. For instance, for $2 \leq \tau \leq 9$ we have $0.0027 \leq \operatorname{erfc}\left(\sqrt{\frac{\tau}{2}}\right) \leq 0.1579$, and for the usual choice $\tau = 5$ we have $\operatorname{erfc}\left(\sqrt{\frac{\tau}{2}}\right) = 0.0253$.

The results in this subsection show that $\|\widetilde{\mathbf{w}}(k+1)\|^2 \leq \|\widetilde{\mathbf{w}}(k)\|^2$ for most iterations of the SM-NLMS algorithm, meaning that the SM-NLMS algorithm uses the input data efficiently. Indeed, having $\|\widetilde{\mathbf{w}}(k+1)\|^2 > \|\widetilde{\mathbf{w}}(k)\|^2$ means that the input data was used to obtain an estimate $\mathbf{w}(k+1)$ which is further away from the quantity we aim to estimate $\mathbf{w}_o$, which is a waste of computational resources (it would be better not to update at all). Here, we showed that this rarely happens for the SM-NLMS algorithm, a property not shared by the classical algorithms, as it will be verified experimentally in Section 3.5.

3.3.3 Convergence of $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ with known noise bound

In this subsection, we demonstrate that if the noise bound is known, then it is possible to set the threshold parameter $\overline{\gamma}$ of the SM-NLMS algorithm so that $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ is a monotonic decreasing sequence. Theorem 2 and Corollary 3 address this issue.

[1] This is because Corollary 2 provides a sufficient, but not necessary, condition for $\|\widetilde{\mathbf{w}}(k+1)\|^2 < \|\widetilde{\mathbf{w}}(k)\|^2$.

Theorem 2 (Strong Local Robustness of SM-NLMS)**.** *Assume the noise is bounded by a known constant $B \in \mathbb{R}_+$, i.e., $|n(k)| \leq B, \forall k$. If one chooses $\overline{\gamma} \geq 2B$, then $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ is a monotonic decreasing sequence, i.e., $\|\widetilde{\mathbf{w}}(k+1)\|^2 \leq \|\widetilde{\mathbf{w}}(k)\|^2, \forall k$.*

Proof. If $f(e(k), \overline{\gamma}) = 1$, then $|e(k)| = |\widetilde{e}(k) + n(k)| > \overline{\gamma}$, which means that: (i) $\widetilde{e}(k) > \overline{\gamma} - n(k)$ for the positive values of $\widetilde{e}(k)$ or (ii) $\widetilde{e}(k) < -\overline{\gamma} - n(k)$ for the negative values of $\widetilde{e}(k)$. Recalling that $n(k) \in [-B, B]$ and $\overline{\gamma} \in [2B, \infty)$, now we can find the bound for $\widetilde{e}(k)$ by finding the minimum of (i) and the maximum of (ii) as follows:
(i) $\widetilde{e}(k) > \overline{\gamma} - n(k) \Rightarrow \widetilde{e}_{\min} > \overline{\gamma} - B \geq B$;
(ii) $\widetilde{e}(k) < -\overline{\gamma} - n(k) \Rightarrow \widetilde{e}_{\max} < -\overline{\gamma} + B \leq -B$.
Results (i) and (ii) above state that if $\overline{\gamma} \geq 2B$, then $|\widetilde{e}(k)| > B$, which means that $|\widetilde{e}(k)| > |n(k)|, \forall k$. Consequently, by using Corollary 2 it follows that $\|\widetilde{\mathbf{w}}(k+1)\|^2 < \|\widetilde{\mathbf{w}}(k)\|^2, \forall k$ in which $f(e(k), \overline{\gamma}) = 1$. In addition, if $f(e(k), \overline{\gamma}) = 0$ we have $\|\widetilde{\mathbf{w}}(k+1)\|^2 = \|\widetilde{\mathbf{w}}(k)\|^2$. Therefore, we can conclude that $\overline{\gamma} \geq 2B \Rightarrow \|\widetilde{\mathbf{w}}(k+1)\|^2 \leq \|\widetilde{\mathbf{w}}(k)\|^2, \forall k$. □

Corollary 3 (Strong Global Robustness of SM-NLMS)**.** *Consider the SM-NLMS algorithm running from iteration 0 (initialization) to a given iteration K. If $\overline{\gamma} \geq 2B$, then $\|\widetilde{\mathbf{w}}(K)\|^2 \leq \|\widetilde{\mathbf{w}}(0)\|^2$, in which the equality holds only when no update is performed along all the iterations.*

The proof of Corollary 3 is omitted because it is a straightforward consequence of Theorem 2.

3.3.4 Time-varying $\overline{\gamma}(k)$

After reading Subsections 3.3.2 and 3.3.3, one might be tempted to set $\overline{\gamma}$ as a high value since it reduces the number of updates, thus saving computational resources and also leading to a well-behaved sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ that has high probability of being monotonously decreasing. However, a high value of $\overline{\gamma}$ leads to slow convergence, because the updates during the learning stage (transient period) are less frequent and the step-size $\mu(k)$ is reduced as well. Hence, $\overline{\gamma}$ represents a compromise between convergence speed and efficiency and, therefore, should be chosen carefully according to the specific characteristics of the application.

An alternative approach is to allow a time-varying error bound $\overline{\gamma}(k)$ generally defined as $\overline{\gamma}(k) \triangleq \sqrt{\tau(k)\sigma_n^2}$, where

$$\tau(k) \triangleq \begin{cases} \text{Low value (e.g., } \tau(k) \in [1,5]), & \text{if } k \in \text{transient period,} \\ \text{High value (e.g., } \tau(k) \in [5,9]), & \text{if } k \in \text{steady-state.} \end{cases} \tag{3.33}$$

By using such a $\overline{\gamma}(k)$, one obtains the best features of the high and low values of $\overline{\gamma}$ discussed in the first paragraph of this subsection. In addition, if the noise bound B is known, then one should set $\overline{\gamma}(k) \geq 2B$ for all k during the steady-state, as explained in Subsection 3.3.3. It is worth mentioning that (3.33) provides a general expression for $\tau(k)$ that allows it to vary smoothly along the iterations even within a single period (i.e., transient period or steady-state).

In order to apply the $\overline{\gamma}(k)$ defined above, the algorithm should be able to monitor the environment to determine when there is a transition between transient and steady-state periods. An intuitive way to do this is to monitor the values of $|e(k)|$. In this case, one should form a window with the $E \in \mathbb{N}$ most recent values of the error, compute the average of these $|e(k)|$ within the window, and compare it against a threshold parameter to make the decision. An even more intuitive and efficient way to monitor the iterations relies on how often the algorithm is updating. In this case, one should form a window of length E containing Boolean variables (flags, i.e., 1-bit information) indicating the iterations in which an update was performed considering the E most recent iterations. If many updates were performed within the window, then the algorithm must be in the transient period; otherwise, the algorithm is likely to be in steady-state.

3.4 Robustness of the SM-AP Algorithm

In this section, we address the robustness of the set-membership affine projection (SM-AP) algorithm. We study its robustness properties in Subsection 3.4.1, whereas in Subsection 3.4.2, we demonstrate that the SM-AP algorithm does not diverge.

3.4.1 Robustness of the SM-AP algorithm

Suppose that in a system identification problem the unknown system is denoted by $\mathbf{w}_o \in \mathbb{R}^{N+1}$ and the desired (reference) vector is given by

$$\mathbf{d}(k) \triangleq \mathbf{X}^T(k)\mathbf{w}_o + \mathbf{n}(k). \tag{3.34}$$

By defining the coefficient mismatch $\widetilde{\mathbf{w}}(k) \triangleq \mathbf{w}_o - \mathbf{w}(k)$, the error vector can be written as

$$\mathbf{e}(k) = \mathbf{X}^T(k)\mathbf{w}_o + \mathbf{n}(k) - \mathbf{X}^T(k)\mathbf{w}(k) = \underbrace{\mathbf{X}^T(k)\widetilde{\mathbf{w}}(k)}_{\triangleq \widetilde{\mathbf{e}}(k)} + \mathbf{n}(k)\ , \tag{3.35}$$

where $\widetilde{\mathbf{e}}(k)$ denotes the noiseless error vector (i.e., the error due to a nonzero $\widetilde{\mathbf{w}}(k)$). By defining the indicator function $f : \mathbb{R} \times \mathbb{R}_+ \to \{0, 1\}$ as in (3.9) and using it in (3.6), the update rule of the SM-AP algorithm can be written as follows:

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{X}(k)\mathbf{A}(k)(\mathbf{e}(k) - \boldsymbol{\gamma}(k))f(e(k), \overline{\gamma}), \tag{3.36}$$

where $\mathbf{A}(k) = [\mathbf{X}^T(k)\mathbf{X}(k)]^{-1}$. After subtracting $\mathbf{w}_o$ from both sides of (3.36), we obtain

$$\widetilde{\mathbf{w}}(k+1) = \widetilde{\mathbf{w}}(k) - \mathbf{X}(k)\mathbf{A}(k)(\mathbf{e}(k) - \boldsymbol{\gamma}(k))f(e(k), \overline{\gamma}). \tag{3.37}$$

Notice that $\mathbf{A}(k)$ is a symmetric positive definite matrix. To simplify our notation, we will omit the index k and the arguments of function f that appear on the right-hand side (RHS) of the previous equation, then by decomposing $\mathbf{e}(k)$ as in (3.35) we obtain

$$\widetilde{\mathbf{w}}(k+1) = \widetilde{\mathbf{w}} - \mathbf{X}\mathbf{A}\widetilde{\mathbf{e}}f - \mathbf{X}\mathbf{A}\mathbf{n}f + \mathbf{X}\mathbf{A}\boldsymbol{\gamma}f, \tag{3.38}$$

from which Theorem 3 can be derived.

Theorem 3 (Local Robustness of SM-AP)**.** *For the SM-AP algorithm, at every iteration we have*

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 = \|\widetilde{\mathbf{w}}(k)\|^2, \ \textit{if } f(e(k), \overline{\gamma}) = 0 \tag{3.39}$$

otherwise

$$\begin{cases} \frac{\|\widetilde{\mathbf{w}}(k+1)\|^2+\widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}}{\|\widetilde{\mathbf{w}}(k)\|^2+\mathbf{n}^T\mathbf{A}\mathbf{n}} < 1, & \textit{if } \boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} < 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}, \\ \frac{\|\widetilde{\mathbf{w}}(k+1)\|^2+\widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}}{\|\widetilde{\mathbf{w}}(k)\|^2+\mathbf{n}^T\mathbf{A}\mathbf{n}} = 1, & \textit{if } \boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} = 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}, \\ \frac{\|\widetilde{\mathbf{w}}(k+1)\|^2+\widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}}{\|\widetilde{\mathbf{w}}(k)\|^2+\mathbf{n}^T\mathbf{A}\mathbf{n}} > 1, & \textit{if } \boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} > 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}, \end{cases} \tag{3.40}$$

where the iteration index k has been dropped for the sake of clarity, and we assume that $\|\widetilde{\mathbf{w}}(k)\|^2 + \mathbf{n}^T\mathbf{A}\mathbf{n} \neq 0$ just to allow us to write the theorem in a compact form.

Proof. By computing the Euclidean norm of Equation (3.38) and rearranging the terms we get

$$\begin{aligned} \|\widetilde{\mathbf{w}}(k+1)\|^2 =& \widetilde{\mathbf{w}}^T\widetilde{\mathbf{w}} - \widetilde{\mathbf{w}}^T\mathbf{X}\mathbf{A}\widetilde{\mathbf{e}}f - \widetilde{\mathbf{w}}^T\mathbf{X}\mathbf{A}\mathbf{n}f + \widetilde{\mathbf{w}}^T\mathbf{X}\mathbf{A}\boldsymbol{\gamma}f - \widetilde{\mathbf{e}}^T\mathbf{A}^T\mathbf{X}^T\widetilde{\mathbf{w}}f \\ & + \widetilde{\mathbf{e}}^T\mathbf{A}^T\mathbf{A}^{-1}\mathbf{A}\widetilde{\mathbf{e}}f^2 + \widetilde{\mathbf{e}}^T\mathbf{A}^T\mathbf{A}^{-1}\mathbf{A}\mathbf{n}f^2 - \widetilde{\mathbf{e}}^T\mathbf{A}^T\mathbf{A}^{-1}\mathbf{A}\boldsymbol{\gamma}f^2 \\ & - \mathbf{n}^T\mathbf{A}^T\mathbf{X}^T\widetilde{\mathbf{w}}f + \mathbf{n}^T\mathbf{A}^T\mathbf{A}^{-1}\mathbf{A}\widetilde{\mathbf{e}}f^2 + \mathbf{n}^T\mathbf{A}^T\mathbf{A}^{-1}\mathbf{A}\mathbf{n}f^2 \\ & - \mathbf{n}^T\mathbf{A}^T\mathbf{A}^{-1}\mathbf{A}\boldsymbol{\gamma}f^2 + \boldsymbol{\gamma}^T\mathbf{A}^T\mathbf{X}^T\widetilde{\mathbf{w}}f - \boldsymbol{\gamma}^T\mathbf{A}^T\mathbf{A}^{-1}\mathbf{A}\widetilde{\mathbf{e}}f^2 \\ & - \boldsymbol{\gamma}^T\mathbf{A}^T\mathbf{A}^{-1}\mathbf{A}\mathbf{n}f^2 + \boldsymbol{\gamma}^T\mathbf{A}^T\mathbf{A}^{-1}\mathbf{A}\boldsymbol{\gamma}f^2 \\ =& \|\widetilde{\mathbf{w}}\|^2 - \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}f - \widetilde{\mathbf{e}}^T\mathbf{A}\mathbf{n}f + \widetilde{\mathbf{e}}^T\mathbf{A}\boldsymbol{\gamma}f - \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}f + \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}f^2 \\ & + \widetilde{\mathbf{e}}^T\mathbf{A}\mathbf{n}f^2 - \widetilde{\mathbf{e}}^T\mathbf{A}\boldsymbol{\gamma}f^2 - \mathbf{n}^T\mathbf{A}\widetilde{\mathbf{e}}f + \mathbf{n}^T\mathbf{A}\widetilde{\mathbf{e}}f^2 + \mathbf{n}^T\mathbf{A}\mathbf{n}f^2 \\ & - \mathbf{n}^T\mathbf{A}\boldsymbol{\gamma}f^2 + \boldsymbol{\gamma}^T\mathbf{A}\widetilde{\mathbf{e}}f - \boldsymbol{\gamma}^T\mathbf{A}\widetilde{\mathbf{e}}f^2 - \boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}f^2 + \boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma}f^2 \ , \end{aligned} \tag{3.41}$$

where it was used that $\mathbf{A}^{-1} = \mathbf{X}^T(k)\mathbf{X}(k)$ and $\widetilde{\mathbf{e}}(k) = \mathbf{X}^T(k)\widetilde{\mathbf{w}}(k)$. From the above equation we observe that when $f = 0$ we have

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 = \|\widetilde{\mathbf{w}}(k)\|^2 \tag{3.42}$$

as expected, since $f = 0$ means that the algorithm does not update its coefficients. However, when $f = 1$ the following equality is achieved from (3.41):

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 = \|\widetilde{\mathbf{w}}\|^2 - \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}} + \mathbf{n}^T\mathbf{A}\mathbf{n} - 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n} + \boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} \ . \tag{3.43}$$

After rearranging the terms of the previous equation we obtain

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 + \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}} = \|\widetilde{\mathbf{w}}\|^2 + \mathbf{n}^T\mathbf{A}\mathbf{n} - 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n} + \boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} \ . \tag{3.44}$$

Therefore, $\|\widetilde{\mathbf{w}}(k+1)\|^2 + \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}} < \|\widetilde{\mathbf{w}}\|^2 + \mathbf{n}^T\mathbf{A}\mathbf{n}$ if $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} < 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$, $\|\widetilde{\mathbf{w}}(k+1)\|^2 +$

$\widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}} = \|\widetilde{\mathbf{w}}\|^2 + \mathbf{n}^T\mathbf{A}\mathbf{n}$ if $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} = 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$, and $\|\widetilde{\mathbf{w}}(k+1)\|^2 + \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}} > \|\widetilde{\mathbf{w}}\|^2 + \mathbf{n}^T\mathbf{A}\mathbf{n}$ if $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} > 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$.

Assuming $\|\widetilde{\mathbf{w}}\|^2 + \mathbf{n}^T\mathbf{A}\mathbf{n} \neq 0$ we can summarize the discussion above in a compact form as follows:

$$\begin{cases} \frac{\|\widetilde{\mathbf{w}}(k+1)\|^2 + \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}}{\|\widetilde{\mathbf{w}}(k)\|^2 + \mathbf{n}^T\mathbf{A}\mathbf{n}} < 1, & \text{if } \boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} < 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}, \\ \frac{\|\widetilde{\mathbf{w}}(k+1)\|^2 + \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}}{\|\widetilde{\mathbf{w}}(k)\|^2 + \mathbf{n}^T\mathbf{A}\mathbf{n}} = 1, & \text{if } \boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} = 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}, \\ \frac{\|\widetilde{\mathbf{w}}(k+1)\|^2 + \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}}{\|\widetilde{\mathbf{w}}(k)\|^2 + \mathbf{n}^T\mathbf{A}\mathbf{n}} > 1, & \text{if } \boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} > 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}. \end{cases} \tag{3.45}$$

□

The combination of the first two inequalities in (3.40), which corresponds to the case $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} \leq 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$, has an interesting interpretation. It describes that for any constraint vector $\boldsymbol{\gamma}$ satisfying this condition we have

$$\|\widetilde{\mathbf{w}}(k+1)\|^2 + \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}} \leq \|\widetilde{\mathbf{w}}(k)\|^2 + \mathbf{n}^T\mathbf{A}\mathbf{n}, \tag{3.46}$$

no matter what the noise vector $\mathbf{n}(k)$ is. In this way, we can derive the global robustness property of the SM-AP algorithm.

Corollary 4 (Global Robustness of SM-AP)**.** *Suppose that the SM-AP algorithm running from 0 (initialization) to a given iteration K employs a constraint vector $\boldsymbol{\gamma}$ satisfying $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} \leq 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$ at every iteration in which an update occurs. Then, it always holds that*

$$\frac{\|\widetilde{\mathbf{w}}(K)\|^2 + \sum\limits_{k\in\mathcal{K}_{\rm up}} \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}}{\|\widetilde{\mathbf{w}}(0)\|^2 + \sum\limits_{k\in\mathcal{K}_{\rm up}} \mathbf{n}^T\mathbf{A}\mathbf{n}} \leq 1, \tag{3.47}$$

where $\mathcal{K}_{\rm up} \neq \emptyset$ is the set comprised of the iteration indexes k in which $\mathbf{w}(k)$ is indeed updated and the equality holds when $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} = 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$ for every $k \in \mathcal{K}_{\rm up}$. If $\mathcal{K}_{\rm up} = \emptyset$, then $\|\widetilde{\mathbf{w}}(K)\|^2 = \|\widetilde{\mathbf{w}}(0)\|^2$, a case that has no practical interest since no update is performed.

Proof. Denote by $\mathcal{K} \triangleq \{0, 1, 2, \ldots, K-1\}$ the set of all iterations. Let $\mathcal{K}_{\rm up} \subseteq \mathcal{K}$ be the subset containing only the iterations in which an update occurs, whereas $\mathcal{K}_{\rm up}^c \triangleq \mathcal{K}\backslash\mathcal{K}_{\rm up}$ is comprised of the iterations in which the filter coefficients are not updated.

As a consequence of Theorem 3, when an update occurs the inequality given in (3.46) is valid provided $\boldsymbol{\gamma}$ is chosen such that $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} \leq 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$ is respected. In this way, by summing such inequality for all $k \in \mathcal{K}_{\rm up}$ we obtain

$$\sum_{k\in\mathcal{K}_{\rm up}} \Big(\|\widetilde{\mathbf{w}}(k+1)\|^2 + \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}\Big) \leq \sum_{k\in\mathcal{K}_{\rm up}} \Big(\|\widetilde{\mathbf{w}}(k)\|^2 + \mathbf{n}^T\mathbf{A}\mathbf{n}\Big). \tag{3.48}$$

Observe that $\boldsymbol{\gamma}$, $\widetilde{\mathbf{e}}$, $\mathbf{n}$, and $\mathbf{A}$ all depend on the independent variable k, which we have omitted for the sake of simplification. In addition, for the iterations without coefficient update, we have (3.39), which can be summed for all $k \in \mathcal{K}_{\rm up}^c$ leading to

$$\sum_{k\in\mathcal{K}_{\rm up}^c} \|\widetilde{\mathbf{w}}(k+1)\|^2 = \sum_{k\in\mathcal{K}_{\rm up}^c} \|\widetilde{\mathbf{w}}(k)\|^2. \tag{3.49}$$

Summing (3.48) and (3.49) yields

$$\sum_{k\in\mathcal{K}} \|\widetilde{\mathbf{w}}(k+1)\|^2 + \sum_{k\in\mathcal{K}_{\rm up}} \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}} \leq \sum_{k\in\mathcal{K}} \|\widetilde{\mathbf{w}}(k)\|^2 + \sum_{k\in\mathcal{K}_{\rm up}} \mathbf{n}^T\mathbf{A}\mathbf{n}. \tag{3.50}$$

Then, we can cancel several of the terms $\|\widetilde{\mathbf{w}}(k)\|^2$ from both sides of the above inequality simplifying it as follows

$$\|\widetilde{\mathbf{w}}(K)\|^2 + \sum_{k\in\mathcal{K}_{\rm up}} \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}} \leq \|\widetilde{\mathbf{w}}(0)\|^2 + \sum_{k\in\mathcal{K}_{\rm up}} \mathbf{n}^T\mathbf{A}\mathbf{n}. \tag{3.51}$$

Assuming a nonzero denominator, we can write the previous inequality in a compact form

$$\frac{\|\widetilde{\mathbf{w}}(K)\|^2 + \sum\limits_{k\in\mathcal{K}_{\rm up}} \widetilde{\mathbf{e}}^T\mathbf{A}\widetilde{\mathbf{e}}}{\|\widetilde{\mathbf{w}}(0)\|^2 + \sum\limits_{k\in\mathcal{K}_{\rm up}} \mathbf{n}^T\mathbf{A}\mathbf{n}} \leq 1. \tag{3.52}$$

This relation holds for all K, provided $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} \leq 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$ is satisfied for every iteration in which an update occurs, i.e., for every $k \in \mathcal{K}_{\rm up}$. The only assumption used in the derivation is that $\mathcal{K}_{\rm up} \neq \emptyset$. Otherwise, we would have $\|\widetilde{\mathbf{w}}(K)\|^2 = \|\widetilde{\mathbf{w}}(0)\|^2$, which would occur only if $\mathbf{w}(k)$ is never updated, which is not of practical interest. $\square$

Observe that, unlike the SM-NLMS algorithm, the SM-AP algorithm requires the condition $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} \leq 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$ to be satisfied in order to guarantee l_2-stability from its uncertainties $\{\widetilde{\mathbf{w}}(0), \{n(k)\}_{0\leq k\leq K}\}$ to its errors $\{\widetilde{\mathbf{w}}(K), \{\widetilde{e}(k)\}_{0\leq k\leq K}\}$. The next question is: are there constraint vectors $\boldsymbol{\gamma}$ satisfying such a condition? This is a very interesting point because the left-hand side (LHS) of the condition is always positive, whereas the RHS is not. Corollary 5 answers this question and shows an example of such a constraint vector.

Corollary 5. *Suppose the CV is chosen as* $\boldsymbol{\gamma}(k) = c\mathbf{n}(k)$ *in the SM-AP algorithm, where* $\mathbf{n}(k)$ *is the noise vector defined in* (3.34). *If* $0 \leq c \leq 2$, *then the condition* $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} \leq 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$ *always holds, implying that the SM-AP algorithm is globally robust by Corollary 4.*

Proof. Substituting $\boldsymbol{\gamma}(k) = c\mathbf{n}(k)$ in $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} \leq 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$ leads to the following condition $(c^2 - 2c)\mathbf{n}^T(k)\mathbf{A}(k)\mathbf{n}(k) \leq 0$, which is satisfied for $c^2 - 2c \leq 0 \Rightarrow 0 \leq c \leq 2$ since $\mathbf{A}(k)$ is positive definite. Hence, due to Corollary 4 the proposed $\boldsymbol{\gamma}(k)$ leads to a globally robust SM-AP algorithm. $\square$

It is worth mentioning that the constraint vector $\boldsymbol{\gamma}(k)$ in Corollary 5 is not practical because $\mathbf{n}(k)$ is not observable. Therefore, Corollary 5 is actually related to the existence of $\boldsymbol{\gamma}(k)$ satisfying $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} < 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$.

Unlike the SM-NLMS algorithm, the l_2-stability of the SM-AP algorithm is not guaranteed. Indeed, as demonstrated in Theorem 3 and Corollary 4, a judicious choice of the CV is required for the SM-AP algorithm to be l_2-stable. *It is worth mentioning that practical choices of* $\boldsymbol{\gamma}(k)$ *satisfying the robustness condition* $\boldsymbol{\gamma}^T\mathbf{A}\boldsymbol{\gamma} \leq 2\boldsymbol{\gamma}^T\mathbf{A}\mathbf{n}$ *for every iteration* k *are not known yet!* Even widely used CVs, like the simple choice CV (SC-CV) [51], sometimes violate this condition as will be shown in Section 3.5. However, this does not mean that the SM-AP algorithm diverges. In fact, it does not diverge regardless the choice of $\boldsymbol{\gamma}(k)$, as demonstrated in the next subsection.

3.4.2 The SM-AP algorithm does not diverge

When the SM-AP algorithm updates (i.e., when $|e(k)| > \overline{\gamma}$), it generates $\mathbf{w}(k+1)$ as the solution to the following optimization problem [2, 31]:

$$\begin{aligned} &\text{minimize } \|\mathbf{w}(k+1) - \mathbf{w}(k)\|^2 \\ &\text{subject to } \mathbf{d}(k) - \mathbf{X}^T(k)\mathbf{w}(k+1) = \boldsymbol{\gamma}(k). \end{aligned} \tag{3.53}$$

The constraint essentially states that the a posteriori errors $\epsilon(k-l) \triangleq d(k-l) - \mathbf{x}^T(k-l)\mathbf{w}(k+1)$ are equal to their respective $\gamma_l(k)$, which in turn are bounded by $\overline{\gamma}$. This leads to the following derivation:

$$\begin{aligned}|\epsilon(k-l)| = |d(k-l) - \mathbf{x}^T(k-l)\mathbf{w}(k+1)| &\leq \overline{\gamma},\\ |\mathbf{x}^T(k-l)\widetilde{\mathbf{w}}(k+1) + n(k-l)| &\leq \overline{\gamma},\end{aligned} \tag{3.54}$$

which should be valid for all iterations and suitable values of the involved variables. Therefore, we have

$$-\overline{\gamma} - n(k-l) \leq \mathbf{x}^T(k-l)\widetilde{\mathbf{w}}(k+1) \leq \overline{\gamma} - n(k-l). \tag{3.55}$$

Since the noise sequence is bounded and $\overline{\gamma} < \infty$, we have

$$-\infty < \sum_{i=0}^{N} x_i(k-l)\tilde{w}_i(k+1) < \infty, \tag{3.56}$$

where $x_i(k-l), \tilde{w}_i(k+1) \in \mathbb{R}$ denote the ith entry of vectors $\mathbf{x}(k-l), \widetilde{\mathbf{w}}(k+1) \in \mathbb{R}^{N+1}$, respectively. As a result, $|\tilde{w}_i(k+1)|$ is also bounded implying $\|\widetilde{\mathbf{w}}(k+1)\|^2 < \infty$, which means that the SM-AP algorithm does not diverge even when its CV is not properly chosen. In Section 3.5 we verify this fact experimentally by using a *general CV*, i.e., a CV whose entries are randomly chosen but satisfying $|\gamma_i(k)| \leq \overline{\gamma}$. Such general CV leads to poor performance, in comparison to the SM-AP algorithm using adequate CVs, but the algorithm does not diverge.

The same reasoning could be applied to demonstrate that the SM-NLMS algorithm does not diverge as well. However, from Corollary 1 it is straightforward to verify that $\|\widetilde{\mathbf{w}}(K)\|^2 < \infty$ for every K, as the denominator in (3.25) is finite.

3.5 Simulations

In this section, we provide simulation results for the SM-NLMS and SM-AP algorithms in order to verify their robustness properties addressed in the previous sections. These results are obtained by applying the aforementioned algorithms to a system identification problem. The unknown system $\mathbf{w}_o$ is comprised of 10 coefficients drawn from a standard Gaussian distribution. The noise $n(k)$ is a zero-mean white Gaussian noise with variance $\sigma_n^2 = 0.01$ yielding a signal-to-noise ratio (SNR) equal to 20 dB. The regularization factor and the initialization for the adaptive filtering coefficient vector are $\delta = 10^{-12}$ and $\mathbf{w}(0) = [0 \ \cdots \ 0]^T \in \mathbb{R}^{10}$, respectively. The error bound parameter

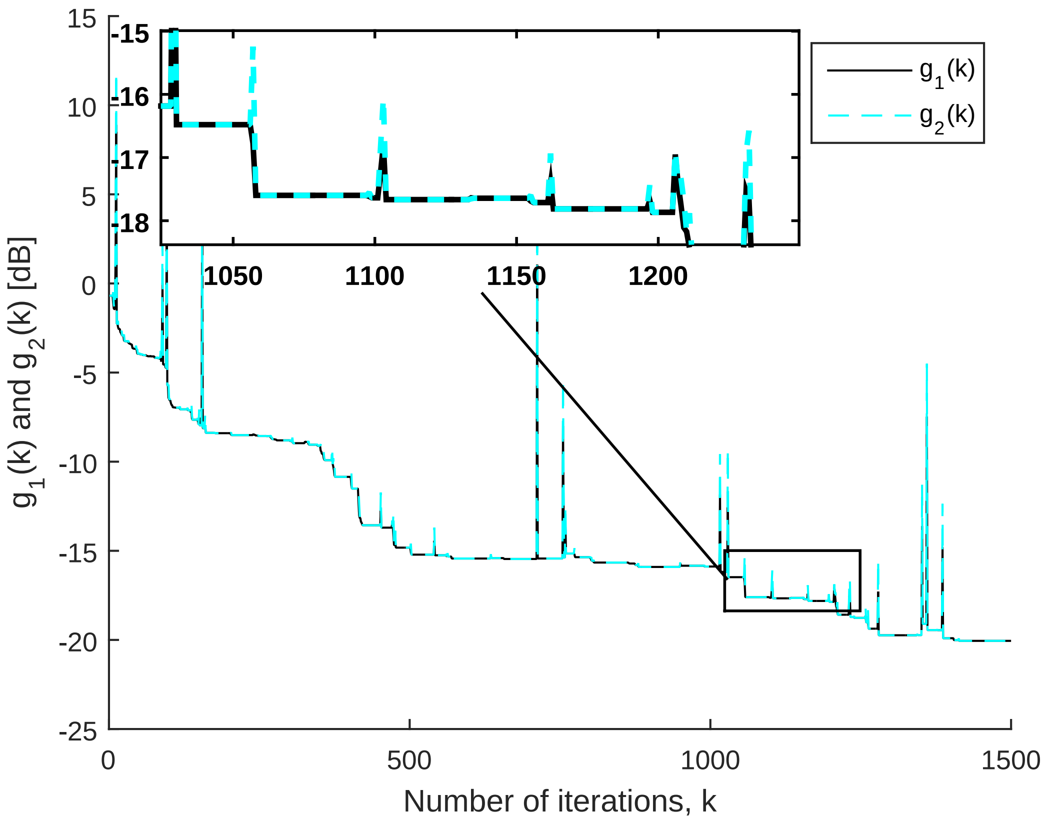


Figure 3.1: Values of $g_1(k)$ and $g_2(k)$ over the iterations for the SM-NLMS algorithm corroborating Theorem 1.

is usually set as $\overline{\gamma} = \sqrt{5\sigma_n^2} = 0.2236$, unless otherwise stated.

3.5.1 Confirming the results for the SM-NLMS algorithm

Here, the input signal $x(k)$ is a zero-mean white Gaussian noise with variance equal to 1. Fig. 3.1 aims at verifying Theorem 1. Thus, for the iterations k with coefficient update, let us denote the left-hand side (LHS) and the right-hand side (RHS) of (3.17) as $g_1(k)$ and $g_2(k)$, respectively. In addition, to simultaneously account for (3.16), we define $g_1(k) = \|\widetilde{\mathbf{w}}(k+1)\|^2$ and $g_2(k) = \|\widetilde{\mathbf{w}}(k)\|^2$ for the iterations without coefficient update. Fig. 3.1 depicts $g_1(k)$ and $g_2(k)$ considering the system identification scenario described in the beginning of Section 3.5. In this figure, we can observe that $g_1(k) \leq g_2(k)$ for all k. Indeed, we verified that $g_1(k) = g_2(k)$ (i.e., curves are overlaid) only in the iterations without update, i.e., $\mathbf{w}(k+1) = \mathbf{w}(k)$. In the remaining iterations we have $g_1(k) < g_2(k)$, corroborating Theorem 1.

Fig. 3.2 depicts the sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ for the SM-NLMS algorithm and its classical counterpart, the NLMS algorithm. For the SM-NLMS algorithm, we consider three cases: fixed $\overline{\gamma}$ with unknown noise bound (blue solid line), fixed $\overline{\gamma}$ with known noise bound $B = 0.11$ (cyan solid line), and time-varying $\overline{\gamma}(k)$, defined as $\sqrt{5\sigma_n^2}$ during the transient period and $\sqrt{9\sigma_n^2}$ during the steady-state, with unknown noise bound (green solid line). For the results using the time-varying $\overline{\gamma}(k)$, the window length is $E = 20$, and when the number of updates in the window is less than 4, we assume the algorithm is in the steady-state period. For the NLMS algorithm, two different step-sizes are used: $\mu = 0.9$, which leads to fast convergence but high misadjustment, and $\mu = 0.05$, which leads to slow convergence but low misadjustment.

In Fig. 3.2, the blue curve confirms the discussion in Subsection 3.3.2. Indeed, we can observe that the sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ represented by this blue curve increases only 30 times along the 2500 iterations, meaning that the SM-NLMS algorithm did not improve its estimate $\mathbf{w}(k+1)$ only in 30 iterations. Thus, in this experiment we have $\mathbb{P}[\|\widetilde{\mathbf{w}}(k+1)\|^2 > \|\widetilde{\mathbf{w}}(k)\|^2] = 0.012$, whose value is lower than its corresponding upper bound given by $\text{erfc}(\sqrt{2.5}) = 0.0253$, as explained in Subsection 3.3.2. Also, we can observe that the event $\|\widetilde{\mathbf{w}}(k+1)\|^2 > \|\widetilde{\mathbf{w}}(k)\|^2$ did not occur in the early iterations because in these iterations $\widetilde{e}^2(k)$ is usually large due to a significant mismatch between $\mathbf{w}(k)$ and $\mathbf{w}_o$, i.e., the condition specified in Corollary 2 is frequently satisfied.

Also in Fig. 3.2, the cyan curve shows that when the noise bound is known we can obtain a monotonic decreasing sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ by selecting $\overline{\gamma} \geq 2B$, corroborating Theorem 2 and Corollary 3. The sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ represented by the green curve in Fig. 3.2 increases only 3 times, thus confirming the advantage of using a time-varying $\overline{\gamma}(k)$ when the noise bound is unknown, as explained in Subsection 3.3.4. As compared to the SM-NLMS algorithm, the behavior of the sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ for the NLMS algorithm is very irregular. Indeed, for the NLMS algorithm there are many iterations in which $\|\widetilde{\mathbf{w}}(k+1)\|^2 > \|\widetilde{\mathbf{w}}(k)\|^2$, even when using a small step-size μ. Hence, the NLMS algorithm does not use the input data as efficiently as the SM-NLMS algorithm does, given that the NLMS performs many "useless updates".

In conclusion, an interesting advantage of the SM-NLMS algorithm over the NLMS algorithm is that the former can achieve fast convergence and has a well-behaved sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ (which rarely increases) at the same time. In addition, the SM-NLMS algorithm also saves computational resources by not updating the filter coefficients at every iteration. In Fig. 3.2, the update rates of the blue, cyan, and green curves are 4.6%, 1.5%, and 1.9%, respectively. They confirm that the computational

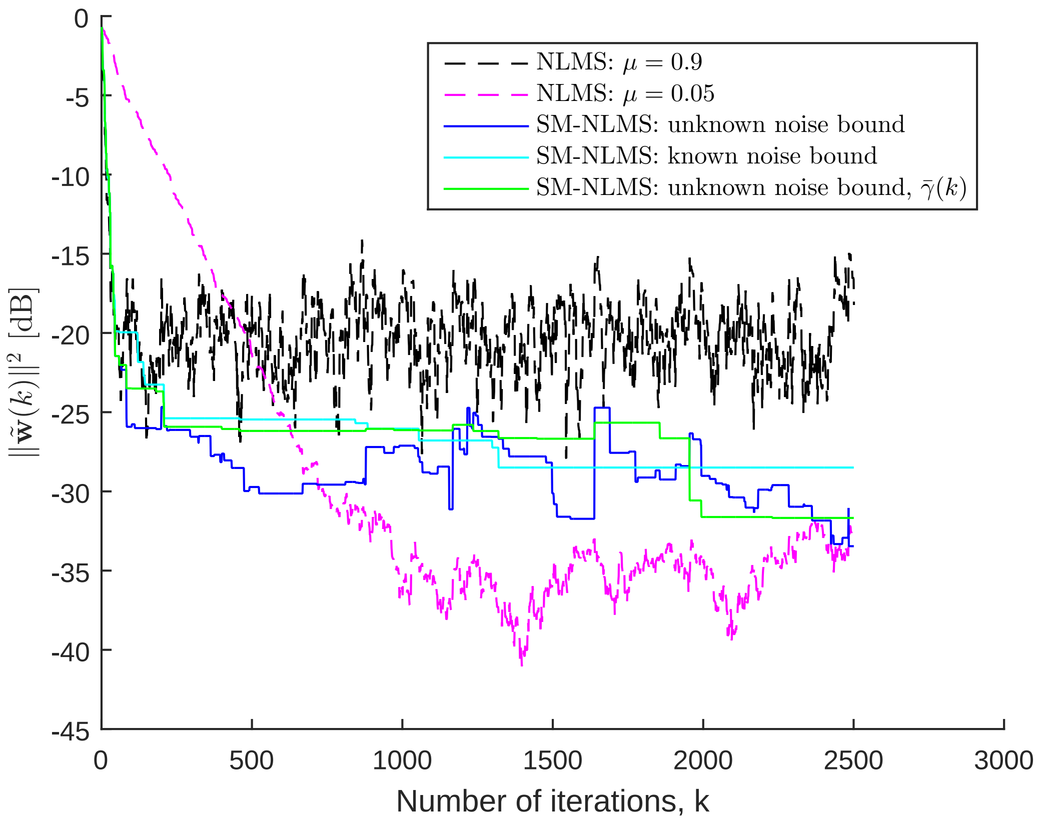


Figure 3.2: $\|\widetilde{\mathbf{w}}(k)\|^2 \triangleq \|\mathbf{w}_o - \mathbf{w}(k)\|^2$ for the NLMS and the SM-NLMS algorithms.

cost of the SM-NLMS algorithm is significantly lower than that of the NLMS algorithm.[2]

3.5.2 Confirming the results for the SM-AP algorithm

For the case of the SM-AP algorithm, the input is a first-order autoregressive signal generated as $x(k) = 0.95x(k-1) + n(k-1)$. We test the SM-AP algorithm employing $L = 2$ (i.e., reuse of two previous input data) and three different constraint vectors (CVs) $\boldsymbol{\gamma}(k)$: a general CV, the SC-CV, and the noise vector CV. The general CV $\boldsymbol{\gamma}(k)$, in which the entries are set as $\gamma_l(k) = \overline{\gamma}$ for $0 \leq l \leq L$, illustrates a case where the CV is not properly chosen [24, 51]. The SC-CV [24, 51] is defined as $\gamma_0(k) = \overline{\gamma}\frac{e(k)}{|e(k)|}$ and $\gamma_l(k) = \epsilon(k-l)$ for $1 \leq l \leq L$. The noise vector CV is given by $\boldsymbol{\gamma}(k) = \mathbf{n}(k)$.

[2]In comparison to the NLMS algorithm, whenever the SM-NLMS algorithm updates it performs two additional operations: One division and one subtraction due to the computation of $\mu(k)$. However, for most of the iterations the SM-NLMS algorithm requires fewer operations because it does not update often.

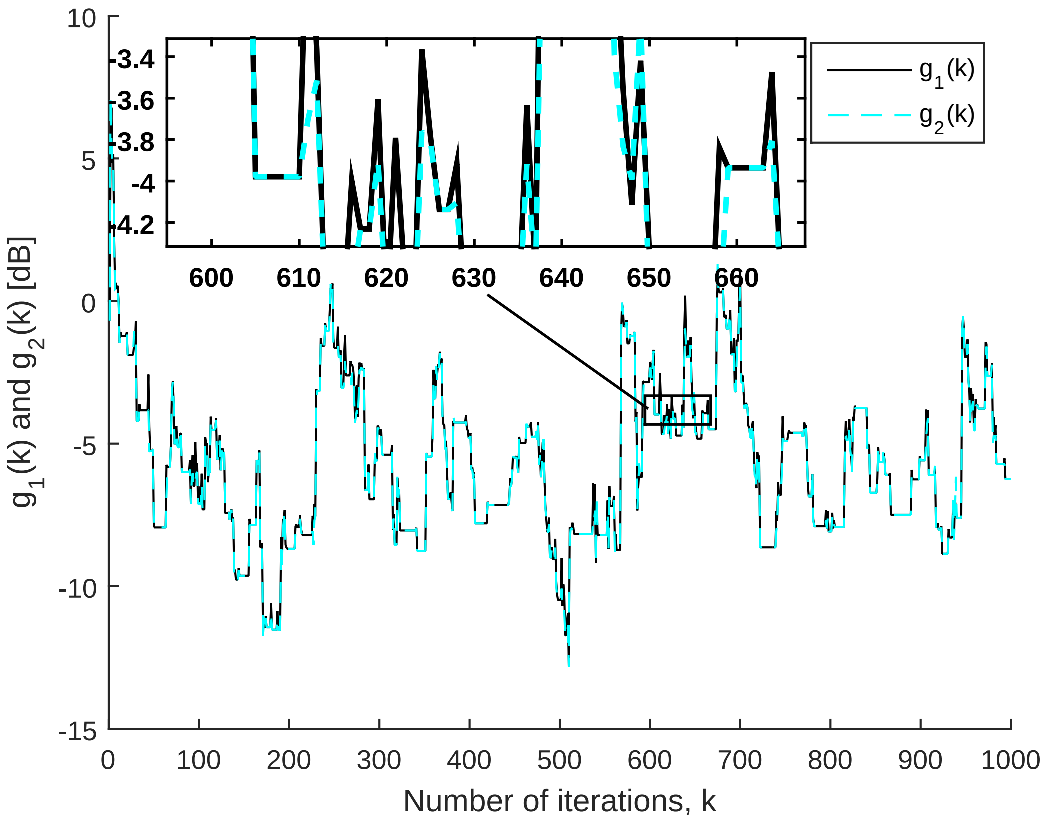


Figure 3.3: Values of $g_1(k)$ and $g_2(k)$ over the iterations for the SM-AP algorithm with $\boldsymbol{\gamma}(k)$ as the general CV, where $g_1(k)$ and $g_2(k)$ are the numerator and denominator of (3.40) in Theorem 3, when an update occurs; otherwise, $g_1(k) = \|\widetilde{\mathbf{w}}(k+1)\|^2$ and $g_2(k) = \|\widetilde{\mathbf{w}}(k)\|^2$.

The results depicted in Figs. 3.3, 3.4, 3.5, and 3.6 aim at verifying Theorem 3 and Corollary 5. We define $g_1(k)$ and $g_2(k)$ as the numerator and the denominator of (3.40) in Theorem 3, respectively, when an update occurs; otherwise, we define $g_1(k) = \|\widetilde{\mathbf{w}}(k+1)\|^2$ and $g_2(k) = \|\widetilde{\mathbf{w}}(k)\|^2$.

The results depicted in Fig. 3.3 illustrate that, for the general CV, there are many iterations in which $g_1(k) > g_2(k)$ (about 293 out of 1000 iterations). This is an expected behavior since the general CV does not take into account (directly or indirectly) the value of $n(k)$ and, therefore, it does not consider the robustness condition $\boldsymbol{\gamma}^T(k)\mathbf{A}(k)\boldsymbol{\gamma}(k) \leq 2\boldsymbol{\gamma}^T(k)\mathbf{A}(k)\mathbf{n}(k)$.

For the SM-AP algorithm employing the SC-CV, however, there are very few iterations in which $g_1(k) > g_2(k)$ (only 19 out of 1000 iterations), as shown in Fig. 3.4. This means that even the widely used SC-CV does not lead to global robustness.

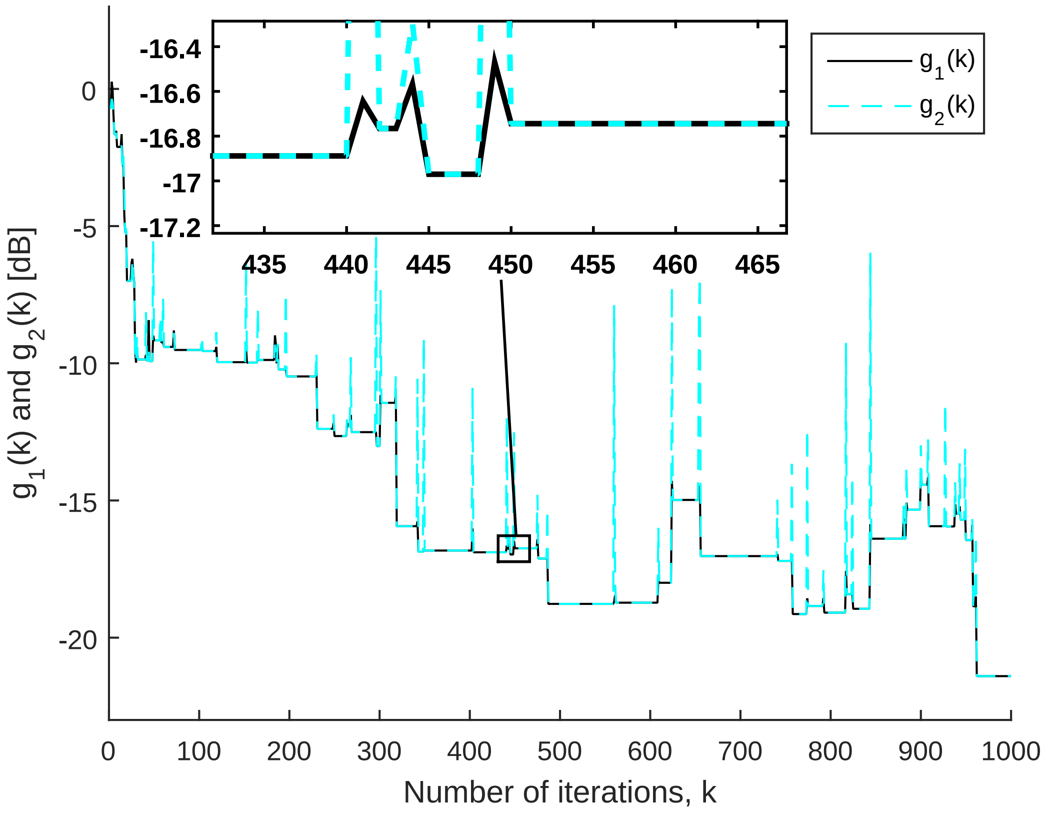


Figure 3.4: Values of $g_1(k)$ and $g_2(k)$ over the iterations for the SM-AP algorithm with $\boldsymbol{\gamma}(k)$ as the SC-CV, where $g_1(k)$ and $g_2(k)$ are the numerator and denominator of (3.40) in Theorem 3, when an update occurs; otherwise, $g_1(k) = \|\widetilde{\mathbf{w}}(k+1)\|^2$ and $g_2(k) = \|\widetilde{\mathbf{w}}(k)\|^2$.

Fig. 3.5 depicts the results for the SM-AP algorithm with $\boldsymbol{\gamma}(k) = \mathbf{n}(k)$. In this case, we can observe that $g_1(k) \leq g_2(k)$ for all k, corroborating Corollary 5. In other words, this CV guarantees the global robustness of the SM-AP algorithm.

Fig. 3.6 illustrates $g_1(k)$ and $g_2(k)$ for the SM-AP algorithm with SC-CV when the noise bound is known and 10 times smaller than $\overline{\gamma}$. In contrast with the SM-NLMS algorithm, for the SM-AP algorithm even when the noise bound is known and much smaller than $\overline{\gamma}$, we cannot guarantee that $g_1(k) \leq g_2(k)$ for all k. In Fig. 3.6, for example, we observe $g_1(k) > g_2(k)$ in 15 iterations.

Fig. 3.7 depicts the sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ for the AP and the SM-AP algorithms. For the AP algorithm, the step-size μ is set as 0.9 and 0.05, whereas for the SM-AP algorithm the three previously defined CVs are tested. For the AP algorithm, we can observe an irregular behavior of $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$, i.e., this sequence increases and decreases

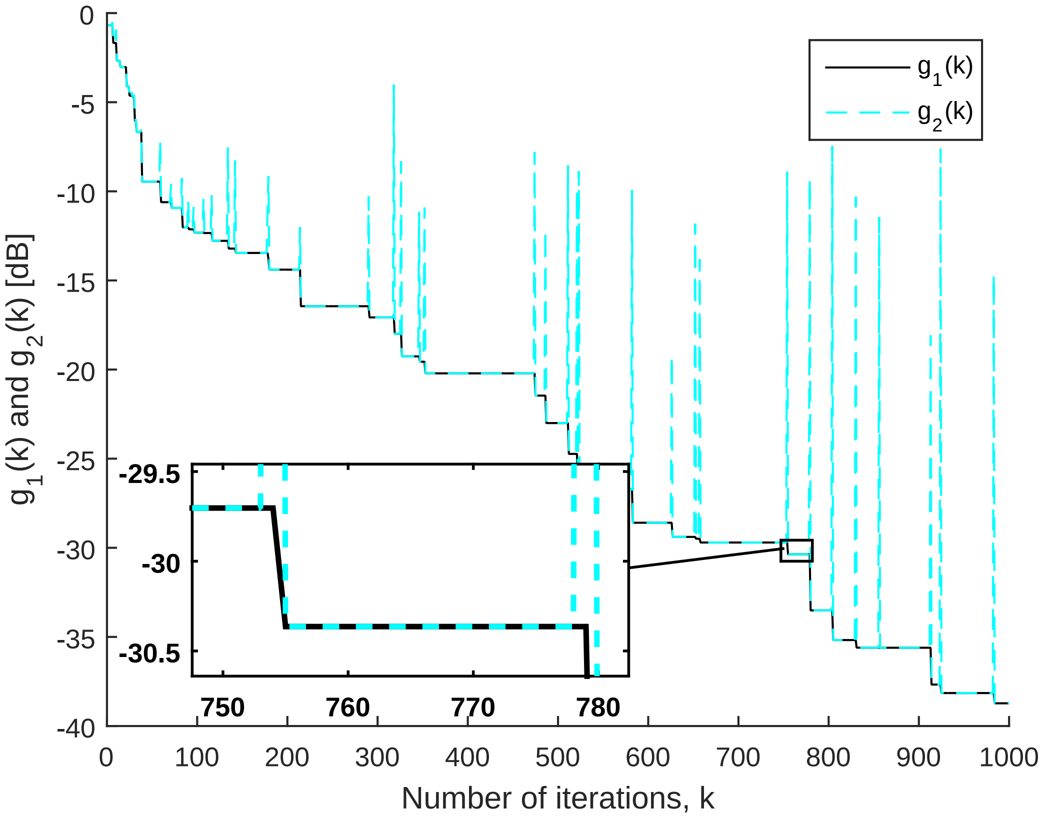


Figure 3.5: Values of $g_1(k)$ and $g_2(k)$ over the iterations for the SM-AP algorithm with $\boldsymbol{\gamma}(k) = \mathbf{n}(k)$, where $g_1(k)$ and $g_2(k)$ are the numerator and denominator of (3.40) in Theorem 3, when an update occurs; otherwise, $g_1(k) = \|\widetilde{\mathbf{w}}(k+1)\|^2$ and $g_2(k) = \|\widetilde{\mathbf{w}}(k)\|^2$.

very often. Even when a low value of μ is applied we still observe many iterations in which $\|\widetilde{\mathbf{w}}(k+1)\|^2 > \|\widetilde{\mathbf{w}}(k)\|^2$ (425 out of 1000 iterations). The SM-AP algorithm using the general CV performs similar to the AP algorithm with high μ. But when the CV is properly chosen, like the SC-CV for example, we observe that the number of iterations in which $\|\widetilde{\mathbf{w}}(k+1)\|^2 > \|\widetilde{\mathbf{w}}(k)\|^2$ is dramatically reduced (26 out of 1000 iterations), which means that the SM-AP with an adequate CV performs fewer "useless updates" than the AP algorithm. Another interesting, although not practical, choice of CV is $\boldsymbol{\gamma}(k) = \mathbf{n}(k)$, which leads to a monotonic decreasing sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$.

The MSE learning curves for the AP and the SM-AP algorithms are depicted in Fig. 3.8. These results were computed by averaging the squared error over 1000 trials for each curve. Observing the results of the AP algorithm, the trade-off between convergence rate and steady-state MSE is evident. Indeed, excluding the SM-AP with

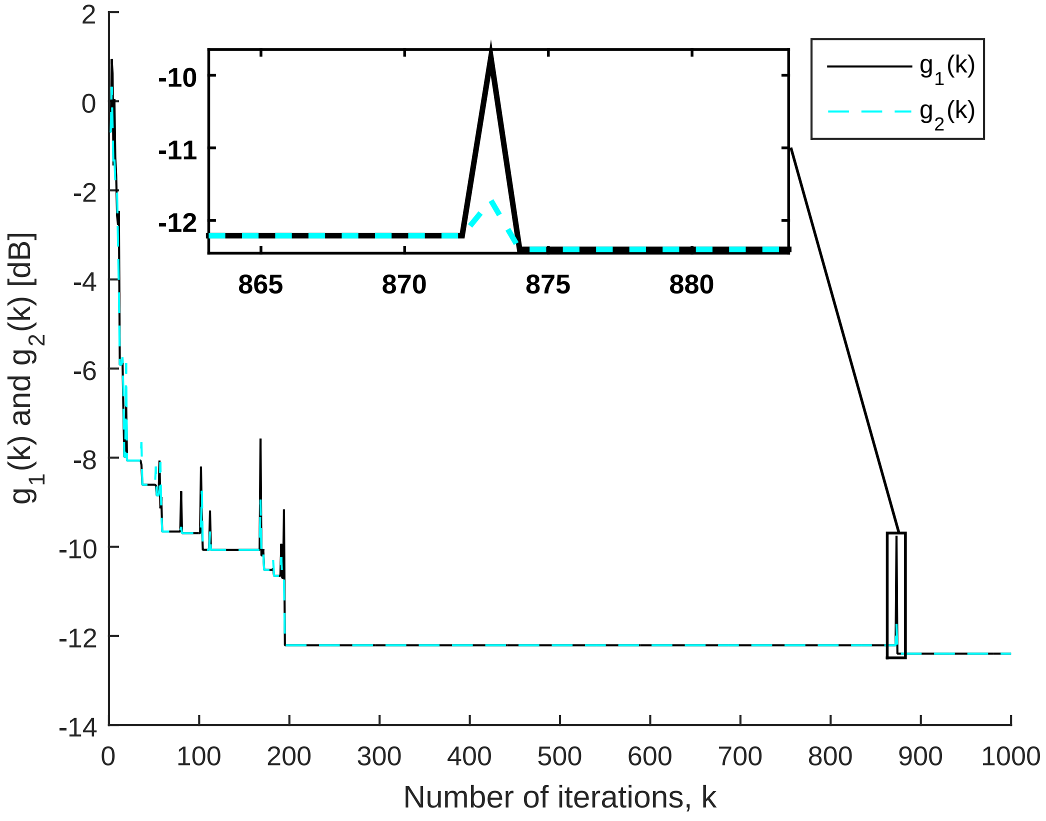


Figure 3.6: Values of $g_1(k)$ and $g_2(k)$ over the iterations for the SM-AP algorithm with $\boldsymbol{\gamma}(k)$ as the SC-CV when the noise bound is known, where $g_1(k)$ and $g_2(k)$ are the numerator and denominator of (3.40) in Theorem 3, when an update occurs; otherwise, $g_1(k) = \|\widetilde{\mathbf{w}}(k+1)\|^2$ and $g_2(k) = \|\widetilde{\mathbf{w}}(k)\|^2$.

general CV (which is not an adequate choice for the CV), the AP algorithm could not achieve fast convergence and low MSE simultaneously, as the SM-AP algorithm did. In addition, observe that $\boldsymbol{\gamma}(k) = \mathbf{n}(k)$ leads to the best results in terms of convergence rate and steady-state MSE, but the performance of the SM-AP with SC-CV is quite close. The average number of updates required by the SM-AP algorithm using the general CV, the SC-CV, and the noise CV are 35%, 9.7%, and 3.6%, respectively, implying that the last two CVs also have lower computational cost. It is worth noticing that even when using the general CV, the SM-AP algorithm still converges although it presents poor performance, as explained in Subsection 3.4.2.

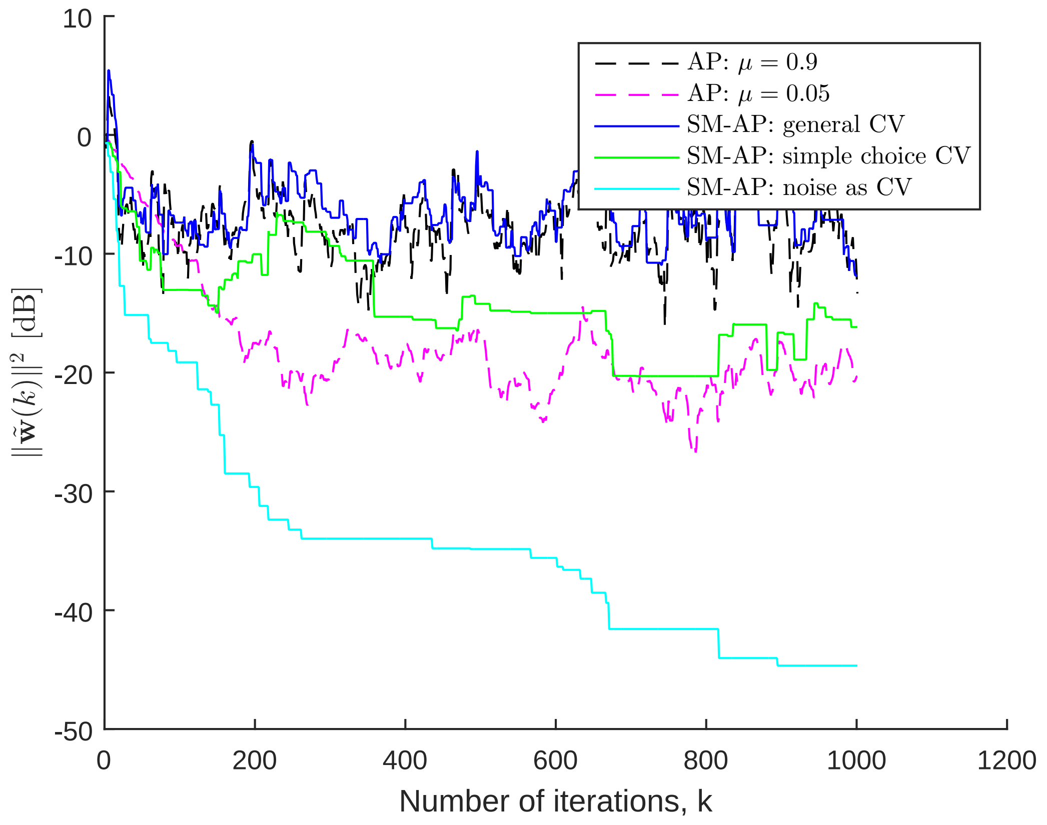


Figure 3.7: $\|\widetilde{\mathbf{w}}(k)\|^2 \triangleq \|\mathbf{w}(k) - \mathbf{w}_o\|^2$ for the AP and the SM-AP algorithms.

3.6 Conclusion

In this chapter, we addressed the robustness (in the sense of l_2-stability) of the SM-NLMS and the SM-AP algorithms. In addition to the already known advantages of the SM-NLMS algorithm over the NLMS algorithm, regarding accuracy and computational cost, in this chapter we demonstrated that: (i) the SM-NLMS algorithm is robust regardless the choice of its parameters and (ii) the SM-NLMS algorithm uses the input data very efficiently, i.e., it rarely produces a worse estimate $\mathbf{w}(k+1)$ during its update process. For the case where the noise bound is known, we explained how to set appropriately the parameter $\overline{\gamma}$ so that the SM-NLMS algorithm *never generates a worse estimate*, i.e., the sequence $\{\|\widetilde{\mathbf{w}}(k)\|^2\}$ (the squared Euclidean norm of the parameters deviation) becomes monotonously decreasing. For the case where the noise bound is unknown, we designed a time-varying parameter $\overline{\gamma}(k)$ that achieves simultaneously fast convergence and efficient use of the input data.

Unlike the SM-NLMS algorithm, we demonstrated that there exists a condition to

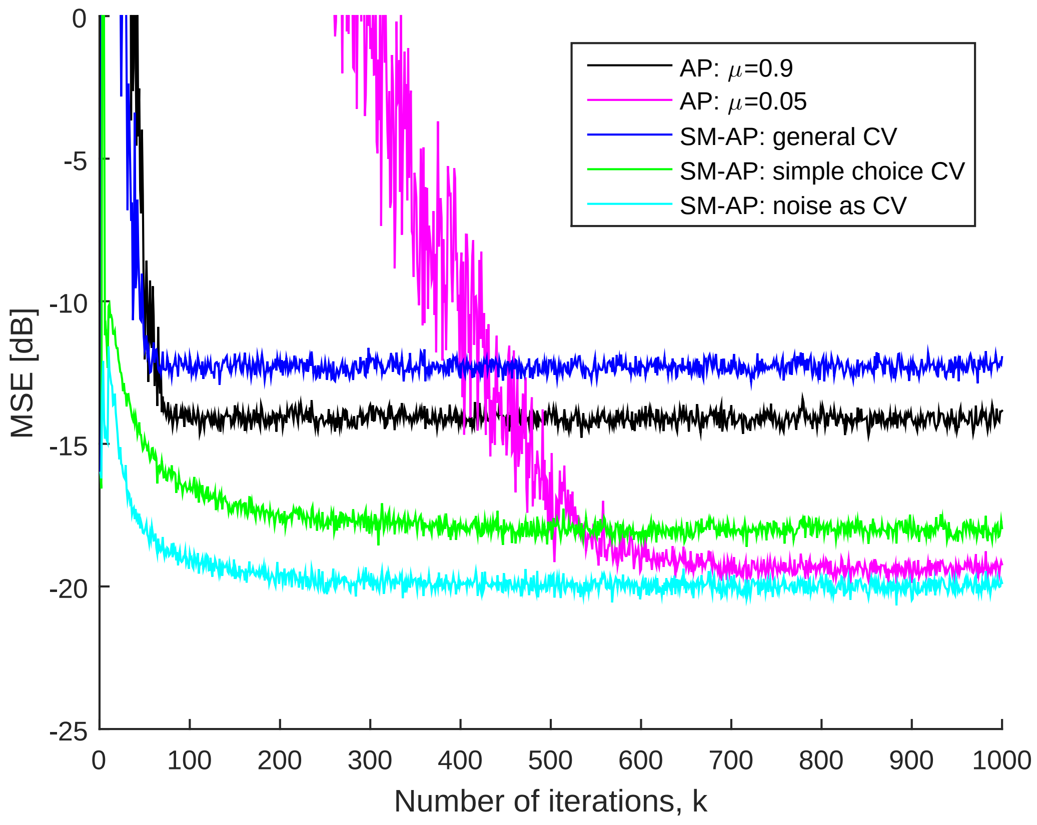


Figure 3.8: Learning curves for the AP and SM-AP algorithm using different constraint vectors.

guarantee the l_2-stability of the SM-AP algorithm. This robustness condition depends on a parameter known as the constraint vector (CV) $\boldsymbol{\gamma}(k)$. We proved the existence of vectors $\boldsymbol{\gamma}(k)$ satisfying such a condition, but practical choices remain unknown. In addition, it was shown that the SM-AP with an adequate CV uses the input data more efficiently than the AP algorithm.

We also demonstrated that both the SM-AP and SM-NLMS algorithms do not diverge, even when their parameters are not properly selected, provided the noise is bounded. Finally, numerical results that corroborate our study were presented.

Chapter 4

Trinion and Quaternion Set-Membership Affine Projection Algorithms

The quaternions are a number system that extends the complex numbers. They were introduced by William Rowan Hamilton in 1843 for the first time [52]. Quaternions have several applications in multivariate signal processing problems, such as color image processing [53, 54], wind profile prediction [55–57], and adaptive beamforming [58]. A wide family of quaternion based algorithms have been introduced in adaptive filtering literatures [59–62].

As a generalization of the complex domain, the quaternion domain provides a useful way to process 3- and 4-dimensional signals. Recently, several quaternion based adaptive filtering algorithms have appeared and they take benefit from the fact that the quaternion domain is a division algebra and it has a suitable data representation [63–65]. Therefore, the quaternion algorithms allow a coupling between the components of 3- and 4-dimensional processes. Also, the quaternion-valued algorithm results in better performance compared to the real-valued algorithms, since it accounts for the coupling of the wind measurements and can be developed to exploit the augmented quaternion statistics [55]. As a by-product, in comparison with the real-valued algorithms in $\mathbb{R}^3$ and $\mathbb{R}^4$, they show enhanced stability and more degrees of freedom in the control of the adaptation mechanism.

However, when the signals involved in the adaptation process have only three dimensions, i.e., one real and two imaginary components, we can apply the trinion based algorithms. Using a data set for wind profile prediction, the trinion-valued least mean square (TLMS) algorithm is proposed [66] and its learning speed is compared with the quaternion least mean square (QLMS) algorithm [56]. In the TLMS algorithm, the computational complexity is lower than QLMS algorithm, since the implementation of a full quaternion-valued multiplication requires 16 and 12 real-valued multiplications and additions, respectively. In the trinion case, to multiply two 3-D numbers we only need 9 and 6 real-valued multiplications and additions, respectively. The quaternion affine projection (QAP) algorithm [67] has been applied to predict noncircular real-world 4-D wind, but it can also be used to 3-D profile wind prediction.

Here we consider a powerful approach to decrease the computational complexity of an adaptive filter by employing set-membership filtering (SMF) approach [2, 9]. For real numbers, the set-membership NLMS [2, 9] and AP [2, 30, 31] algorithms were reviewed in Chapter 2. This chapter aims to generalize these algorithms to operate with trinion and quaternion numbers. The trinion number system is not a mathematical field since there are elements which are not invertible. Therefore, to address this drawback, we replace the non-invertible element with an invertible one. In the quaternion number system, each nonzero element has inverse while the product operation is not commutative. The proposed algorithms get around these drawbacks.

Finally, we apply the trinion based algorithms to predicting the wind profile and compare their competitive performance with the quaternion based algorithms. However, the quaternion algorithms require remarkably higher computational complexity compared to their trinion counterparts. Also, we study the quaternion adaptive beamforming as an application of the quaternion-valued algorithms. In this manner, we will reduce the number of involved sensors in the adaptation mechanism. As a result, we can decrease the computational complexity and the energy consumption of the system.

Part of the content of this chapter was published in [17]. This chapter introduces new data selective adaptive filtering algorithms for trinion and quaternion number systems $\mathbb{T}$ and $\mathbb{H}$. The work advances the set-membership trinion and quaternion-valued normalized least mean square (SMTNLMS and SMQNLMS) and the set-membership trinion and quaternion-valued affine projection (SMTAP and SMQAP) algorithms. Also, as individual cases, we obtain trinion and quaternion algorithms not employing the set-membership strategy.

This chapter is organized as follows. Short introductions to quaternions and trinions are provided in Sections 4.1 and 4.2, respectively. Section 4.3 briefly reviews the concept of SMF but instead of real numbers we use trinions and quaternions.

The new trinion based SMTAP algorithm is derived in Section 4.4. Section 4.5 introduces the quaternion based SMQAP algorithm. Section 4.6 reviews the application of quaternion-valued adaptive algorithms to adaptive beamforming. Simulations are presented in Section 4.7 and Section 4.8 contains the conclusions.

4.1 Quaternions

The quaternion number system is a non-commutative extension of complex numbers, denoted by $\mathbb{H}$. A quaternion $q \in \mathbb{H}$ is defined as [52]

$$q = q_a + q_b \imath + q_c \jmath + q_d \kappa, \tag{4.1}$$

where q_a, q_b, q_c, and q_d are in $\mathbb{R}$. q_a is the real component, while q_b, q_c, and q_d are the three imaginary components. The orthogonal unit imaginary axis vectors $\imath$, $\jmath$, and κ obey the following rules

$$\begin{aligned} \imath\jmath = \kappa \qquad \jmath\kappa = \imath \qquad \kappa\imath = \jmath, \\ \imath^2 = \jmath^2 = \kappa^2 = \imath\jmath\kappa = -1. \end{aligned} \tag{4.2}$$

Note that due to non-commutativity of the quaternion multiplication, we have $\jmath\imath = -\kappa \neq \imath\jmath$ for example. The element 1 is the identity element of $\mathbb{H}$, i.e., multiplication by 1 does nothing. The conjugate of a quaternion, denoted by q^*, is defined as

$$q^* = q_a - q_b \imath - q_c \jmath - q_d \kappa, \tag{4.3}$$

and the norm $|q|$ is given by

$$|q| = \sqrt{qq^*} = \sqrt{q_a^2 + q_b^2 + q_c^2 + q_d^2}. \tag{4.4}$$

The inverse of q is introduced as

$$q^{-1} = \frac{q^*}{|q|^2}. \tag{4.5}$$

Observe that q can be reformulated into the Cayley-Dickson [58] form as

$$q = \underbrace{(q_a + q_c\jmath)}_{z_1} + \imath \underbrace{(q_b + q_d\jmath)}_{z_2}, \tag{4.6}$$

where z_1 and z_2 are complex numbers.

The quaternion involutions are defined as follows [68, 69]

$$\begin{aligned} q^\imath &= -\imath q \imath = q_a + q_b\imath - q_c\jmath - q_d\kappa, \\ q^\jmath &= -\jmath q \jmath = q_a - q_b\imath + q_c\jmath - q_d\kappa, \\ q^\kappa &= -\kappa q \kappa = q_a - q_b\imath - q_c\jmath + q_d\kappa. \end{aligned} \tag{4.7}$$

Therefore, we can present the four real components of a quaternion q by the convolutions of q

$$\begin{aligned} q_a &= \frac{1}{4}(q + q^\imath + q^\jmath + q^\kappa), \\ q_b &= \frac{1}{4\imath}(q + q^\imath - q^\jmath - q^\kappa), \\ q_c &= \frac{1}{4\jmath}(q - q^\imath + q^\jmath - q^\kappa), \\ q_d &= \frac{1}{4\kappa}(q - q^\imath - q^\jmath + q^\kappa). \end{aligned} \tag{4.8}$$

These expressions allow us presenting any quadrivariate or quaternion-valued function $f(q)$ as [68]

$$f(q) = f(q_a, q_b, q_c, q_d) = f(q, q^\imath, q^\jmath, q^\kappa). \tag{4.9}$$

We know that the quaternion ring and $\mathbb{R}^4$ are isomorphic. Hence, by the same argument in the $\mathbb{CR}$ calculus [70], to introduce the duality between the derivatives of $f(q) \in \mathbb{H}$ and the derivatives of the corresponding quadrivariate real function $g(q_a, q_b, q_c, q_d) \in \mathbb{R}^4$, we begin with [69]

$$\begin{aligned} f(q) = f_a(q_a, q_b, q_c, q_d) &+ f_b(q_a, q_b, q_c, q_d)\imath + f_c(q_a, q_b, q_c, q_d)\jmath \\ &+ f_d(q_a, q_b, q_c, q_d)\kappa = g(q_a, q_b, q_c, q_d). \end{aligned} \tag{4.10}$$

The real variable function $g(q_a, q_b, q_c, q_d)$ has the following differential

$$\begin{aligned}dg =&\frac{\partial g}{\partial q_a}dq_a + \frac{\partial g}{\partial q_b}dq_b + \frac{\partial g}{\partial q_c}dq_c + \frac{\partial g}{\partial q_d}dq_d \\ =&\frac{\partial f(q)}{\partial q_a}dq_a + \frac{\partial f(q)}{\partial q_b}dq_b\imath + \frac{\partial f(q)}{\partial q_c}dq_c\jmath + \frac{\partial f(q)}{\partial q_d}dq_d\kappa. \end{aligned} \tag{4.11}$$

By using the relations in (4.8), the derivatives of the components of a quaternion q are given by

$$\begin{aligned} dq_a &= \frac{1}{4}(dq + dq^\imath + dq^\jmath + dq^\kappa), \\ dq_b &= \frac{-\imath}{4}(dq + dq^\imath - dq^\jmath - dq^\kappa), \\ dq_c &= \frac{-\jmath}{4}(dq - dq^\imath + dq^\jmath - dq^\kappa), \\ dq_d &= \frac{-\kappa}{4}(dq - dq^\imath - dq^\jmath + dq^\kappa). \end{aligned} \tag{4.12}$$

Also, using (4.9) we obtain

$$\begin{aligned} df(q) =&\frac{\partial f(q, q^\imath, q^\jmath, q^\kappa)}{\partial q}dq + \frac{\partial f(q, q^\imath, q^\jmath, q^\kappa)}{\partial q^\imath}dq^\imath \\ &+ \frac{\partial f(q, q^\imath, q^\jmath, q^\kappa)}{\partial q^\jmath}dq^\jmath + \frac{\partial f(q, q^\imath, q^\jmath, q^\kappa)}{\partial q^\kappa}dq^\kappa. \end{aligned} \tag{4.13}$$

Therefore, by replacing the components of dq from (4.12) in Equation (4.11), and solving for the coefficients of dq, $dq^\imath$, $dq^\jmath$, dq^κ from (4.11) and (4.13), we will obtain the $\mathbb{HR}$-derivatives identities as follows

$$\begin{bmatrix} \frac{\partial f(q,q^\imath,q^\jmath,q^\kappa)}{\partial q} \\ \frac{\partial f(q,q^\imath,q^\jmath,q^\kappa)}{\partial q^\imath} \\ \frac{\partial f(q,q^\imath,q^\jmath,q^\kappa)}{\partial q^\jmath} \\ \frac{\partial f(q,q^\imath,q^\jmath,q^\kappa)}{\partial q^\kappa} \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1 & -\imath & -\jmath & -\kappa \\ 1 & -\imath & \jmath & \kappa \\ 1 & \imath & -\jmath & \kappa \\ 1 & \imath & \jmath & -\kappa \end{bmatrix}\begin{bmatrix} \frac{\partial f}{\partial q_a} \\ \frac{\partial f}{\partial q_b} \\ \frac{\partial f}{\partial q_c} \\ \frac{\partial f}{\partial q_d} \end{bmatrix}. \tag{4.14}$$

Our interest is in the derivative $\frac{\partial f(q,q^\imath,q^\jmath,q^\kappa)}{\partial q}$, thus the gradient of $f(q)$ with respect to q is given by [69]

$$\nabla_q f = \frac{1}{4}(\frac{\partial f}{\partial q_a} - \frac{\partial f}{\partial q_b}\imath - \frac{\partial f}{\partial q_c}\jmath - \frac{\partial f}{\partial q_d}\kappa) = \frac{1}{4}(\nabla_{q_a} f - \nabla_{q_b} f\imath - \nabla_{q_c} f\jmath - \nabla_{q_d} f\kappa). \tag{4.15}$$

The real values elements q_a, q_b, q_c, q_d of a quaternion q can be presented in terms of q^*, $q^{\imath^*}$, $q^{\jmath^*}$, q^{κ^*} as follows [69]

$$
\begin{aligned}
q_a &= \frac{1}{4}(q^* + q^{\imath^*} + q^{\jmath^*} + q^{\kappa^*}),\\
q_b &= \frac{1}{4\imath}(-q - q^{\imath^*} + q^{\jmath^*} + q^{\kappa^*}),\\
q_c &= \frac{1}{4\jmath}(-q + q^{\imath^*} - q^{\jmath^*} + q^{\kappa^*}),\\
q_d &= \frac{1}{4\kappa}(-q^* + q^{\imath^*} + q^{\jmath^*} - q^{\kappa^*}).
\end{aligned}
\tag{4.16}
$$

Then the derivative of the function $f(q) = f(q^*, q^{\imath^*}, q^{\jmath^*}, q^{\kappa^*})$ can be expressed as

$$
\begin{aligned}
df(q) =& \frac{\partial f(q^*, q^{\imath^*}, q^{\jmath^*}, q^{\kappa^*})}{\partial q^*} dq^* + \frac{\partial f(q^*, q^{\imath^*}, q^{\jmath^*}, q^{\kappa^*})}{\partial q^{\imath^*}} dq^{\imath^*}\\
&+ \frac{\partial f(q^*, q^{\imath^*}, q^{\jmath^*}, q^{\kappa^*})}{\partial q^{\jmath^*}} dq^{\jmath^*} + \frac{\partial f(q^*, q^{\imath^*}, q^{\jmath^*}, q^{\kappa^*})}{\partial q^{\kappa^*}} dq^{\kappa^*}.
\end{aligned}
\tag{4.17}
$$

Also, the derivative of the quadrivariate $g(q_a, q_b, q_c, q_d)$ is given by

$$
dg(q_a, q_b, q_c, q_d) = A dq^* + B dq^{\imath^*} + C dq^{\jmath^*} + D dq^{\kappa^*}. \tag{4.18}
$$

By the same argument above, if we solve for the coefficients of dq^*, $dq^{\imath^*}$, $dq^{\jmath^*}$, dq^{κ^*} then we will obtain the $\mathbb{HR}^*$-derivatives identities,

$$
\begin{bmatrix}
\frac{\partial f(q^*,q^{\imath^*},q^{\jmath^*},q^{\kappa^*})}{\partial q^*}\\
\frac{\partial f(q^*,q^{\imath^*},q^{\jmath^*},q^{\kappa^*})}{\partial q^{\imath^*}}\\
\frac{\partial f(q^*,q^{\imath^*},q^{\jmath^*},q^{\kappa^*})}{\partial q^{\jmath^*}}\\
\frac{\partial f(q^*,q^{\imath^*},q^{\jmath^*},q^{\kappa^*})}{\partial q^{\kappa^*}}
\end{bmatrix}
= \frac{1}{4}
\begin{bmatrix}
1 & \imath & \jmath & \kappa\\
1 & \imath & -\jmath & -\kappa\\
1 & -\imath & \jmath & -\kappa\\
1 & -\imath & -\jmath & \kappa
\end{bmatrix}
\begin{bmatrix}
\frac{\partial f}{\partial q_a}\\
\frac{\partial f}{\partial q_b}\\
\frac{\partial f}{\partial q_c}\\
\frac{\partial f}{\partial q_d}
\end{bmatrix}.
\tag{4.19}
$$

The derivative $\frac{\partial f(q^*,q^{\imath^*},q^{\jmath^*},q^{\kappa^*})}{\partial q^*}$ is of particular interest, thus the gradient of $f(q)$ with respect to q^* is given by [69]

$$
\nabla_{q^*} f = \frac{1}{4}\left(\frac{\partial f}{\partial q_a} + \frac{\partial f}{\partial q_b}\imath + \frac{\partial f}{\partial q_c}\jmath + \frac{\partial f}{\partial q_d}\kappa\right) = \frac{1}{4}(\nabla_{q_a} f + \nabla_{q_b} f \imath + \nabla_{q_c} f \jmath + \nabla_{q_d} f \kappa). \tag{4.20}
$$

4.2 Trinions

As a group, the trinion number system $\mathbb{T}$ is isomorphic to $\mathbb{R}^3$. A number v in $\mathbb{T}$ is composed of one real part, v_a, and two imaginary parts, v_b and v_c,

$$v = v_a + v_b\bar{\imath} + v_c\bar{\jmath}. \tag{4.21}$$

The number system $\mathbb{T}$ has three operations: addition, scalar multiplication, and trinion multiplication. The sum of two elements of $\mathbb{T}$ is defined to be their sum as elements of $\mathbb{R}^3$. Similarly the product of an element of $\mathbb{T}$ by a real number is defined to be the same as the product by a scalar in $\mathbb{R}^3$. To make a commutative algebraic group of the basis elements 1, $\bar{\imath}$, and $\bar{\jmath}$ the following rules apply [71]

$$\bar{\imath}^2 = \bar{\jmath},\ \overline{\imath\jmath} = \overline{\jmath\imath} = -1,\ \bar{\jmath}^2 = -\bar{\imath}. \tag{4.22}$$

Trinions with these rules set a commutative mathematical ring, i.e., $vw = wv$ for $v, w \in \mathbb{T}$. The basis element 1 will be the identity element of $\mathbb{T}$, meaning that multiplication by 1 does nothing. The conjugate of v is given by [66]

$$v^* = v_a - v_b\bar{\jmath} - v_c\bar{\imath}, \tag{4.23}$$

and the norm by [66]

$$|v| = \sqrt{\Re(vv^*)} = \sqrt{v_a^2 + v_b^2 + v_c^2}. \tag{4.24}$$

The inverse of v, if exists, is $w = (w_a + w_b\bar{\imath} + w_c\bar{\jmath}) \in \mathbb{T}$ such that $vw = wv = 1$. To solve this equation we consider $v = [v_a\ v_b\ v_c]^T$ and $w = [w_a\ w_b\ w_c]^T$ then we get

$$\begin{cases} v_a w_a - v_c w_b - v_b w_c = 1, \\ v_b w_a + v_a w_b - v_c w_c = 0, \\ v_c w_a + v_b w_b + v_a w_c = 0, \end{cases} \tag{4.25}$$

or in the matrix form $\mathbf{A}w = [1\ 0\ 0]^T$ where $\mathbf{A}$ is given by

$$\mathbf{A} = \begin{bmatrix} v_a & -v_c & -v_b \\ v_b & v_a & -v_c \\ v_c & v_b & v_a \end{bmatrix}, \tag{4.26}$$

thus $w = \mathbf{A}^{-1}[1\ 0\ 0]^T$. When the determinant of $\mathbf{A}$ is zero, the inverse of v does not exist. In order to get around this problem when the determinant of $\mathbf{A}$ is zero, we define $\mathbf{A} = \delta\mathbf{I}$ where δ is a small positive constant and $\mathbf{I}$ is a 3×3 identity matrix. Note that $\mathbf{A}$ is replaced by the identity matrix multiplied by a small constant in order to avoid numerical problems in the matrix inversion. This strategy avoids division by zero in the trinion-valued algorithms. We will now define $v^{-1} = \mathbf{A}^{-1}[1\ 0\ 0]^T$.

In the field of complex numbers, a variable z and its conjugate z^* can be considered as two independent variables, so that the complex-valued gradient can be defined [72]. As far as we know, the trinion involutions, $v^{\bar{\imath}}$ and $v^{\bar{\jmath}}$, are not available in general. In this chapter, we use the following formulas for the gradients of a function $f(v)$ with respect to the trinion-valued variable v and its conjugate [66]

$$\begin{aligned} \nabla_v f &= \frac{1}{3}(\nabla_{v_a} f - \nabla_{v_b} f\bar{\jmath} - \nabla_{v_c} f\bar{\imath}), \\ \nabla_{v^*} f &= \frac{1}{3}(\nabla_{v_a} f + \nabla_{v_b} f\bar{\imath} + \nabla_{v_c} f\bar{\jmath}), \end{aligned} \tag{4.27}$$

where $v = v_a + v_b\bar{\imath} + v_c\bar{\jmath}$.

4.3 Set-Membership Filtering (SMF) in $\mathbb{T}$ and $\mathbb{H}$

The target of the SMF is to design $\mathbf{w}$ such that the magnitude of the estimation error is upper bounded by a predetermined parameter $\overline{\gamma}$. The value of $\overline{\gamma}$ can change with the specific application. If the value of $\overline{\gamma}$ is suitably selected, there are many valid estimates for $\mathbf{w}$. Suppose that $\mathcal{S}$ denotes the set of all possible input-desired data pairs $(\mathbf{x}, d)$ of interest and define Θ as the set of all vectors $\mathbf{w}$ whose magnitudes of their estimation errors are upper bounded by $\overline{\gamma}$ whenever $(\mathbf{x}, d) \in \mathcal{S}$. The set Θ is named feasibility set and is given by

$$\Theta \triangleq \bigcap_{(\mathbf{x},d)\in\mathcal{S}} \{\mathbf{w} \in \mathbb{F}^{N+1} : |d - \mathbf{w}^H\mathbf{x}| \leq \overline{\gamma}\}, \tag{4.28}$$

where $\mathbb{F}$ is $\mathbb{T}$ or $\mathbb{H}$. Let's define the constraint set $\mathcal{H}(k)$ consisting of all vectors $\mathbf{w}$ such that their estimation errors at time instant k are upper bounded in magnitude by $\overline{\gamma}$,

$$\mathcal{H}(k) \triangleq \{\mathbf{w} \in \mathbb{F}^{N+1} : |d(k) - \mathbf{w}^H\mathbf{x}(k)| \leq \overline{\gamma}\}. \tag{4.29}$$

The membership set $\psi(k)$ defined as

$$\psi(k) \triangleq \bigcap_{i=0}^{k} \mathcal{H}(i) \tag{4.30}$$

will include Θ and will coincide with Θ if all data pairs in $\mathcal{S}$ are traversed up to time instant k. Owing to difficulties to compute $\psi(k)$, adaptive approaches are required [9]. The easiest route is to compute a point estimate using, for example, the information provided by the constraint set $\mathcal{H}(k)$ like in the set-membership NLMS algorithm [9], or several previous constraint sets as is done in the set-membership affine projection algorithm [31].

4.4 SMTAP Algorithm

In this section, we propose the SMTAP algorithm. This trinion-valued algorithm is the counterpart of the real-valued SM-AP algorithm. Then we derive the update equations for the simpler algorithms related to the normalized LMS algorithm.

The membership set $\psi(k)$ defined in (4.30) encourages the use of more constraint sets in the update. Therefore, we elaborate an algorithm whose updates belong to a set composed of $L+1$ constraint sets.

For this purpose, we express $\psi(k)$ as

$$\psi(k) = \bigcap_{i=0}^{k-L-1} \mathcal{H}(i) \bigcap_{j=k-L}^{k} \mathcal{H}(j) = \psi^{k-L-1}(k) \bigcap \psi^{L+1}(k), \tag{4.31}$$

where $\psi^{L+1}(k)$ indicates the intersection of the $L+1$ last constraint sets, and $\psi^{k-L-1}(k)$ represents the intersection of the first $k-L$ constraint sets. Our goal is to formulate an algorithm whose coefficient update belongs to the last $L+1$ constraint sets, i.e., $\mathbf{w}(k+1) \in \psi^{L+1}(k)$.

Assume that $\mathcal{S}(k-i)$ denotes the set which includes all vectors $\mathbf{w}$ such that $d(k-i) - \mathbf{w}^H\mathbf{x}(k-i) = \gamma_i(k)$, for $i = 0, \cdots, L$. All choices for $\gamma_i(k)$ satisfying the bound constraint are valid. That is, if all $\gamma_i(k)$ are selected such that $|\gamma_i(k)| \leq \overline{\gamma}$, then $\mathcal{S}(k-i) \in \mathcal{H}(k-i)$, for $i = 0, \cdots, L$.

The objective function which we ought to minimize can now be stated. A coefficient update is implemented whenever $\mathbf{w}(k) \notin \psi^{L+1}(k)$ as follows

$$\begin{aligned}
&\min \frac{1}{2}\|\mathbf{w}(k+1) - \mathbf{w}(k)\|^2 \\
&\text{subject to:} \\
&\mathbf{d}(k) - (\mathbf{w}^H(k+1)\mathbf{X}(k))^T = \boldsymbol{\gamma}(k),
\end{aligned} \tag{4.32}$$

where

$\mathbf{d}(k) \in \mathbb{T}^{(L+1)\times 1}$	contains the desired output from the $L+1$ last time instants;
$\boldsymbol{\gamma}(k) \in \mathbb{T}^{(L+1)\times 1}$	specifies the point in $\psi^{L+1}(k)$;
$\mathbf{X}(k) \in \mathbb{T}^{(N+1)\times(L+1)}$	contains the corresponding input vectors, i.e.,

$$\begin{aligned}
\mathbf{d}(k) &= [d(k)\ d(k-1)\ \cdots\ d(k-L)]^T, \\
\boldsymbol{\gamma}(k) &= [\gamma_0(k)\ \gamma_1(k)\ \cdots\ \gamma_L(k)]^T, \\
\mathbf{X}(k) &= [\mathbf{x}(k)\ \mathbf{x}(k-1)\ \cdots\ \mathbf{x}(k-L)],
\end{aligned} \tag{4.33}$$

with $\mathbf{x}(k)$ being the input-signal vector

$$\mathbf{x}(k) = [x(k)\ x(k-1)\ \cdots\ x(k-N)]^T. \tag{4.34}$$

If we use the method of Lagrange multipliers to transform a constrained minimization into an unconstrained one, then we have to minimize

$$\begin{aligned}
F[\mathbf{w}(k+1)] =& \frac{1}{2}\|\mathbf{w}(k+1) - \mathbf{w}(k)\|^2 \\
&+ \Re\{\boldsymbol{\lambda}^T(k)[\mathbf{d}(k) - (\mathbf{w}^H(k+1)\mathbf{X}(k))^T - \boldsymbol{\gamma}(k)]\},
\end{aligned} \tag{4.35}$$

where $\boldsymbol{\lambda}(k) \in \mathbb{T}^{(L+1)\times 1}$ is a vector of Lagrange multipliers. To find the minimum solution, we must calculate the following gradient

$$\begin{aligned}
\nabla_{\mathbf{w}^*(k+1)} F[\mathbf{w}(k+1)] =& \frac{1}{3}\Big[\nabla_{\mathbf{w}_a(k+1)} F[\mathbf{w}(k+1)] + \nabla_{\mathbf{w}_b(k+1)} F[\mathbf{w}(k+1)]\bar{\imath} \\
&+ \nabla_{\mathbf{w}_c(k+1)} F[\mathbf{w}(k+1)]\bar{\jmath}\Big].
\end{aligned} \tag{4.36}$$

In order to find the above gradient, we ought to calculate the cost function $F[\mathbf{w}(k+1)]$

as a function of real-valued variables. As a result we have,

$$\begin{aligned}\|\mathbf{w}(k+1)-\mathbf{w}(k)\|^2 =& \|\mathbf{w}_a(k+1)-\mathbf{w}_a(k)\|^2+\|\mathbf{w}_b(k+1)-\mathbf{w}_b(k)\|^2\\ &+\|\mathbf{w}_c(k+1)-\mathbf{w}_c(k)\|^2.\end{aligned} \tag{4.37}$$

We drop the time index 'k' for the sake of compact notation. In order to find the second term in (4.35) as a real-valued term we perform the following calculations,

$$\begin{aligned}&\Re\{\boldsymbol{\lambda}^T[\mathbf{d}-\mathbf{X}^T\mathbf{w}^*(k+1)-\boldsymbol{\gamma}]\} = \Re\{(\boldsymbol{\lambda}_a^T+\boldsymbol{\lambda}_b^T\imath+\boldsymbol{\lambda}_c^T\jmath)[(\mathbf{d}_a+\mathbf{d}_b\imath+\mathbf{d}_c\jmath)\\
&\quad -(\mathbf{X}_a^T+\mathbf{X}_b^T\imath+\mathbf{X}_c^T\jmath)(\mathbf{w}_a(k+1)-\mathbf{w}_b(k+1)\jmath-\mathbf{w}_c(k+1)\imath)-(\boldsymbol{\gamma}_a+\boldsymbol{\gamma}_b\imath+\boldsymbol{\gamma}_c\jmath)]\}\\
&=\Re\{(\boldsymbol{\lambda}_a^T+\boldsymbol{\lambda}_b^T\imath+\boldsymbol{\lambda}_c^T\jmath)[(\mathbf{d}_a-\mathbf{X}_a^T\mathbf{w}_a(k+1)-\mathbf{X}_b^T\mathbf{w}_b(k+1)-\mathbf{X}_c^T\mathbf{w}_c(k+1)-\boldsymbol{\gamma}_a)\\
&\quad+(\mathbf{d}_b-\mathbf{X}_b^T\mathbf{w}_a(k+1)+\mathbf{X}_a^T\mathbf{w}_c(k+1)-\mathbf{X}_c^T\mathbf{w}_b(k+1)-\boldsymbol{\gamma}_b)\imath\\
&\quad+(\mathbf{d}_c-\mathbf{X}_c^T\mathbf{w}_a(k+1)+\mathbf{X}_a^T\mathbf{w}_b(k+1)+\mathbf{X}_b^T\mathbf{w}_c(k+1)-\boldsymbol{\gamma}_c)\jmath]\}\\
&=\boldsymbol{\lambda}_a^T(\mathbf{d}_a-\mathbf{X}_a^T\mathbf{w}_a(k+1)-\mathbf{X}_b^T\mathbf{w}_b(k+1)-\mathbf{X}_c^T\mathbf{w}_c(k+1)-\boldsymbol{\gamma}_a)\\
&\quad-\boldsymbol{\lambda}_b^T(\mathbf{d}_c-\mathbf{X}_c^T\mathbf{w}_a(k+1)+\mathbf{X}_a^T\mathbf{w}_b(k+1)+\mathbf{X}_b^T\mathbf{w}_c(k+1)-\boldsymbol{\gamma}_c)\\
&\quad-\boldsymbol{\lambda}_c^T(\mathbf{d}_b-\mathbf{X}_b^T\mathbf{w}_a(k+1)+\mathbf{X}_a^T\mathbf{w}_c(k+1)-\mathbf{X}_c^T\mathbf{w}_b(k+1)-\boldsymbol{\gamma}_b).\end{aligned} \tag{4.38}$$

Therefore, by (4.35), (4.37), and (4.38) we obtain

$$F[\mathbf{w}(k+1)] = \frac{1}{2}\text{Eq.}(4.37)+\text{Eq.}(4.38). \tag{4.39}$$

Thus, the three component-wise gradients can be attained as

$$\nabla_{\mathbf{w}_a(k+1)}F[\mathbf{w}(k+1)] =(\mathbf{w}_a(k+1)-\mathbf{w}_a(k))-\boldsymbol{\lambda}_a^T\mathbf{X}_a^T+\boldsymbol{\lambda}_b^T\mathbf{X}_c^T+\boldsymbol{\lambda}_c^T\mathbf{X}_b^T, \tag{4.40}$$

$$\nabla_{\mathbf{w}_b(k+1)}F[\mathbf{w}(k+1)] =(\mathbf{w}_b(k+1)-\mathbf{w}_b(k))-\boldsymbol{\lambda}_a^T\mathbf{X}_b^T-\boldsymbol{\lambda}_b^T\mathbf{X}_a^T+\boldsymbol{\lambda}_c^T\mathbf{X}_c^T, \tag{4.41}$$

$$\nabla_{\mathbf{w}_c(k+1)}F[\mathbf{w}(k+1)] =(\mathbf{w}_c(k+1)-\mathbf{w}_c(k))-\boldsymbol{\lambda}_a^T\mathbf{X}_c^T-\boldsymbol{\lambda}_b^T\mathbf{X}_b^T-\boldsymbol{\lambda}_c^T\mathbf{X}_a^T. \tag{4.42}$$

On the other hand, we have

$$\begin{aligned}\mathbf{X}\boldsymbol{\lambda} =&(\mathbf{X}_a+\mathbf{X}_b\imath+\mathbf{X}_c\jmath)(\boldsymbol{\lambda}_a+\boldsymbol{\lambda}_b\imath+\boldsymbol{\lambda}_c\jmath)\\
=&(\mathbf{X}_a\boldsymbol{\lambda}_a-\mathbf{X}_b\boldsymbol{\lambda}_c-\mathbf{X}_c\boldsymbol{\lambda}_b)+(\mathbf{X}_a\boldsymbol{\lambda}_b+\mathbf{X}_b\boldsymbol{\lambda}_a-\mathbf{X}_c\boldsymbol{\lambda}_c)\imath\\
&+(\mathbf{X}_a\boldsymbol{\lambda}_c+\mathbf{X}_c\boldsymbol{\lambda}_a+\mathbf{X}_b\boldsymbol{\lambda}_b)\jmath.\end{aligned} \tag{4.43}$$

Overall, by employing Equations (4.36) and (4.40)-(4.43), we get,

$$\begin{aligned}\nabla_{\mathbf{w}^*(k+1)}F[\mathbf{w}(k+1)] =& \frac{1}{3}\{[(\mathbf{w}_a(k+1)-\mathbf{w}_a(k))-(\mathbf{X}(k)\boldsymbol{\lambda}(k))_a] \\ &+[(\mathbf{w}_b(k+1)-\mathbf{w}_b(k))-(\mathbf{X}(k)\boldsymbol{\lambda}(k))_b]\bar{\imath} \\ &+[(\mathbf{w}_c(k+1)-\mathbf{w}_c(k))-(\mathbf{X}(k)\boldsymbol{\lambda}(k))_c]\bar{\jmath}\} \\ =&\frac{1}{3}[\mathbf{w}(k+1)-\mathbf{w}(k)-\mathbf{X}(k)\boldsymbol{\lambda}(k)]. \end{aligned} \tag{4.44}$$

After setting the above equation equal to zero, we obtain

$$\mathbf{w}(k+1)=\mathbf{w}(k)+\mathbf{X}(k)\boldsymbol{\lambda}(k). \tag{4.45}$$

If we substitute (4.45) in the constraint relation (4.32) the following expression results,

$$\mathbf{X}^T(k)\mathbf{X}^*(k)\boldsymbol{\lambda}^*(k)=\mathbf{d}(k)-\mathbf{X}^T(k)\mathbf{w}^*(k)-\boldsymbol{\gamma}(k)=(\mathbf{e}(k)-\boldsymbol{\gamma}(k)). \tag{4.46}$$

From the above equation we get $\boldsymbol{\lambda}(k)$ as

$$\boldsymbol{\lambda}(k)=(\mathbf{X}^H(k)\mathbf{X}(k))^{-1}(\mathbf{e}(k)-\boldsymbol{\gamma}(k))^*, \tag{4.47}$$

where

$$\mathbf{e}(k)=[e(k)\ \epsilon(k-1)\ \cdots\ \epsilon(k-L)]^T, \tag{4.48}$$

with $e(k)=d(k)-\mathbf{w}^H(k)\mathbf{x}(k)$, and $\epsilon(k-i)=d(k-i)-\mathbf{w}^H(k)\mathbf{x}(k-i)$ for $i=1,\cdots,L$. We can now conclude the SMTAP algorithm by starting from (4.45) with $\boldsymbol{\lambda}(k)$ being given by (4.47), i.e.,

$$\mathbf{w}(k+1)=\begin{cases}\mathbf{w}(k)+\mathbf{p}_{\text{ap}}(k) & \text{if } |e(k)|>\overline{\gamma}, \\ \mathbf{w}(k) & \text{otherwise,}\end{cases} \tag{4.49}$$

where

$$\mathbf{p}_{\text{ap}}(k)=\mathbf{X}(k)(\mathbf{X}^H(k)\mathbf{X}(k))^{-1}(\mathbf{e}(k)-\boldsymbol{\gamma}(k))^*. \tag{4.50}$$

Remark 1: In order to check if an update $\mathbf{w}(k+1)$ is required, we only have to test if $\mathbf{w}(k)\notin\mathcal{H}(k)$ since in the previous updates $\mathbf{w}(k)\in\mathcal{H}(k-i+1)$ is guaranteed for $i = 2, \cdots, L+1$.

Remark 2: For the initial time instants $k < L+1$, i.e., during initialization, only the knowledge of $\mathcal{H}(i)$ for $i = 0, 1, \cdots, k$ is available. As a consequence, if an update is required for $k < L+1$, the algorithm is implemented with the available $k+1$ accessible constraint sets.

Remark 3: By adopting the bound $\overline{\gamma} = 0$, the algorithm will convert to the trinion affine projection (TAP) algorithm with unity step size which is the generalization of the conventional real-valued AP algorithm in $\mathbb{T}$. Therefore, the TAP algorithm can be described as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu \mathbf{p}'_{\text{ap}}(k), \tag{4.51}$$

where μ is the convergence factor and

$$\mathbf{p}'_{\text{ap}}(k) = \mathbf{X}(k)(\mathbf{X}^H(k)\mathbf{X}(k))^{-1}\mathbf{e}^*(k). \tag{4.52}$$

Note that we can utilize (4.49) and derive the update equation of the SMTNLMS algorithm. In this case we have to evade data-reusing in (4.49), $L = 0$, so that the updating equation becomes,

$$\mathbf{w}(k+1) = \begin{cases} \mathbf{w}(k) + \mathbf{p}(k) & \text{if } |e(k)| > \overline{\gamma}, \\ \mathbf{w}(k) & \text{otherwise,} \end{cases} \tag{4.53}$$

where

$$\mathbf{p}(k) = \mathbf{x}(k)(\mathbf{x}^H(k)\mathbf{x}(k))^{-1}(e(k) - \gamma(k))^*, \tag{4.54}$$

$$e(k) = d(k) - \mathbf{w}^H(k)\mathbf{x}(k). \tag{4.55}$$

We will now choose $\gamma(k) = \frac{\overline{\gamma}e(k)}{|e(k)|}$, hence from (4.53) we attain the SMTNLMS update equation as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu(k)\mathbf{x}(k)(\mathbf{x}^H(k)\mathbf{x}(k))^{-1}e^*(k), \tag{4.56}$$

where

$$\mu(k) = \begin{cases} 1 - \frac{\overline{\gamma}}{|e(k)|} & \text{if } |e(k)| > \overline{\gamma}, \\ 0 & \text{otherwise.} \end{cases} \tag{4.57}$$

Recalling that the normalized LMS algorithm can be derived as a particular case of AP algorithm for $L = 0$.

Remark 4: By choosing the bound $\overline{\gamma} = 0$ in (4.56), the algorithm will reduce to the TNLMS algorithm with unity step size which is the generalization of the popular real-valued NLMS algorithm in $\mathbb{T}$. As a result, TNLMS algorithm can be described as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu\mathbf{x}(k)(\mathbf{x}^H(k)\mathbf{x}(k))^{-1}e^*(k), \tag{4.58}$$

where μ is the convergence factor.

4.5 SMQAP Algorithm

This section outlines the derivation of the SMQAP algorithm. Then we obtain an update equation for the SMQNLMS algorithm that follows the same steps as the derivation of the SMTNLMS algorithm. The SMQAP and the SMQNLMS algorithms are the quaternion versions of the real-valued SM-AP and SM-NLMS algorithms, respectively.

The membership set $\psi(k)$ introduced in (4.30) suggests the use of more constraint sets in the update. Let us express $\psi(k)$ as in (4.31), our purpose is to derive an algorithm whose coefficient update belongs to the last $L+1$ constraint set, i.e., $\mathbf{w}(k+1) \in \psi^{L+1}(k)$. Suppose that $\mathcal{S}(k-i)$ describes the set which contains all vectors $\mathbf{w}$ such that $d(k-i) - \mathbf{w}^H\mathbf{x}(k-i) = \gamma_i(k)$, for $i = 0, \cdots, L$. All choices for $\gamma_i(k)$ satisfying the bound constraint are valid. That is, if all $\gamma_i(k)$ are chosen such that $|\gamma_i(k)| \leq \overline{\gamma}$, then $\mathcal{S}(k-i) \in \mathcal{H}(k-i)$, for $i = 0, \cdots, L$.

The objective function to be minimized in case of the SMQAP algorithm can be stated as follows: perform a coefficient update whenever $\mathbf{w}(k) \notin \psi^{L+1}(k)$ as in Equation (4.32). Note that $\mathbf{d}(k), \boldsymbol{\gamma}(k) \in \mathbb{H}^{(L+1)\times 1}$, $\mathbf{X}(k) \in \mathbb{H}^{(N+1)\times(L+1)}$, and $\mathbf{x}(k)$ are defined as in (4.33) and (4.34).

By employing the method of Lagrange multipliers, the unconstrained function to be minimized becomes as in Equation (4.35), where $\boldsymbol{\lambda}(k) \in \mathbb{H}^{(L+1)\times 1}$ is a vector of Lagrange multipliers. After setting the gradient of $F[\mathbf{w}(k+1)]$ with respect to $\mathbf{w}^*(k+1)$ equal to zero, we will get the equation

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{X}(k)\boldsymbol{\lambda}(k). \tag{4.59}$$

Then, by invoking the constraints in (4.32), the expression of $\boldsymbol{\lambda}(k)$ is as

$$\boldsymbol{\lambda}(k) = (\mathbf{X}^H(k)\mathbf{X}(k))^{-1}(\mathbf{e}(k) - \boldsymbol{\gamma}(k))^*, \tag{4.60}$$

where $\mathbf{e}(k)$ is defined as in (4.48). Finally, the update equation for the SMQAP algorithm is given by

$$\mathbf{w}(k+1) = \begin{cases} \mathbf{w}(k) + \mathbf{q}_{\text{ap}}(k) & \text{if } |e(k)| > \overline{\gamma}, \\ \mathbf{w}(k) & \text{otherwise,} \end{cases} \tag{4.61}$$

where

$$\mathbf{q}_{\text{ap}}(k) = \mathbf{X}(k)(\mathbf{X}^H(k)\mathbf{X}(k))^{-1}(\mathbf{e}(k) - \boldsymbol{\gamma}(k))^*. \tag{4.62}$$

Note that the *Remarks 1* and *2* of Subsection 4.4 also apply to the SMQAP algorithm.

Remark 5: We can quickly verify that adopting the bound $\overline{\gamma} = 0$, the algorithm will reduce to QAP algorithm [67] with unity step size. Therefore, the QAP algorithm cab be expressed as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu\mathbf{X}(k)(\mathbf{X}^H(k)\mathbf{X}(k))^{-1}\mathbf{e}^*(k), \tag{4.63}$$

where μ is the convergence factor.

Note that we can use the SMQAP algorithm to derive the update equation of the SMQNLMS algorithm. In fact, the SMQNLMS does not require data-reusing as the SMQAP algorithm [9], thus by taking $L = 0$ and $\gamma(k) = \frac{\overline{\gamma}e(k)}{|e(k)|}$ we obtain the update equation of the SMQNLMS algorithm as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu(k)\|\mathbf{x}(k)\|^{-2}\mathbf{x}(k)e^*(k), \tag{4.64}$$

where $e(k)$ and $\mu(k)$ are defined as in (4.55) and (4.57), respectively.

Remark 6: By adopting the bound $\overline{\gamma} = 0$ in (4.64), the algorithm will reduce to the QNLMS algorithm with unity step size. Therefore, the QNLMS algorithm can be described as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu\|\mathbf{x}(k)\|^{-2}\mathbf{x}(k)e^*(k), \tag{4.65}$$

Table 4.1: COMPUTATIONAL COMPLEXITY PER UPDATE OF THE WEIGHT VECTOR

Algorithm	Real Multiplications	Real additions
QNLMS	$20N + 4$	$20N - 1$
QAP	$32L^3 + 16NL^2 + 16L^2$ $+19NL + 26L$	$32L^3 + 16NL^2 + 4L^2$ $+16NL + 8L$
TNLMS	$12N + 3$	$12N - 1$
TAP	$18L^3 + 9NL^2 + 9L^2$ $+11NL + 50L$	$18L^3 + 9NL^2$ $+9NL + 39L$

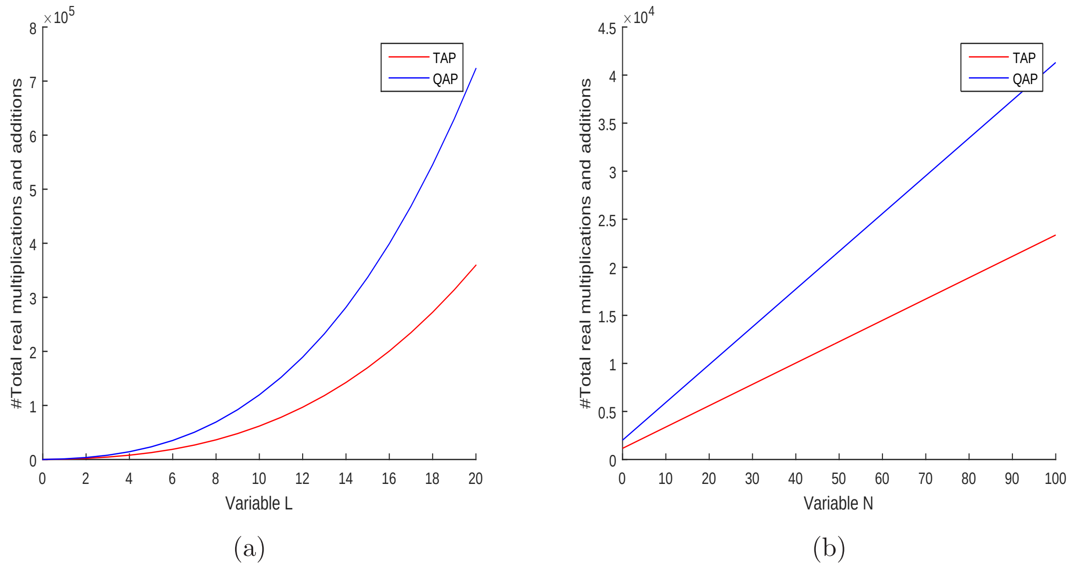


Figure 4.1: The numerical complexity of the TAP and the QAP algorithms for two cases: (a) $N = 15$, variable L; (b) $L = 3$, variable N.

where μ is the convergence factor.

The computational complexity for each update of the weight vector of the trinion based and quaternion based adaptive filtering algorithms are listed in Table 4.1. The filter length and the memory length are N and L, respectively. Also, Figures 4.1(a) and 4.1(b) show a comparison between the total number of real multiplications and additions required by the TAP and the QAP algorithms for two cases: $N = 15$, variable L and $L = 3$, variable N. As can be seen, the trinion model can efficiently decrease the computational complexity in comparison with the quaternion model, whenever the problem at hand suits both the quaternion and trinion solutions.

4.6 Application of quaternion-valued adaptive algorithms to adaptive beamforming

As an illustration for the use of quaternions, we can study its application to adaptive beamforming. By utilizing the crossed-dipole array and quaternions, we can decrease the number of engaged sensors in the adaptive beamforming process. Therefore, the computational complexity and the energy consumption of the system will reduce without losing the quality of the performance [3, 73–75].

A uniform linear array (ULA) is illustrated in Figure 4.2 [3, 57]. It contains M crossed-dipole pairs, they are placed on y-axis and the distance between neighboring antennas is d. At each position, the two crossed components are parallel to x-axis and y-axis, respectively. The direction of arrival (DOA) of a far-field incident signal is defined by the angles θ and ϕ. Assume that this signal impinges upon the array from the y-z plane. Thus, $\phi = \frac{\pi}{2}$ or $-\frac{\pi}{2}$, and $0 \leq \theta \leq \frac{\pi}{2}$. As a consequence, the spatial steering vector for this far-field incident signal is given by

$$\mathbf{s}_c(\theta, \phi) = [1, e^{-\jmath 2\pi d \sin\theta \sin\phi/\lambda}, \cdots, e^{-\jmath 2\pi (M-1) d \sin\theta \sin\phi/\lambda}]^T, \tag{4.66}$$

where λ stands for the wavelength of the incident signal. For a crossed-dipole the spatial-polarization coherent vector can be expressed by [76, 77]

$$\mathbf{s}_p(\theta, \phi, \gamma, \eta) = \begin{cases} [-\cos\gamma, \cos\theta \sin\gamma e^{\jmath\eta}] & \text{for } \phi = \frac{\pi}{2}, \\ [\cos\gamma, -\cos\theta \sin\gamma e^{\jmath\eta}] & \text{for } \phi = -\frac{\pi}{2}, \end{cases} \tag{4.67}$$

where $\gamma \in [0, \frac{\pi}{2}]$ and $\eta \in [-\pi, \pi]$ are the auxiliary polarization angle and the polarization phase difference, respectively.

We can divide the array structure into two sub-arrays so that one of them is parallel to the x-axis and the other one is parallel to the y-axis. Then the complex-valued steering vector parallel to the x-axis is presented as

$$\mathbf{s}_x(\theta, \phi, \gamma, \eta) = \begin{cases} -\cos\gamma \mathbf{s}_c(\theta, \phi) & \text{for } \phi = \frac{\pi}{2}, \\ \cos\gamma \mathbf{s}_c(\theta, \phi) & \text{for } \phi = -\frac{\pi}{2}, \end{cases} \tag{4.68}$$

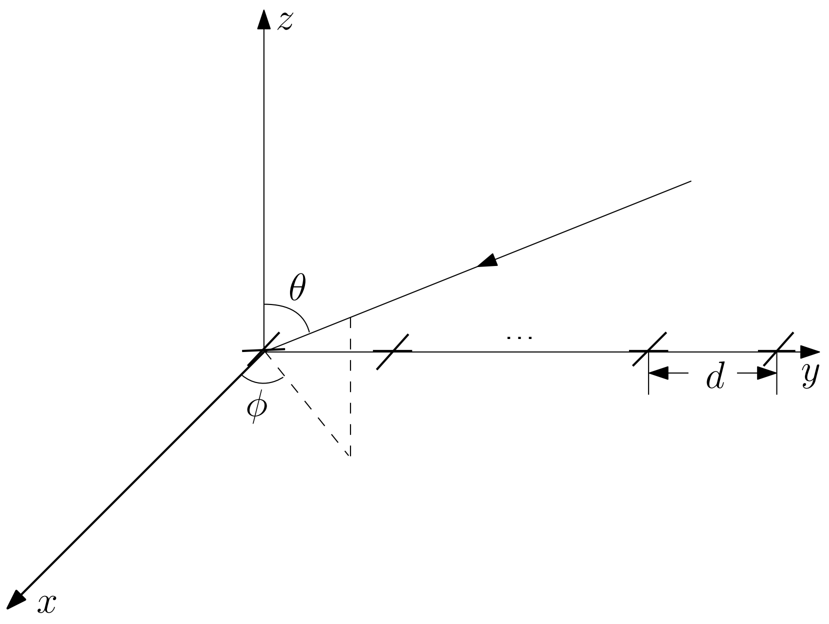


Figure 4.2: A ULA with crossed-dipole [3].

and the one parallel to the y-axis is given by

$$\mathbf{s}_y(\theta, \phi, \gamma, \eta) = \begin{cases} \cos\theta \sin\gamma e^{\jmath\eta}\mathbf{s}_c(\theta, \phi) & \text{for } \phi = \frac{\pi}{2}, \\ -\cos\theta \sin\gamma e^{\jmath\eta}\mathbf{s}_c(\theta, \phi) & \text{for } \phi = -\frac{\pi}{2}. \end{cases} \tag{4.69}$$

Using the Cayley-Dickson formula (4.6), we can combine $\mathbf{s}_x(\theta, \phi, \gamma, \eta)$ and $\mathbf{s}_y(\theta, \phi, \gamma, \eta)$ together, we obtain a quaternion-valued steering vector as follows

$$\mathbf{s}_q(\theta, \phi, \gamma, \eta) = \mathbf{s}_x(\theta, \phi, \gamma, \eta) + \imath\mathbf{s}_y(\theta, \phi, \gamma, \eta). \tag{4.70}$$

The response of the array for the quaternion-valued weight vector $\mathbf{w}$ is given as below

$$r(\theta, \phi, \gamma, \eta) = \mathbf{w}^H\mathbf{s}_q(\theta, \phi, \gamma, \eta). \tag{4.71}$$

In the case of reference signal based quaternion-valued adaptive beamforming, the reference signal $d(k)$ is available. Therefore, the response of the array is the quaternion-valued beamformer output and it is defined as $y(k) = \mathbf{w}^H(k)\mathbf{x}(k)$, where $\mathbf{x}(k)$ is the received quaternion-valued vector sensor signals and $\mathbf{w}(k)$ is the quaternion-valued weigh vector. Also, the quaternion-valued error signal can be defined as $e(k) = d(k) - y(k)$.

4.7 Simulations

In this section, we apply the proposed algorithms to two scenarios. Scenario 1 verifies the performance of the trinion based and the quaternion based algorithms when they are used to wind profile prediction. In Scenario 2, we implement quaternionic adaptive beamforming by quaternion-valued algorithms.

4.7.1 Scenario 1

In this scenario, all the proposed algorithms in this chapter are applied to anemometer readings provided by Google's RE<C Initiative [78]. The wind speed recorded on May 25, 2011, is utilized for the algorithms comparisons. The step size, μ, is selected to be 10^{-8} for the TLMS and the QLMS algorithms and 0.9 for the TNLMS, the TAP, the QNLMS, and the QAP algorithms, and $\overline{\gamma}$ is set to be 5. Also, the threshold bound vector $\boldsymbol{\gamma}(k)$ is selected as *simple choice constraint vector* [8] which is defined as $\gamma_0(k) = \frac{\overline{\gamma}e(k)}{|e(k)|}$ and $\gamma_i(k) = d(k-i) - \mathbf{w}^T(k)\mathbf{x}(k-i)$, for $i = 1, \cdots, L$. The filter length is 8, the memory length, L, and the prediction step are chosen equal to 1. All algorithms are initialized with zeros.

The predicted results provided by trinion and quaternion based algorithms are shown in Figures 4.3 and 4.4, respectively. The learning curves using the TNLMS, the SMTNLMS, the TAP, and the SMTAP algorithms are shown in Figures 4.5(a) and 4.5(b). Also, for comparison between the trinion and the quaternion based algorithms, the learning curves related to the TNLMS, the QNLMS, the TAP, and the QAP algorithms are depicted in Figures 4.6(a) and 4.6(b).

The average of implementation times and the number of updates performed by the trinion and the quaternion based algorithms are presented in Table 4.2. From the results, we can observe that all algorithms can track the wind data efficiently; however, the trinion based algorithms need a shorter time for implementation compared to their corresponding quaternion based algorithms. Also, we can observe that the set-membership based versions of the TNLMS, the QNLMS, the TAP, and the QAP algorithms have a low number of updates. Therefore, the set-membership algorithms can save energy effectively.

Moreover, we implemented the same scenario using a real-valued algorithm. Indeed, we used three affine projection (AP) algorithms whose parameters are chosen similar to the TAP algorithm to compare the tracking results between the AP and the TAP

Table 4.2: The Average of implementation times and the number of updates for the trinion and the quaternion based algorithms using MATLAB software

Algorithm	Time (second)	Update rate	Algorithm	Time (second)	Update rate
TLMS	2.45	100%	QLMS	7.2	100%
TNLMS	8	100%	QNLMS	9.4	100%
TAP	67	100%	QAP	142	100%
SMTNLMS	3.8	17.92%	SMQNLMS	9.2	17.87%
SMTAP	13	6.52%	SMQAP	20.1	6.34%

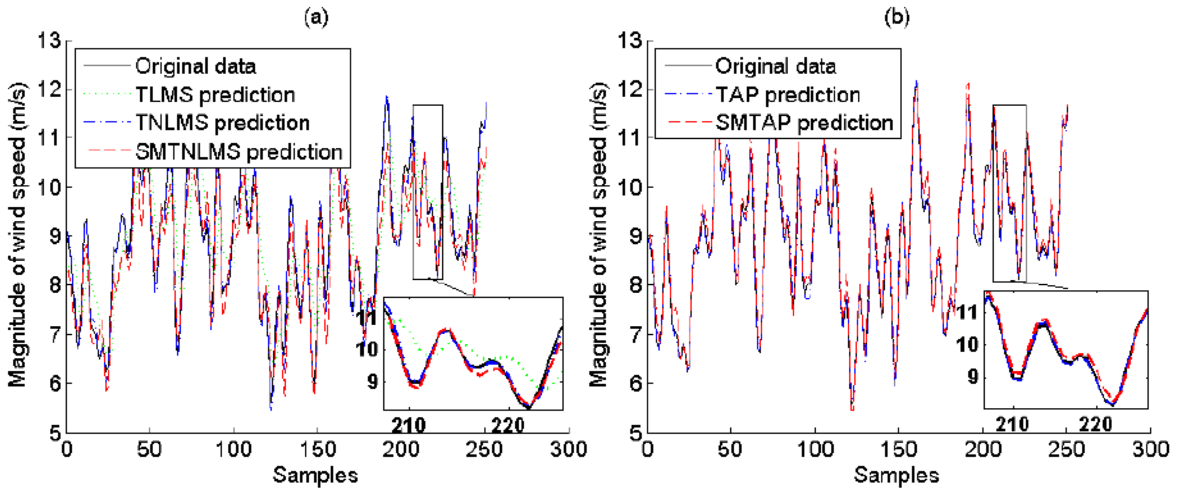


Figure 4.3: Predicted results from the trinion based algorithms.

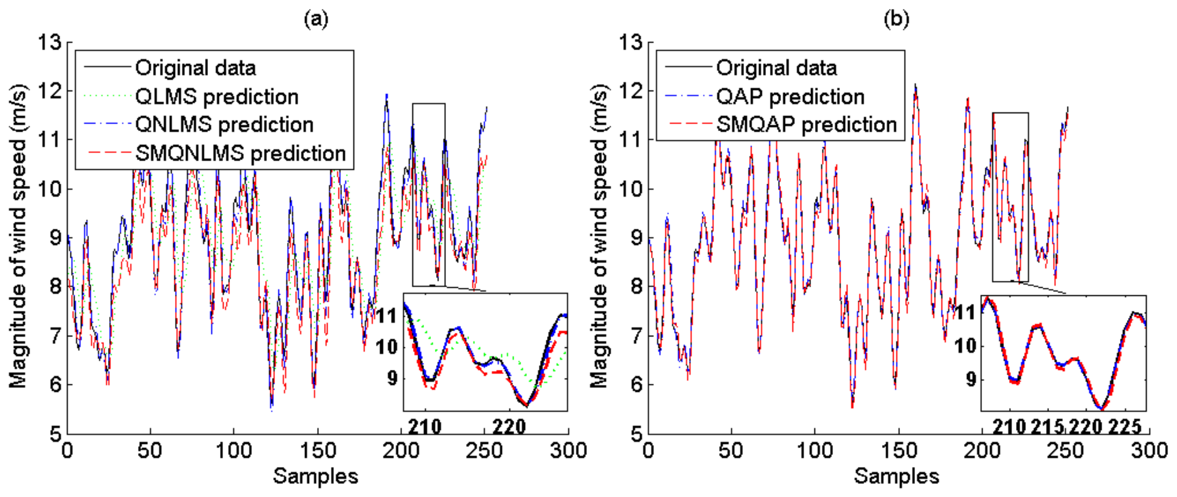
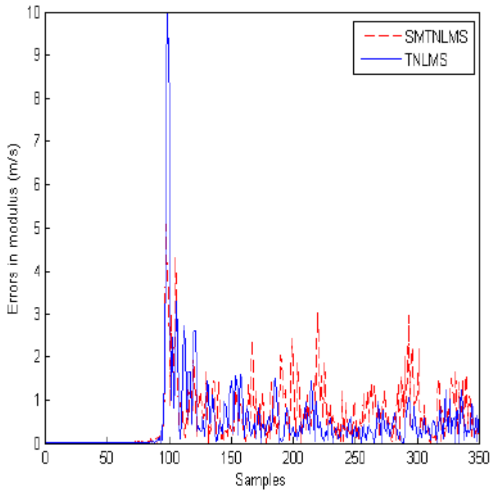
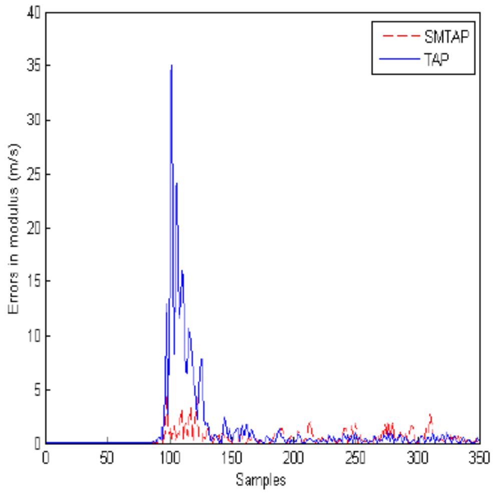


Figure 4.4: Predicted results from the quaternion based algorithms.

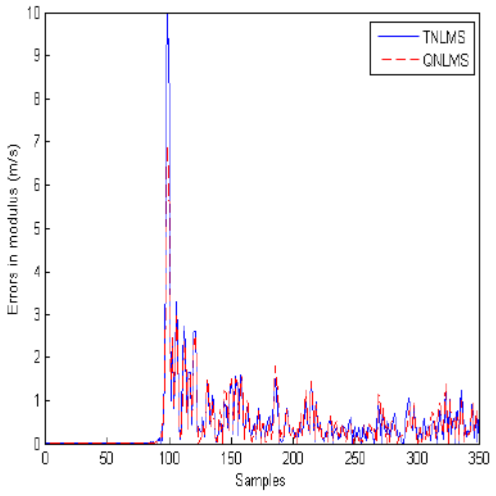


(a)

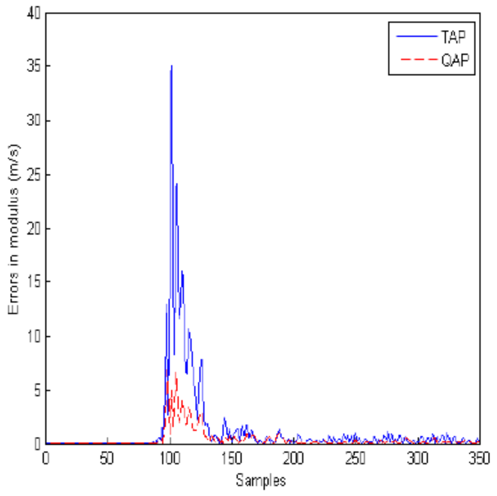


(b)

Figure 4.5: Learning curves of (a) the TNLMS and the SMTNLMS algorithms; (b) the TAP and the SMTAP algorithms.



(a)



(b)

Figure 4.6: Learning curves of (a) the TNLMS and the QNLMS algorithms; (b) the TAP and the QAP algorithms.

algorithms. We did not notify a significant difference between the tracking results of the AP and the TAP algorithms, thus we avoid presenting an additional figure since the results were similar to Figure 4.3(b). However, in wind profile prediction, It would be preferable to employ trinion-valued algorithms since there is some structure between the three components of data.

4.7.2 Scenario 2

In this scenario, we simulate the quaternionic adaptive beamforming [57] using the QLMS, the QNLMS, the SMQNLMS, the QAP, and the SMQAP algorithms. We assume a sensor array with 10 crossed-dipoles and half-wavelength spacing. The step size, μ, for the QLMS, the QNLMS, and the QAP algorithms are 4×10^{-5}, 0.009, and 0.005, respectively. For the QAP and the SMQAP algorithms, the memory length, L, is set to 1. A desired signal with 20 dB SNR ($\sigma_n^2 = 0.01$) impinges from broadside, $\theta = 0$ and $\phi = \frac{\pi}{2}$, and two interfering signals with signal-to-interference ratio (SIR) of -10 dB arrive from $(\theta, \phi) = (\frac{\pi}{9}, \frac{\pi}{2})$ and $(\theta, \phi) = (\frac{\pi}{6}, -\frac{\pi}{2})$, respectively. All the signals have the same polarization of $(\gamma, \eta) = (0, 0)$. $\overline{\gamma}$ is set to be $\sqrt{2\sigma_n^2}$, and the vector $\boldsymbol{\gamma}(k)$ is selected as simple choice constraint vector defined in Scenario 1.

The learning curves of quaternion algorithms over 100 trials are shown in Figure 4.7. The average number of updates performed by the SMQNLMS and the SMQAP algorithms are 1408 and 1815 in a total of 10000 iterations (about 14.08% and 18.15%), respectively. As can be seen, the set-membership quaternion algorithms converge faster while having a lower number of updates. Also, the convergence rate of the QAP algorithm is higher than the SMQNLMS algorithm.

The response of a beamformer to the impinging signals as a function of θ is called beam pattern and is defined as $B(\theta) = \mathbf{w}^H \mathbf{s}(\theta)$, where $\mathbf{s}(\theta)$ is the steering vector. The magnitude of beam pattern explains the variation of a beamformer concerning the signal arriving from different DOA angles. Figure 4.8 illustrates the magnitude of beam pattern of the quaternion algorithms with $\theta = 0$. In this figure, the positive values of θ show the value range $\theta \in [0, \frac{\pi}{2}]$ for $\phi = \frac{\pi}{2}$ and the negative values, $\theta \in [-\frac{\pi}{2}, 0]$, indicate the same range of $\theta \in [0, \frac{\pi}{2}]$ but $\phi = -\frac{\pi}{2}$. We can observe that all the quaternion algorithms attained an acceptable beamforming result since the two nulls at the directions of the interfering signals are clearly visible.

The output signal to desired plus noise ratio (OSDR) and the output signal to interference plus noise ratio (OSIR) for the quaternion algorithms are presented in Table 4.3. The OSDR is achieved by calculating the power of the output signal and the total power of desired plus one-third of the noise signal, then we compute the ratio between these two values. Also, the OSIR is obtained by computing the power of the output signal and the total power of interference plus one-third of the noise signal, then we find the ratio between the two. As can be seen, the best results are obtained by the SMQNLMS and the SMQAP algorithms.

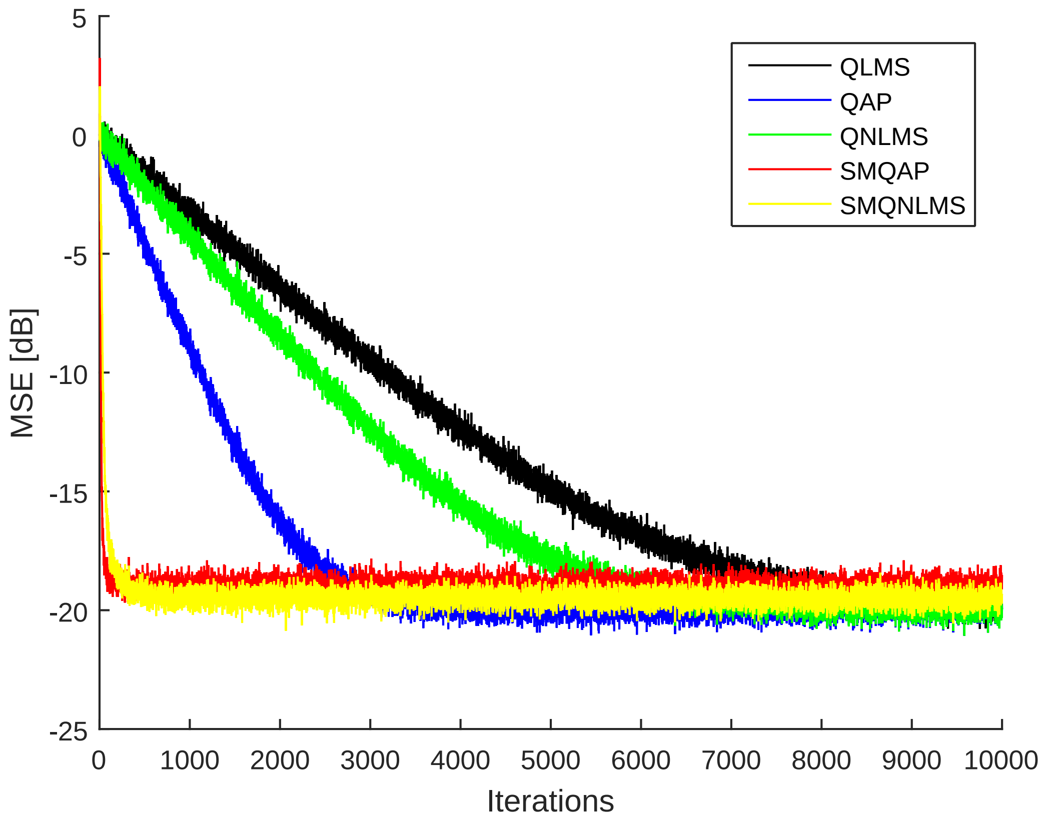


Figure 4.7: Learning curves of the QLMS, the QNLMS, the QAP, the SMQNLMS, and the SMQAP algorithms.

Table 4.3: The OSDR and the OSIR for the quaternion algorithms

Algorithms	QLMS	QNLMS	QAP	SMQNLMS	SMQAP
OSDR (dB)	-1.645	-1.502	-0.647	-0.024	0.004
OSIR (dB)	-11.699	-11.557	-10.701	-10.079	-10.050

4.8 Conclusions

In this chapter, we have generalized the set-membership model for the trinion and the quaternion number systems. First, we have reviewed some properties of the quaternion and the trinion systems. Then we have derived the set-membership trinion based algorithms and, by the same argument, the quaternion based adaptive filtering algorithms have been introduced. Also, we have presented the counterparts of the proposed algorithms without employing the set-membership approach. Moreover, we have reviewed

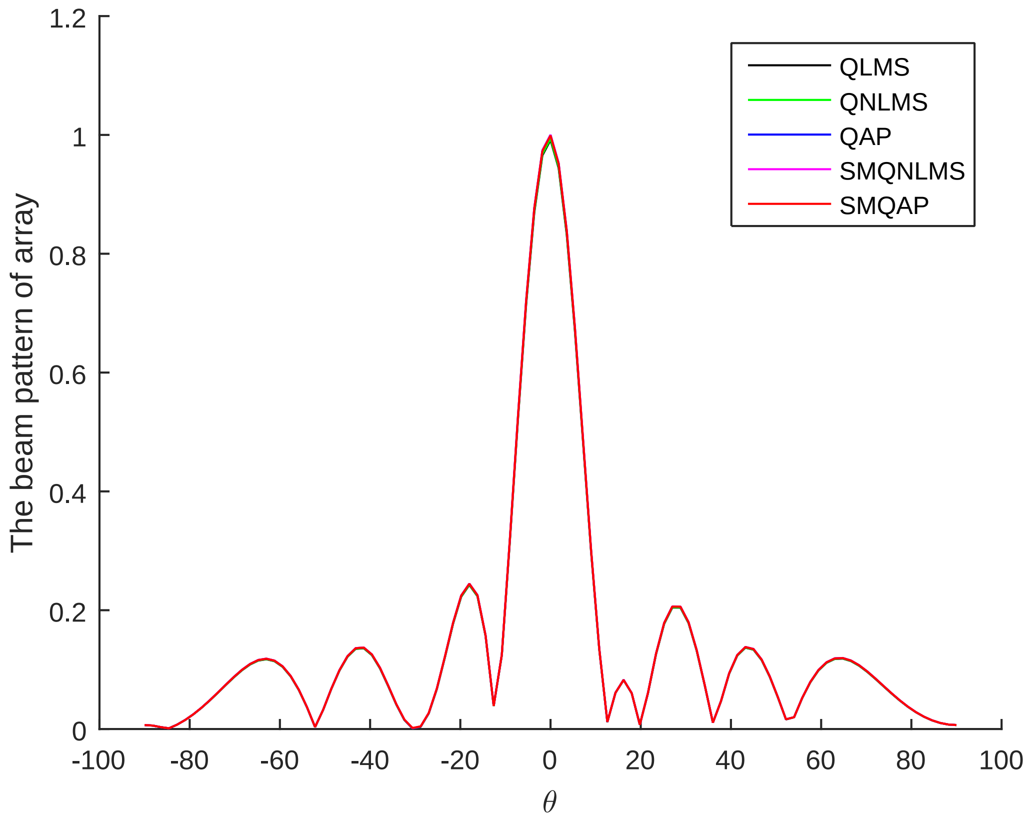


Figure 4.8: Beam patterns of the QLMS, the QNLMS, the QAP, the SMQNLMS, and the SMQAP algorithms when DOA of desired signal is $(\theta, \phi) = (0, \frac{\pi}{2})$.

the application of quaternion algorithms to adaptive beamforming. Numerical simulations for the recorded wind data and the adaptive beamforming have proven that the set-membership based algorithms have significantly lower update rates, while the penalty to be paid for that is not noteworthy. Also, we have observed that the trinion based algorithms have comparable performance to the quaternion based ones, however with striking lower computational complexity.

Chapter 5

Improved Set-Membership Partial-Update Affine Projection Algorithm

Adaptive filters have applications in a wide range of areas such as noise cancellation, signal prediction, echo cancellation, communications, radar, and speech processing. In several applications, a large number of coefficients to be updated leads to high computational complexity, turning the adaptation of the filter coefficients prohibitive regarding hardware requirements. In some cases, like acoustic echo cancellation, the adaptive filter might use a few thousand coefficients in order to model the underlying physical system with sufficient accuracy. In these applications, the convergence would entail a large number of iterations, calling for a more sophisticated updating rule which is inherently more computationally intensive. For a given adaptive filter, the computational complexity can be reduced by updating only part of the filter coefficients at each iteration, forming a family of algorithms called partial-update (PU) algorithms. In the literature, several variants of adaptive filtering algorithms with partial-update have been proposed [2, 79–91].

Another powerful approach to decrease the computational complexity of an adaptive filter is to employ set-membership filtering (SMF) approach [2, 9]. Algorithms developed from the SMF framework employ a deterministic objective function related to a bounded error constraint on the filter output, such that the updates belong to a set of feasible solutions. Implementation of SMF algorithms involves two main steps: 1) information evaluation, 2) parameter update. As compared with the standard normalized least mean square (NLMS) and affine projection (AP) algorithms, the set-membership normalized least mean square (SM-NLMS) and the set-membership affine projection (SM-AP) algorithms lead to reduced computational complexity chiefly due to data-selective updates [9, 30, 31, 43, 47, 92–94].

The use of PU strategy decreases the computational complexity while reducing convergence speed. We employ SMF technique to reduce further the computational load due to a lower number of updates. However applying the SMF and PU strategies together might result in slow convergence speed. One approach to accelerate the convergence speed is choosing a smaller error estimation bound, but it might increase the number of updates. Also, if we adopt a higher error estimation threshold to reduce the number of updates, the convergence rate will decrease. Therefore, convergence speed and computational complexity are conflicting requirements.

In this chapter, we introduce an interesting algorithm which can accelerate the convergence speed and simultaneously reduce the number of updates (and as a result decrease the computational complexity) in the set-membership partial-update affine projection (SM-PUAP) algorithm. In the SM-PUAP algorithm, some updates move too far from their SM-AP update; especially when the angle between the updating direction and the threshold hyperplane is small. In this case, we might have a significant disturbance in the coefficient update while attempting to reach the feasibility set. Therefore, to limit the distance between two consecutive updates, first, we will construct a hypersphere centered at the present weight vector whose radius equals the distance between the current weight vector and the weight vector that would be obtained with the SM-AP algorithm. This radius is an upper bound on the Euclidean norm of the coefficient disturbance that is allowed in the proposed improved set-membership partial-update affine projection (I-SM-PUAP) algorithm.

The content of this chapter was published in [95]. In this chapter, first of all, we review the SM-PUAP algorithm in Section 5.1. Then, in Section 5.2, we derive the I-SM-PUAP algorithm. Section 5.3 presents simulations of the algorithms. Finally, Section 5.4 contains the conclusions.

5.1 Set-Membership Partial-Update Affine Projection Algorithm

In this section, we present the SM-PUAP algorithm [2]. The main objective of the partial-update adaptation is to perform updates in M out of $N+1$ adaptive filter coefficients, where N is the order of the adaptive filter. The M coefficients to be updated at time instant k are specified by an index set $\mathcal{I}_M(k) = \{i_0(k), \cdots, i_{M-1}(k)\}$ with $\{i_j(k)\}_{j=0}^{M-1}$ chosen from the set $\{0, \cdots, N\}$. The subset of coefficients with indices in $\mathcal{I}_M(k)$ plays an essential role in the performance and the effectiveness of the partial-update strategy. Note that $\mathcal{I}_M(k)$ varies with the time instant k. As a result, the M coefficients to be updated can change according to the time instant. The choice of which M coefficients should be updated is related to the optimization criterion chosen for algorithm derivation. The SM-PUAP algorithm [2] takes the update vector $\mathbf{w}(k+1)$ as the vector minimizing the Euclidean distance $\|\mathbf{w}(k+1) - \mathbf{w}(k)\|^2$ subject to the constraint $\mathbf{w}(k+1) \in \mathcal{H}(k)$ in such a way that only M coefficients are updated.

The optimization criterion in the SM-PUAP algorithm is described as follows. Let $\psi^{L+1}(k)$ indicate the intersection of the last $L+1$ constraint sets. A coefficient update is implemented whenever $\mathbf{w}(k) \notin \psi^{L+1}(k)$ as follows

$$\begin{aligned}
&\min \|\mathbf{w}(k+1) - \mathbf{w}(k)\|^2 \\
&\text{subject to :} \\
&\mathbf{d}(k) - \mathbf{X}^T(k)\mathbf{w}(k+1) = \boldsymbol{\gamma}(k) \\
&\tilde{\mathbf{C}}_{\mathcal{I}_M(k)}[\mathbf{w}(k+1) - \mathbf{w}(k)] = 0
\end{aligned} \tag{5.1}$$

where

$\mathbf{d}(k) \in \mathbb{R}^{(L+1)\times 1}$	contains the desired output from the $L+1$ last time instants;
$\boldsymbol{\gamma}(k) \in \mathbb{R}^{(L+1)\times 1}$	specifies the point in $\psi^{L+1}(k)$;
$\mathbf{X}(k) \in \mathbb{R}^{(N+1)\times(L+1)}$	contains the corresponding input vectors, i.e.,

$$\begin{aligned}
\mathbf{d}(k) &= [d(k)\ d(k-1)\ \cdots\ d(k-L)]^T, \\
\boldsymbol{\gamma}(k) &= [\gamma_0(k)\ \gamma_1(k)\ \cdots\ \gamma_L(k)]^T, \\
\mathbf{X}(k) &= [\mathbf{x}(k)\ \mathbf{x}(k-1)\ \cdots\ \mathbf{x}(k-L)],
\end{aligned} \tag{5.2}$$

with $\mathbf{x}(k)$ being the input-signal vector

$$\mathbf{x}(k) = [x(k)\ x(k-1)\ \cdots\ x(k-N)]^T. \tag{5.3}$$

Moreover, the matrix $\tilde{\mathbf{C}}_{\mathcal{I}_M(k)} = \mathbf{I} - \mathbf{C}_{\mathcal{I}_M(k)}$ is a complementary matrix that gives $\tilde{\mathbf{C}}_{\mathcal{I}_M(k)}\mathbf{w}(k+1) = \tilde{\mathbf{C}}_{\mathcal{I}_M(k)}\mathbf{w}(k)$, which means that only M coefficients are updated. The threshold vector elements are such that $|\gamma_i(k)| \leq \overline{\gamma}$, for $i = 0, \cdots, L$. The matrix $\mathbf{C}_{\mathcal{I}_M(k)}$ is a diagonal matrix that identifies the coefficients to be updated at instant k, if an update is required. This matrix has M nonzero elements equal to one located at positions declared by $\mathcal{I}_M(k)$.

Using the method of Lagrange multipliers we obtain the following updating rule

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{C}_{\mathcal{I}_M(k)}\mathbf{X}(k)[\mathbf{X}^T(k)\mathbf{C}_{\mathcal{I}_M(k)}\mathbf{X}(k)]^{-1}[\mathbf{e}(k) - \boldsymbol{\gamma}(k)] \tag{5.4}$$

The updating equation of the SM-PUAP algorithm is given by

$$\mathbf{w}(k+1) = \begin{cases} \mathbf{w}(k) + \mathbf{C}_{\mathcal{I}_M(k)}\mathbf{X}(k)\mathbf{P}(k)(\mathbf{e}(k) - \boldsymbol{\gamma}(k)) & \text{if } |e(k)| > \overline{\gamma} \\ \mathbf{w}(k) & \text{otherwise} \end{cases}, \tag{5.5}$$

where

$$\mathbf{P}(k) = (\mathbf{X}^T(k)\mathbf{C}_{\mathcal{I}_M(k)}\mathbf{X}(k) + \delta\mathbf{I})^{-1}, \tag{5.6}$$

$$\mathbf{e}(k) = [e(k)\ \epsilon(k-1)\ \cdots\ \epsilon(k-L)]^T, \tag{5.7}$$

with $e(k) = d(k) - \mathbf{w}^T(k)\mathbf{x}(k)$, and $\epsilon(k-i) = d(k-i) - \mathbf{w}^T(k)\mathbf{x}(k-i)$ for $i = 1, \cdots, L$. In the Equation (5.6), δ and $\mathbf{I}$ are a small positive constant and an $(L+1) \times (L+1)$ identity matrix, respectively. The diagonal matrix $\delta\mathbf{I}$ is added to the matrix to be inverted in order to avoid numerical problems in the inversion operation in the cases $\mathbf{X}^T(k)\mathbf{C}_{\mathcal{I}_M(k)}\mathbf{X}(k)$ is ill conditioned.

A natural choice for the M nonzero diagonal elements of $\mathbf{C}_{\mathcal{I}_M(k)}$ is those corresponding to the coefficients of $\mathbf{w}(k)$ with the most significant norms. In fact, by this selection, the M coefficients with the largest norms will be updated, and the rest of the parameters will remain unchanged.

Figure 5.1 illustrates a possible update in SM-PUAP algorithm in $\mathbb{R}^3$ for $L = 0$. As can be seen, $\mathbf{w}(k+1)$ is far from the $\mathbf{w}_{\text{SM-AP}}(k)$, and it will reduce the convergence rate of the SM-PUAP algorithm. In the next section, we will address this issue by

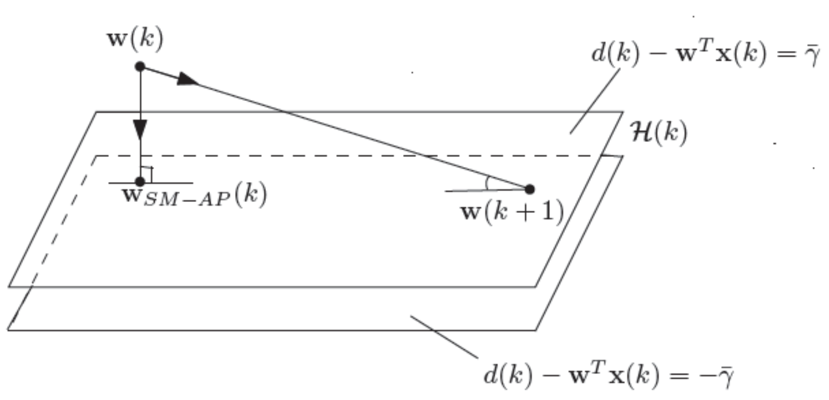

$\mathbf{w}(k)$
$d(k) - \mathbf{w}^T\mathbf{x}(k) = \bar{\gamma}$
$\mathcal{H}(k)$
$\mathbf{w}_{SM-AP}(k)$
$\mathbf{w}(k+1)$
$d(k) - \mathbf{w}^T\mathbf{x}(k) = -\bar{\gamma}$

Figure 5.1: Update in SM-PUAP algorithm in $\mathbb{R}^3$ for $L = 0$.

presenting the I-SM-PUAP algorithm.

5.2 Improved Set-membership Partial-Update Affine Projection Algorithm

In this section, we propose the I-SM-PUAP algorithm aiming at accelerating the convergence speed of SM-PUAP algorithm and decreasing the number of updates.

Since the partial update strategy deviates the updating direction from the one determined by the input signal vector $\mathbf{x}(k)$ utilized by the SM-PUAP algorithm, it is natural that the size of the step for a partial update algorithm should be smaller than the corresponding algorithm that updates all coefficients. A solution to this problem is to constrain the Euclidean norm of the coefficient disturbance of the partial update algorithm to the disturbance implemented by the originating nonpartial updating algorithm, in our case the SM-AP algorithm. For that, we build hypersphere, $S(k)$, whose radius is the distance between $\mathbf{w}(k)$ and the SM-AP update. The SM-AP update takes a step towards the hyperplanes $d(k) - \mathbf{w}^T\mathbf{x}(k) = \pm\overline{\gamma}$ with the minimum disturbance, i.e., when the step in the direction $\mathbf{x}(k)$ touches the hyperplane perpendicularly. Therefore, the radius of the hypersphere $S(k)$ is given by

$$\mu(k) = \min\Big(\frac{|\mathbf{w}^T(k)\mathbf{x}(k) - d(k) \pm \overline{\gamma}|}{\|\mathbf{x}(k)\|_2}\Big), \tag{5.8}$$

where $\|\cdot\|_2$ is the Euclidean norm in $\mathbb{R}^{N+1}$. The equation describing the hypersphere

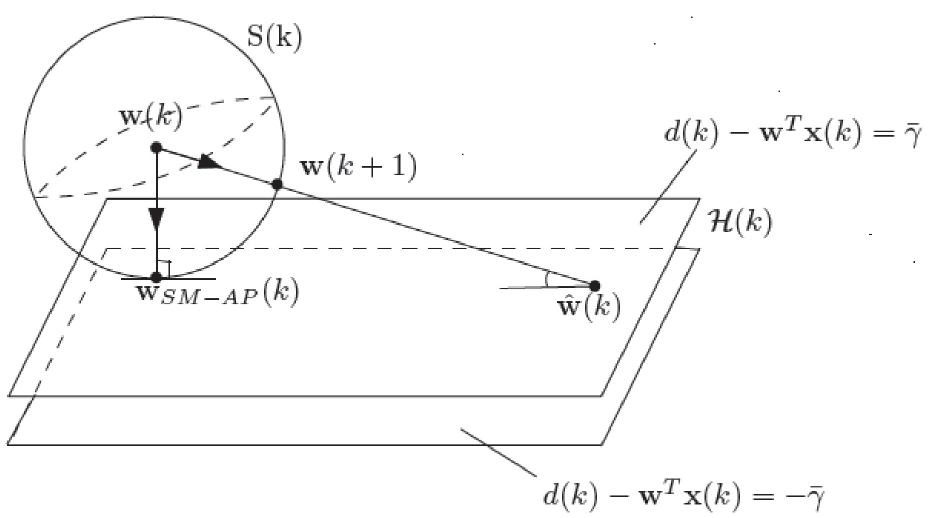

S(k)
$\mathbf{w}(k)$
$\mathbf{w}(k+1)$
$d(k) - \mathbf{w}^T\mathbf{x}(k) = \bar{\gamma}$
$\mathcal{H}(k)$
$\mathbf{w}_{SM-AP}(k)$
$\hat{\mathbf{w}}(k)$
$d(k) - \mathbf{w}^T\mathbf{x}(k) = -\bar{\gamma}$

Figure 5.2: Update in I-SM-PUAP algorithm in $\mathbb{R}^3$ for $L = 0$.

$S(k)$ with the radius $\mu(k)$ and centered at $\mathbf{w}(k)$ is as follows

$$(w_0 - w_0(k))^2 + \cdots + (w_N - w_N(k))^2 = \mu^2(k). \tag{5.9}$$

As can be observed in Figure 5.1, $\mathbf{w}(k+1)$ is the point where, starting from $\mathbf{w}(k)$, the vector representing the $\mathbf{w}(k+1)$ direction touches the hyperplane $d(k) - \mathbf{w}^T\mathbf{x}(k) = \overline{\gamma}$. Unlike the SM-PUAP algorithm, in the I-SM-PUAP algorithm $\mathbf{w}(k+1)$ is the point where, starting from $\mathbf{w}(k)$, the vector representing the partial direction touches the defined N dimensional hypersphere $S(k)$ and points at a sparse version of $\mathbf{x}(k)$. A visual interpretation of the I-SM-PUAP algorithm is described in Figure 5.2.

Define $\hat{\mathbf{w}}(k)$ as the update result of Equation (5.5) with $\boldsymbol{\gamma}(k) = [0 \ \cdots \ 0]^T$. In order to find the update of $\mathbf{w}(k)$ to the boundary of hypersphere $S(k)$ such that $\tilde{\mathbf{C}}_{\mathcal{I}_M(k)}\mathbf{w}(k+1) = \tilde{\mathbf{C}}_{\mathcal{I}_M(k)}\mathbf{w}(k)$ we have to find the intersection of the hypersphere $S(k)$ with the line $l(k)$ passing through $\mathbf{w}(k)$ and $\hat{\mathbf{w}}(k)$. This line is parallel to the vector $\mathbf{u}(k) = \frac{\mathbf{a}(k)}{\|\mathbf{a}(k)\|_2}$, where $\mathbf{a}(k) = [\hat{w}_0(k) - w_0(k) \ \cdots \ \hat{w}_N(k) - w_N(k)]^T$. Hence, the equation of the line $l(k)$ is given as follows

$$\begin{cases} \frac{w_0 - w_0(k)}{u_0(k)} = \cdots = \frac{w_i - w_i(k)}{u_i(k)} = \cdots = \frac{w_N - w_N(k)}{u_N(k)}, & \text{for } i \in \mathcal{I}_M(k), \\ w_i = w_i(k), & \text{for } i \notin \mathcal{I}_M(k). \end{cases} \tag{5.10}$$

In order to find the intersection of the line $l(k)$ with the hypersphere $S(k)$, we should replace Equation (5.10) in Equation (5.9). Thus, we will attain $w_i = w_i(k)$ for

$i \notin \mathcal{I}_M(k)$, and for $i \in \mathcal{I}_M(k)$ we have

$$\frac{u_0^2(k)}{u_i^2(k)}(w_i - w_i(k))^2 + \cdots + (w_i - w_i(k))^2 + \cdots + \frac{u_N^2(k)}{u_i^2(k)}(w_i - w_i(k))^2 = \mu^2(k). \tag{5.11}$$

Then,

$$(w_i - w_i(k))^2 = u_i^2(k)\mu^2(k), \tag{5.12}$$

where we obtained the last equality owing to $\|\mathbf{u}(k)\|_2 = 1$. Therefore, the intersections of the line $l(k)$ and the hypersphere $S(k)$ are given by

$$w_i = w_i(k) \pm u_i(k)\mu(k). \tag{5.13}$$

We will choose the positive sign in Equation (5.13) since the direction of the vector $\mathbf{a}(k)$ is from $\mathbf{w}(k)$ to $\hat{\mathbf{w}}(k)$. As a result, vector $\mathbf{w}(k+1)$ becomes as below

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu(k)\mathbf{u}(k). \tag{5.14}$$

Also, as an alternative method, we can get $\mathbf{w}(k+1)$ through an elegant geometrical view. Denote $\mathbf{w}(k+1)$ in Equation (5.5) as $\hat{\mathbf{w}}(k)$ while taking $\boldsymbol{\gamma}(k) = [0 \ \cdots \ 0]^T$. Define $\mathbf{a}(k)$ as

$$\mathbf{a}(k) = \hat{\mathbf{w}}(k) - \mathbf{w}(k) = \mathbf{C}_{\mathcal{I}_M(k)}\mathbf{X}(k)\mathbf{P}(k)\mathbf{e}(k). \tag{5.15}$$

If we take the step size equal to $\|\mathbf{a}(k)\|_2$ and do the update in the direction of $\frac{\mathbf{a}(k)}{\|\mathbf{a}(k)\|_2}$, then the parameters will reach $\hat{\mathbf{w}}(k)$. However, our objective is to reach the boundary of hypersphere $S(k)$ centered at $\mathbf{w}(k)$ with radius $\mu(k)$ in the direction of $\frac{\mathbf{a}(k)}{\|\mathbf{a}(k)\|_2}$, thus the step size must be equal to the radius of $S(k)$ so that the update equation becomes

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu(k)\frac{\mathbf{a}(k)}{\|\mathbf{a}(k)\|_2} = \mathbf{w}(k) + \mu(k)\mathbf{u}(k). \tag{5.16}$$

Table 5.1 summarizes the I-SM-PUAP algorithm.

Table 5.1: Improved Set-Membership Partial-Update Affine Projection(I-SM-PUAP) Algorithm

```
I-SM-PUAP Algorithm

Initialization
$\mathbf{x}(-1) = \mathbf{w}(0) = [0 \ \cdots \ 0]^T$
$\delta =$ small positive constant
choose $\overline{\gamma}$
Do for $k \geq 0$
  $\mathbf{e}(k) = \mathbf{d}(k) - \mathbf{X}^T(k)\mathbf{w}(k)$
  if $|e(k)| > \overline{\gamma}$
    $\mu(k) = \min\left(\frac{|-e(k) \pm \overline{\gamma}|}{\|\mathbf{x}(k)\|_2}\right)$
    $\mathbf{a}(k) = \mathbf{C}_{\mathcal{I}_M(k)}\mathbf{X}(k)[\mathbf{X}^T(k)\mathbf{C}_{\mathcal{I}_M(k)}\mathbf{X}(k) + \delta\mathbf{I}]^{-1}\mathbf{e}(k)$
    $\mathbf{w}(k+1) = \mathbf{w}(k) + \frac{\mu(k)}{\|\mathbf{a}(k)\|_2}\mathbf{a}(k)$
  else
    $\mathbf{w}(k+1) = \mathbf{w}(k)$
  end
end
```

5.3 Simulations

5.3.1 Scenario 1

In this section, the SM-PUAP algorithm [2] and the proposed I-SM-PUAP algorithm are applied to a system identification problem. The unknown system has order $N = 79$ and its coefficients are random scalars drawn from the standard normal distribution. The input signal is a binary phase-shift keying (BPSK) signal with $\sigma_x^2 = 1$. The signal-to-noise ratio (SNR) is set to 20 dB, i.e., $\sigma_n^2 = 0.01$. The bound on the output estimation error is chosen as $\overline{\gamma} = \sqrt{25\sigma_n^2}$. Also, we adopt the threshold bound vector $\boldsymbol{\gamma}(k)$ as $\gamma_0(k) = \frac{\overline{\gamma}e(k)}{|e(k)|}$ and $\gamma_i(k) = d(k-i) - \mathbf{w}^T(k)\mathbf{x}(k-i)$, for $i = 1, \cdots, L$ [2, 24]. The regularization constant, δ, is 10^{-12} and $\mathbf{w}(0) = [1 \ \cdots \ 1]^T$ which is not close to the unknown system. All learning curves averaged over 200 trials. We are updating 50 percent of the components randomly chosen of the filter to illustrate the partial updating, i.e., half of the elements of $\mathcal{I}_M(k)$ are nonzero at each time instant k.

Figure 5.3 shows the learning curves for the I-SM-PUAP algorithm with $L = 1, 4$, and it illustrates the learning curves for the SM-PUAP algorithm with $L = 64$ and 69. Also, in Figure 5.3 a blue curve is depicted using correlated inputs and $L = 1$. In

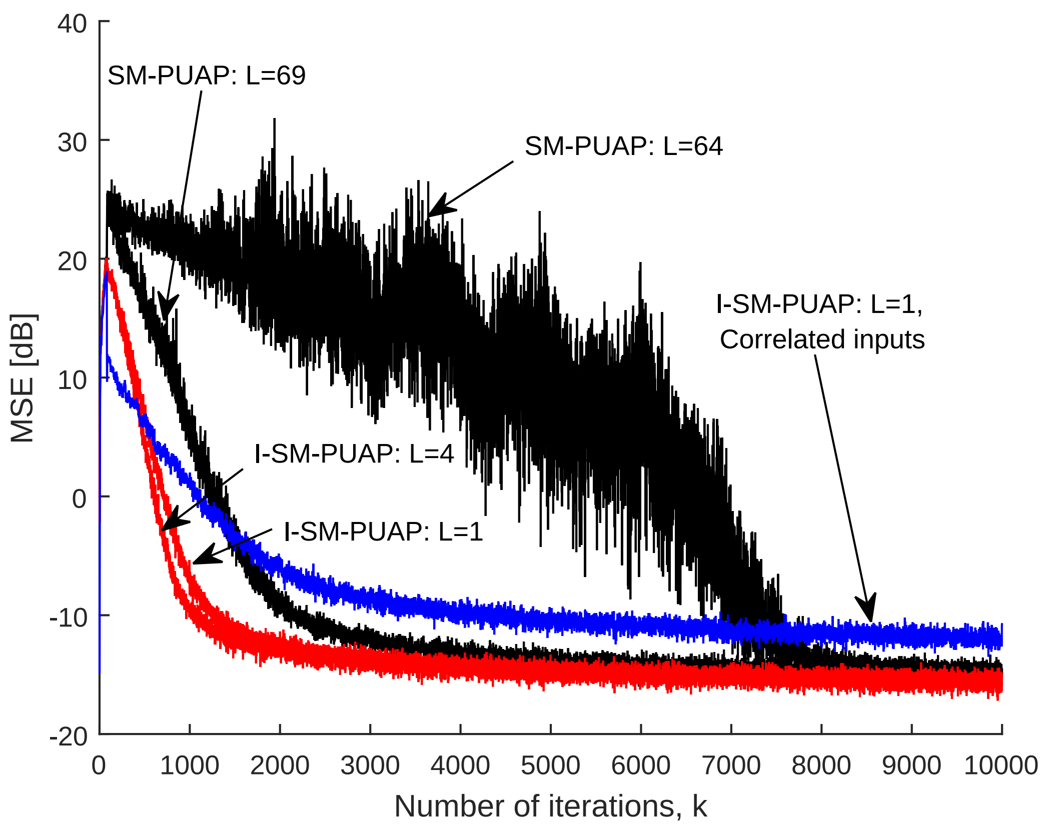


Figure 5.3: Learning curves of the I-SM-PUAP and the SM-PUAP algorithms applied on system identification problem.

fact, for the blue curve all of the specifications of the system are the same as explained above and the only difference is the input signal. The correlated input signal is chosen as $x(k) = 0.95x(k-1) + 0.19x(k-2) + 0.09x(k-3) - 0.5x(k-1) + m(k-4)$, where $m(k)$ is a zero-mean Gaussian noise with unit variance.

The average number of updates performed by the I-SM-PUAP algorithm are 8.3% and 6.5% for $L = 1$ and 4, respectively, and 20% in the case of the correlated input signal. The average number of updates implemented by the SM-PUAP algorithm are 14% and 25% for $L = 69$ and 64, respectively. Note that in both algorithms we have to find the inverse of an $(L+1) \times (L+1)$ matrix, thus large L implies high computational complexity. Therefore, the I-SM-PUAP algorithm requires lower implementation time since it presents fast convergence even for a small value of L. Also, it is worth mentioning that for $L < 64$ the SM-PUAP algorithm does not reach its steady-state in 10000 iterations. From the results, we can observe that the proposed algorithm, I-SM-PUAP, has faster convergence speed and lower number of updates as compared to the SM-PUAP algorithm.

5.3.2 Scenario 2

In this section, we perform the equalization of a channel with the following impulse response

$$\mathbf{h} = [1\ 2\ 3\ 4\ 4\ 3\ 2\ 1]^T. \tag{5.17}$$

We use a known training signal that consists of independent binary samples $(-1, 1)$ and an additional Gaussian white noise with variance 0.01 is present at the channel output. The I-SM-PUAP and the SM-PUAP algorithms are applied to find the impulse response of an equalizer of order 80. The delay in the reference signal is selected as 45. The parameters $\overline{\gamma}$ and $\boldsymbol{\gamma}(k)$ are chosen as $\sqrt{25\sigma_n^2}$ and the simple choice constraint vector is utilized as Scenario 1, respectively. The regularization constant, δ, is 10^{-12} and $\mathbf{w}(0) = [1\ \cdots\ 1]^T$. All learning curves are averaged over 100 trials. At each iteration, half of the elements of $\mathcal{I}_M(k)$ are set nonzero randomly. The memory-length, L, is 3.

Figure 5.4(a) shows the learning curves for the I-SM-PUAP and the SM-PUAP algorithms. The convolution of the equalizer impulse response at a given iteration after convergence with the channel impulse response is shown in Figure 5.4(b). The average number of updates implemented by the I-SM-PUAP and the SM-PUAP algorithms are 61% and 82%, respectively. As can be seen, the I-SM-PUAP algorithm has lower MSE and lower number of updates compared to the SM-PUAP algorithm.

5.4 Conclusions

In this chapter, we have introduced the improved set-membership partial-update affine projection (I-SM-PUAP) algorithm aiming at accelerating the convergence rate of the set-membership partial-update affine projection (SM-PUAP) algorithm, with lower computational complexity and reduced number of updates. To achieve this goal, we use the distance between the present weight vector and the one obtained with the SM-AP update, in order to provide a hypersphere that upper bounds the coefficient disturbance. Numerical simulations for the system identification and the channel

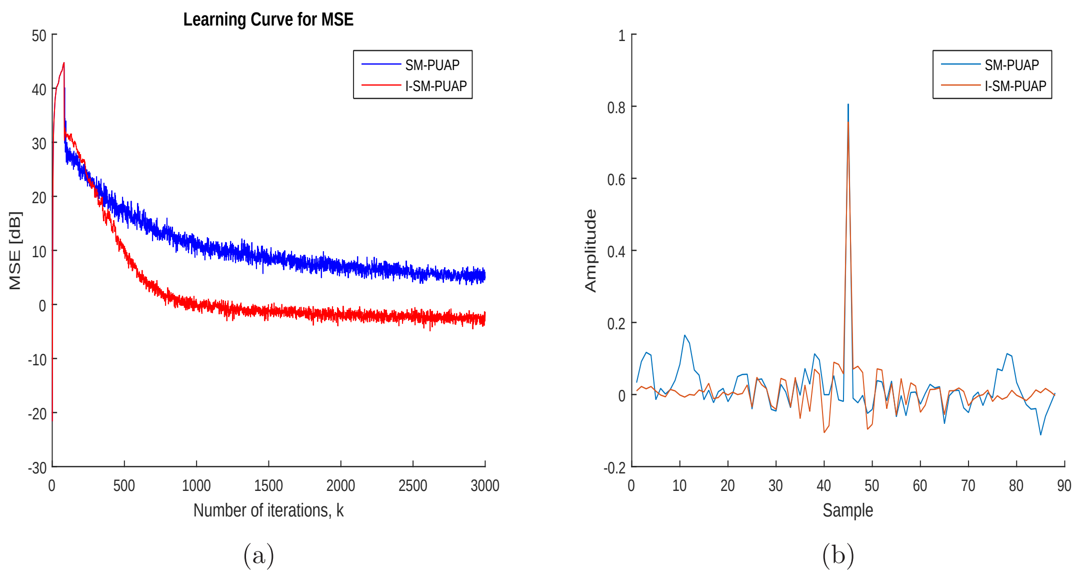


Figure 5.4: (a) Learning curves of the I-SM-PUAP and the SM-PUAP algorithms performing the equalization of a channel; (b) convolution results.

equalization problems have confirmed that the I-SM-PUAP algorithm has not only faster convergence rate, but also it requires a lower number of updates as compared to the SM-PUAP algorithm.

Chapter 6

Adaptive Filtering Algorithms for Sparse System Modeling

Adaptive filtering applied to signals originating from time-varying systems find applications in a wide diversity of areas such as communications, control, radar, acoustics, and speech processing. Nowadays, it is well known that many types of signal or system parameters admit sparse representation in a certain domain. However, classical adaptive algorithms such as the least-mean-square (LMS), the normalized LMS (NLMS), the affine projection (AP), and the recursive least-squares (RLS) do not take into consideration the sparsity in the signal or system models.

Recently, it has been understood that by exploiting appropriately signal sparsity, significant improvement in convergence rate and steady-state performance can be achieved. As a consequence, many extensions of the classical algorithms were proposed aiming at exploiting sparsity. One of the most widely used approaches consists in updating each filter coefficient using a step-size proportional to its magnitude in order to speed up the convergence rate of the coefficients with large magnitudes. This approach led to the development of a family of algorithms known as *proportionate* [41, 96–99]. Another interesting approach to exploit sparsity is to include a *sparsity-promoting penalty* (sometimes called regularization) function into the original optimization problem of classical algorithms [1]. Within this approach, most algorithms employ the l_1 norm as the sparsity-promoting penalty [100–103], but recently an approximation to the l_0 norm has shown some advantages [8, 104–106]. In addition, these two approaches were combined and tested in [107, 108] yielding interesting results. Observe that in all of the aforementioned approaches something is being included/added to the classical algorithms, thus entailing an increase in their computational complexity.

In this chapter, we use a different strategy to exploit sparsity. Instead of including additional features in the algorithm, as the techniques described in the previous paragraph, we actually discard some coefficients, thus reducing the computational burden. This idea is motivated by the existence of some uncertainty in the coefficients in practical applications. Indeed, a measured sparse impulse response of a system presents a few coefficients concentrating most of the energy, whereas the other coefficients are close to zero, but not precisely equal to zero [8] [1]. Thus, if we have some prior information about the uncertainty in those parameters, then we can replace the parameters which are "lower than" this uncertainty with zero (i.e., discard the coefficients) in order to save computational resources.

In addition to this new way of exploiting sparsity, we also employ the set-membership filtering (SMF) approach [2, 9] in order to generate the Simple Set-Membership Affine Projection (S-SM-AP) algorithm, which is mostly the combination of the set-membership affine projection algorithm [31] with our strategy to exploit sparsity. The SMF approach is used just to reduce the computational burden even further since the filter coefficients are updated only when the estimation error is greater than a predetermined threshold.

Moreover, we derive the improved S-SM-AP (IS-SM-AP) algorithm to reduce the overall number of computations required by the S-SM-AP algorithm even further by replacing small coefficients with zero. Also, we obtain the simple affine projection (S-AP) and the improved S-AP (IS-AP) algorithms as special cases of the S-SM-AP and the IS-SM-AP algorithms, respectively. The S-AP and the IS-AP algorithms do not resort to the SMF concept and can be regarded as affine projection algorithms for sparse systems.

Finally, we introduce some sparsity-aware RLS algorithms employing the discard function and the l_0 norm approximation. The first proposed algorithm, the RLS for sparse systems (S-RLS), sets low weights to the coefficients close to zero and exploits system sparsity with low computational complexity. On the other hand, the second algorithm, the l_0 norm RLS (l_0-RLS), has higher computational complexity in comparison with the S-RLS algorithm. For both algorithms, in order to reduce the computational load further, we apply a data-selective strategy [9] leading to the data-selective S-RLS (DS-S-RLS) and the data-selective l_0-RLS (DS-l_0-RLS) algorithms. That is, the proposed algorithms update the weight vector if the output estimation

[1] A system whose impulse response presents this characteristic is formally known as a *compressible system* [1].

error is larger than a prescribed value. By applying the data-selective strategy, both algorithms attain lower computational complexity compared to the RLS algorithm.

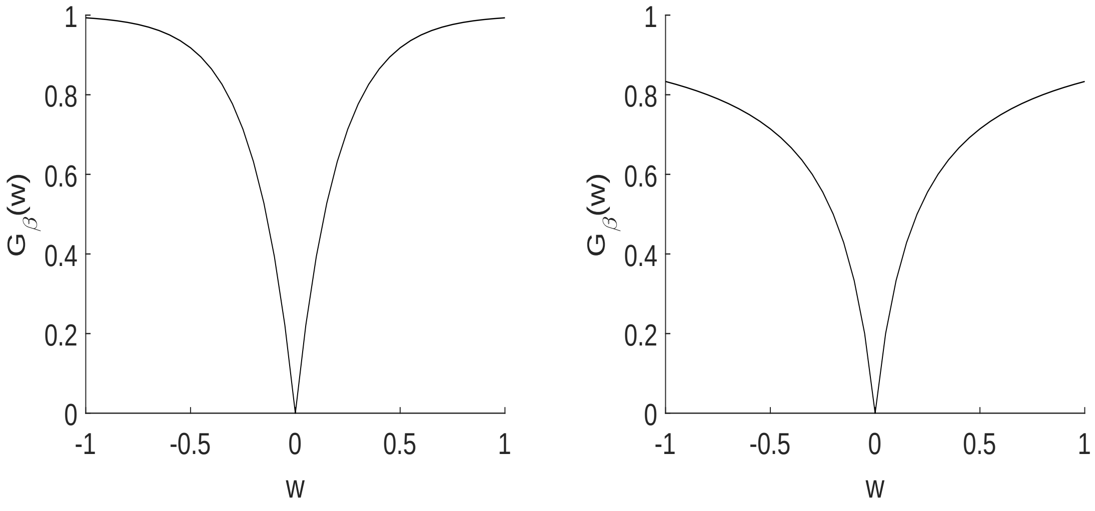
The content of this chapter was published in [109, 110]. In Sections 6.1 and 6.2, we review the sparsity-aware SM-AP (SSM-AP) algorithm and the set-membership proportionate AP algorithm (SM-PAPA), respectively. The proposed S-SM-AP algorithm is derived in Section 6.3. Sections 6.5 and 6.6 propose the S-RLS and the l_0-RLS algorithms, respectively. Simulations are presented in Section 6.7 and Section 6.8 contains the conclusions.

6.1 Sparsity-Aware SM-AP Algorithm

In literature, a method to deal with the sparsity has been obtained by adding a penalty function to the original objective function [1, 8, 100, 104, 105]. This penalty function is generally related to the l_0 or l_1 norms. Utilizing l_0 norm has some difficulties since it leads to an NP-hard problem. Therefore, we must try to approximate the l_0 norm by *almost everywhere* differentiable functions, for then we can apply stochastic gradient methods to solve the optimization problem. In other words, the l_0 norm can be estimated by a continuous function $G_\beta : \mathbb{R}^{N+1} \rightarrow \mathbb{R}_+$, where $\beta \in \mathbb{R}_+$ is a parameter responsible for controlling the agreement between quality of the estimation and smoothness of G_β. This function must satisfy the following condition [1, 8]

$$\lim_{\beta\rightarrow\infty} G_\beta(\mathbf{w}) = \|\mathbf{w}\|_0, \tag{6.1}$$

where $\|\cdot\|_0$ denotes the l_0 norm which, for $\mathbf{w} \in \mathbb{R}^{N+1}$, is defined as $\|\mathbf{w}\|_0 \triangleq \#\{i \in \mathbb{N} : \ w_i \neq 0\}$, in which # stands for the cardinality of a finite set. Here we present



(a) (b)

Figure 6.1: Univariate functions $G_\beta(w)$, with $w \in [-1, 1]$ and $\beta = 5$: (a) LF; (b) GMF.

four examples of function G_β [1, 8]

$$\text{LF} : G_\beta(\mathbf{w}) = \sum_{i=0}^{N}(1 - e^{-\beta|w_i|}), \tag{6.2a}$$

$$\text{MLF} : G_\beta(\mathbf{w}) = \sum_{i=0}^{N}(1 - e^{-0.5\beta^2 w_i^2}), \tag{6.2b}$$

$$\text{GMF} : G_\beta(\mathbf{w}) = \sum_{i=0}^{N}(1 - \frac{1}{1+\beta|w_i|}), \tag{6.2c}$$

$$\text{MGMF} : G_\beta(\mathbf{w}) = \sum_{i=0}^{N}(1 - \frac{1}{1+\beta^2 w_i^2}). \tag{6.2d}$$

The functions expressed in Equations (6.2a) and (6.2c) are called the multivariate Laplace function (LF) and the multivariate Geman-McClure function (GMF), respectively. Equations (6.2b) and (6.2d) are modifications of the LF and the GMF, respectively, so that they have continuous derivatives too. Figure 6.1 shows the univariate Laplace and Geman-McClure functions for $\beta = 5$.

The gradient of G_β is defined as follows

$$\nabla G_\beta(\mathbf{w}) \triangleq \mathbf{g}_\beta(\mathbf{w}) \triangleq [g_\beta(w_0) \ \cdots \ g_\beta(w_N)]^T, \tag{6.3}$$

where $g_\beta(w_i) = \frac{\partial G_\beta(\mathbf{w})}{\partial w_i}$. Note that (6.2a) and (6.2c) are not differentiable at the origin, thus we define their derivatives at the origin equal to zero. The derivatives corresponding to (6.2a)-(6.2d) are, respectively,

$$g_\beta(w_i) = \beta \text{sgn}(w_i) \text{e}^{-\beta|w_i|}, \tag{6.4a}$$

$$g_\beta(w_i) = \beta^2 w_i \text{e}^{-0.5\beta^2 w_i^2}, \tag{6.4b}$$

$$g_\beta(w_i) = \frac{\beta \text{sgn}(w_i)}{(1+\beta|w_i|)^2}, \tag{6.4c}$$

$$g_\beta(w_i) = \frac{2\beta^2 w_i}{(1+\beta^2 w_i^2)^2}, \tag{6.4d}$$

where sgn$(\cdot)$ denotes the sign function. The interested reader can find the details of approximating the l_0 norm in [8].

The SSM-AP algorithm performs an update whenever $|e(k)| = |d(k) - \mathbf{w}^T(k)\mathbf{x}(k)| > \overline{\gamma}$, following an update recursion that is an approximation of the solution to the optimization problem [8]

$$\begin{aligned} &\min \|\mathbf{w}(k+1) - \mathbf{w}(k)\|_2^2 + \alpha \|\mathbf{w}(k+1)\|_0 \\ &\text{subject to} \\ &\mathbf{d}(k) - \mathbf{X}^T(k)\mathbf{w}(k+1) = \boldsymbol{\gamma}(k), \end{aligned} \tag{6.5}$$

where $\alpha \in \mathbb{R}_+$ denotes the weight given to the l_0 norm.

After replacing the l_0 norm with its approximation G_β, and using the method of Lagrange multipliers, the updating equation of the SSM-AP algorithm is reached as follows [8]

$$\mathbf{w}(k+1) = \begin{cases} \mathbf{w}(k) + \mathbf{X}(k)\mathbf{A}(k)[\mathbf{e}(k) - \boldsymbol{\gamma}(k)] & \\ +\frac{\alpha}{2}[\mathbf{X}(k)\mathbf{A}(k)\mathbf{X}^T(k) - \mathbf{I}]\mathbf{g}_\beta(\mathbf{w}(k)) & \text{if } |e(k)| > \overline{\gamma}, \\ \mathbf{w}(k) & \text{otherwise,} \end{cases} \tag{6.6}$$

where $\mathbf{A}(k) = (\mathbf{X}^T(k)\mathbf{X}(k))^{-1}$.

6.2 Set-Membership Proportionate AP Algorithm

The sparsity of the signals in some applications motivates us to update each coefficient of the model independently of the others. Therefore, in adaptive filtering, one of the most widely used methods to exploit sparsity is by implementing coefficient updates that are proportional to the magnitude of the related coefficients. Thus, the coefficients with large magnitude will update with higher convergence rate and, as a result, we have faster overall convergence speed [97]. This approach leads to a well known family of algorithms called proportionate. A noticeable number of algorithms utilizing the proportionate approach have been already introduced in the literature. Some of them are the proportionate NLMS (PNLMS) [96], the proportionate AP algorithm (PAPA) [99], and their set-membership counterparts [41]. In this section, we review the set-membership PAPA (SM-PAPA). The optimization criterion of the SM-PAPA when it implements an update (i.e., when $|e(k)| > \overline{\gamma}$) is given by

$$\begin{aligned}
&\min \|\mathbf{w}(k+1) - \mathbf{w}(k)\|^2_{\mathbf{M}^{-1}(k)} \\
&\text{subject to} \\
&\mathbf{d}(k) - \mathbf{X}^T(k)\mathbf{w}(k+1) = \boldsymbol{\gamma}(k).
\end{aligned} \tag{6.7}$$

The norm in this optimization criterion is defined as $\|\mathbf{w}\|^2_{\mathbf{M}} \triangleq \mathbf{w}^T\mathbf{M}\mathbf{w}$ and matrix $\mathbf{M}(k)$ is a diagonal weighting matrix of the form

$$\mathbf{M}(k) \triangleq \text{diag}[m_0(k) \ \cdots \ m_N(k)], \tag{6.8}$$

where

$$m_i(k) \triangleq \frac{1 - r\mu(k)}{N} + \frac{r\mu(k)|w_i(k)|}{\|\mathbf{w}(k)\|_1}, \tag{6.9}$$

with

$$\mu(k) = \begin{cases} 1 - \frac{\overline{\gamma}}{|e(k)|} & \text{if } |e(k)| > \overline{\gamma}, \\ 0 & \text{otherwise}, \end{cases} \tag{6.10}$$

and $r \in [0, 1]$. Also, $\| \cdot \|_1$ stands for the l_1 norm and for $\mathbf{w} \in \mathbb{R}^{N+1}$ it is defined as $\|\mathbf{w}\|_1 = \sum_{i=0}^{N} |w_i|$. Utilizing the method of Lagrange multipliers to solve (6.7), the update equation of the SM-PAPA is obtained as follows [41]

$$\mathbf{w}(k+1) = \begin{cases} \mathbf{w}(k) + \mathbf{M}(k)\mathbf{X}(k)[\mathbf{X}^T(k)\mathbf{M}(k)\mathbf{X}(k)]^{-1}[\mathbf{e}(k) - \boldsymbol{\gamma}(k)] & \text{if } |e(k)| > \overline{\gamma}, \\ \mathbf{w}(k) & \text{otherwise}. \end{cases} \tag{6.11}$$

6.3 A Simple Set-Membership Affine Projection Algorithm

In the previous sections, we have observed that to exploit sparsity, we require a higher number of arithmetic operations compared to the SM-AP algorithm, which cannot exploit sparsity. Here we introduce a new algorithm to exploit sparsity with low computational complexity. In this algorithm, instead of including/adding something to the classical algorithms, we discard the coefficients close to zero.

In Subsection 6.3.1, we propose a Simple Set-Membership Affine Projection (S-SM-AP) algorithm that exploits the sparsity of the involved system with low computational complexity. For this purpose, the strategy consists in not updating the coefficients of the sparse filter which are close to zero. Then, in Subsection 6.3.2, we include a discussion of some characteristics of the proposed algorithm. In Subsection 6.3.3, we introduce an improved version of the proposed algorithm aiming at reducing the computational burden even further. Finally, in Subsection 6.3.4, we derive the S-AP and IS-AP algorithms by not employing the SMF technique.

6.3.1 Derivation of the S-SM-AP algorithm

Let us define the *discard function* $f_\epsilon : \mathbb{R} \rightarrow \mathbb{R}$ for the positive constant ϵ as follows

$$f_\epsilon(w) = \begin{cases} w & \text{if } |w| > \epsilon, \\ 0 & \text{if } |w| \leq \epsilon. \end{cases} \tag{6.12}$$

That is, function f_ϵ discards the values of w which are close to zero. The parameter ϵ defines what is considered as close to zero and, therefore, should be chosen based on some *a priori* information about the relative importance of a coefficient to the sparse system. Figure 6.2 depicts the function $f_\epsilon(w)$ for $\epsilon = 10^{-4}$. Note that the function $f_\epsilon(w)$ is not differentiable at $\pm\epsilon$, however, we need to differentiate this function in order to derive the S-SM-AP algorithm. To address this issue, we define the derivative of $f_\epsilon(w)$ at $+\epsilon$ and $-\epsilon$ as equal to the left and the right derivatives, respectively. Thus, the derivative of $f_\epsilon(w)$ at $\pm\epsilon$ is zero. Define the *discard vector function* $\mathbf{f}_\epsilon : \mathbb{R}^{N+1} \rightarrow \mathbb{R}^{N+1}$ as

$$\mathbf{f}_\epsilon(\mathbf{w}) = [f_\epsilon(w_0) \ \cdots \ f_\epsilon(w_N)]^T. \tag{6.13}$$

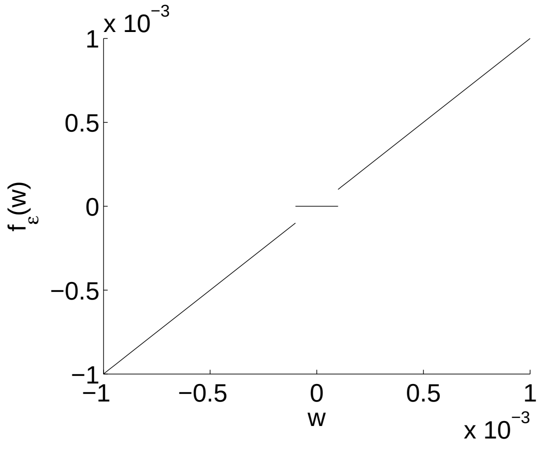


Figure 6.2: Discard function $f_\epsilon(w)$ for $\epsilon = 10^{-4}$.

The S-SM-AP algorithm updates the coefficients whose absolute values are larger than ϵ whenever the error is such that $|e(k)| = |d(k) - \mathbf{w}^T(k)\mathbf{x}(k)| > \overline{\gamma}$. Let $\psi^{L+1}(k)$ denote the intersection of the last $L+1$ constraint sets and state the following optimization criterion for the vector update whenever $\mathbf{w}(k) \notin \psi^{L+1}(k)$

$$\begin{aligned}&\min \frac{1}{2}\|\mathbf{f}_\epsilon(\mathbf{w}(k+1)) - \mathbf{w}(k)\|^2\\&\text{subject to}\\&\mathbf{d}(k) - \mathbf{X}^T(k)\mathbf{w}(k+1) = \boldsymbol{\gamma}(k).\end{aligned} \tag{6.14}$$

In order to solve this optimization problem, we construct the Lagrangian $\mathbb{L}$ as

$$\mathbb{L} = \frac{1}{2}\|\mathbf{f}_\epsilon(\mathbf{w}(k+1)) - \mathbf{w}(k)\|^2 + \boldsymbol{\lambda}^T(k)[\mathbf{d}(k) - \mathbf{X}^T(k)\mathbf{w}(k+1) - \boldsymbol{\gamma}(k)], \tag{6.15}$$

where $\boldsymbol{\lambda}(k) \in \mathbb{R}^{L+1}$ is a vector of Lagrange multipliers. After differentiating the above equation with respect to $\mathbf{w}(k+1)$ and setting the result equal to zero, we obtain

$$\mathbf{f}_\epsilon(\mathbf{w}(k+1)) = \mathbf{w}(k) + \mathbf{F}_\epsilon^{-1}(\mathbf{w}(k+1))\mathbf{X}(k)\boldsymbol{\lambda}(k), \tag{6.16}$$

where $\mathbf{F}_\epsilon(\mathbf{w}(k+1))$ is the Jacobian matrix of $\mathbf{f}_\epsilon(\mathbf{w}(k+1))$. In Equation (6.16), by employing a similar strategy as the PASTd (projection approximation subspace tracking with deflation) [111], we replace $\mathbf{f}_\epsilon(\mathbf{w}(k+1))$ and $\mathbf{F}_\epsilon^{-1}(\mathbf{w}(k+1))$ with $\mathbf{w}(k+1)$ and $\mathbf{F}_\epsilon^{-1}(\mathbf{w}(k))$, respectively, in order to form the recursion, then we obtain

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{F}_\epsilon^{-1}(\mathbf{w}(k))\mathbf{X}(k)\boldsymbol{\lambda}(k). \tag{6.17}$$

If we substitute the above equation in the constraint relation (6.14), then we will find $\boldsymbol{\lambda}(k)$ as follows

$$\boldsymbol{\lambda}(k) = (\mathbf{X}^T(k)\mathbf{F}_\epsilon^{-1}(\mathbf{w}(k))\mathbf{X}(k))^{-1}(\mathbf{e}(k) - \boldsymbol{\gamma}(k)). \tag{6.18}$$

Replacing (6.18) into (6.17) leads to the following updating equation

$$\begin{aligned}\mathbf{w}(k+1) &= \mathbf{w}(k) \\ &\quad + \mathbf{F}_\epsilon^{-1}(\mathbf{w}(k))\mathbf{X}(k)(\mathbf{X}^T(k)\mathbf{F}_\epsilon^{-1}(\mathbf{w}(k))\mathbf{X}(k))^{-1}(\mathbf{e}(k) - \boldsymbol{\gamma}(k)).\end{aligned} \tag{6.19}$$

Note that $\mathbf{F}_\epsilon(\mathbf{w}(k))$ is not an invertible matrix and, therefore, we apply the Moore-Penrose pseudoinverse (generalization of the inverse matrix) instead of the standard inverse. However, $\mathbf{F}_\epsilon(\mathbf{w}(k))$ is a diagonal matrix with diagonal entries equal to zero or one. Indeed, for the components of $\mathbf{w}(k)$ whose absolute values are larger than ϵ, their corresponding entries on the diagonal matrix $\mathbf{F}_\epsilon(\mathbf{w}(k))$ are equal to one, whereas the remaining entries are zero. Hence, the pseudoinverse of $\mathbf{F}_\epsilon(\mathbf{w}(k))$ is again $\mathbf{F}_\epsilon(\mathbf{w}(k))$. As a result, the update equation of the S-SM-AP algorithm is as follows

$$\mathbf{w}(k+1) = \begin{cases} \mathbf{w}(k) + \mathbf{q}(k) & \text{if } |e(k)| > \overline{\gamma}, \\ \mathbf{w}(k) & \text{otherwise,} \end{cases} \tag{6.20}$$

where

$$\mathbf{q}(k) = \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{X}(k)[\mathbf{X}^T(k)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{X}(k) + \delta\mathbf{I}]^{-1}(\mathbf{e}(k) - \boldsymbol{\gamma}(k)). \tag{6.21}$$

Note that, we applied a regularization factor $\delta\mathbf{I}$ in (6.21) in order to avoid numerical problems in the matrix inversion. The S-SM-AP algorithm is described in Table 6.1.

6.3.2 Discussion of the S-SM-AP algorithm

Computational Complexity

The update equation of the S-SM-AP algorithm is similar to the update equation of the SM-AP algorithm, but the former one updates only the subset of coefficients of $\mathbf{w}(k)$ whose absolute values are larger than ϵ. As a result, the role of matrix $\mathbf{F}_\epsilon(\mathbf{w}(k))$ is to discard some coefficients of $\mathbf{w}(k)$, thus reducing the computational complexity when compared to the SM-AP algorithm.

Table 6.1: Simple set-membership affine projection algorithm (S-SM-AP)

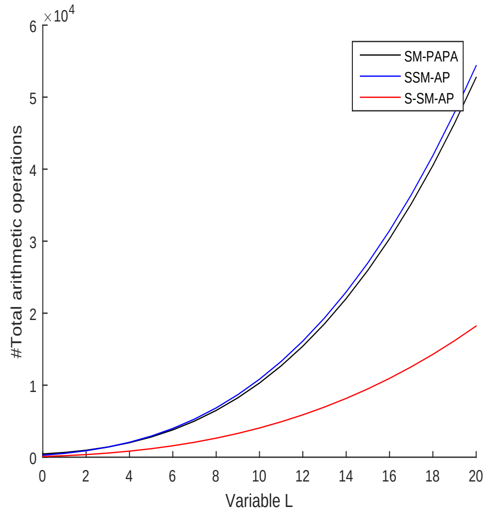
```
S-SM-AP Algorithm

Initialization
$\mathbf{w}(0) = [1\ 1\ \cdots\ 1]^T$
choose $\overline{\gamma}$ around $\sqrt{5\sigma_n^2}$ and small constant $\delta > 0$
Do for $k > 0$
    $\mathbf{e}(k) = \mathbf{d}(k) - \mathbf{X}^T(k)\mathbf{w}(k)$
    if $|e(k)| > \overline{\gamma}$
      $\mathbf{q}(k) = \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{X}(k)[\mathbf{X}^T(k)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{X}(k) + \delta\mathbf{I}]^{-1}(\mathbf{e}(k) - \boldsymbol{\gamma}(k))$
      $\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{q}(k)$
    else
      $\mathbf{w}(k+1) = \mathbf{w}(k)$
    end
end
```

The computational complexity for each update of the weight vector of the SM-PAPA [41], the SSM-AP [8], and the proposed S-SM-AP algorithms are listed in Table 6.2. The filter order and the memory length factors are N and L, respectively. It should be noted that the number of operations in Table 6.2 is presented for the full update of all coefficients. In other words, for the S-SM-AP algorithm we have presented the worst case scenario which is equivalent to setting $\epsilon = 0$,[2] while in practice we are updating only the coefficients with absolute values larger than a predetermined positive constant. Also, it is notable that the number of divisions in the S-SM-AP algorithm is less than the SM-PAPA and SSM-AP algorithms. This is quite significant, as divisions are more complex than other operations. Figures 6.3(a) and 6.3(b) show a comparison of the total number of arithmetic operations required by the SM-PAPA, the SSM-AP, and the S-SM-AP algorithms for two cases: $N = 15$, variable L and $L = 3$, variable N. As can be seen, the S-SM-AP algorithm is much less complex than the other two algorithms, especially for high values of N and L.

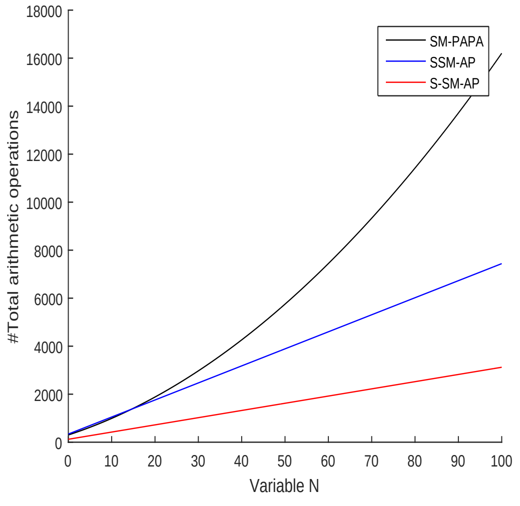
Initialization

Unlike classical algorithms in which the initialization of the weight vector is often chosen as $\mathbf{w}(0) = \mathbf{0}$, this same procedure cannot be applied to the proposed algorithm. If the initial coefficients have absolute values lower than ϵ, then the matrix $\mathbf{F}_\epsilon$ is equal to

[2] In this case, the complexity of the S-SM-AP and SM-AP algorithms are the same.



(a)



(b)

Figure 6.3: The numerical complexity of the SM-PAPA, the SSM-AP, and the IS-SM-AP algorithms for two cases: (a) $N = 15$, variable L; (b) $L = 3$, variable N.

Table 6.2: Number of operations for SM-PAPA, SSM-AP, and S-SM-AP algorithms

Algorithm	Addition & Subtraction	Multiplication	Division
SM-PAPA	$N^2 + (L^2 + 4L + 5)N+$ $(2L^3 + 5L^2 + 7L + 5)$	$(L^2 + 5L + 7)N+$ $(2L^3 + 6L^2 + 9L + 8)$	$2N+$ $(2L^2 + 4L + 4)$
SSM-AP	$(L^2 + 6L + 7)N+$ $(2L^3 + 6L^2 + 9L + 7)$	$(L^2 + 6L + 9)N+$ $(2L^3 + 7L^2 + 12L + 11)$	$N+$ $(2L^2 + 4L + 3)$
S-SM-AP	$\frac{1}{2}(L^2 + 5L + 6)N$ $\frac{1}{2}(L^3 + 4L^2 + 11L + 8)$	$\frac{1}{2}(L^2 + 5L + 6)N$ $\frac{1}{2}(L^3 + 6L^2 + 11L + 8)$	L^2

the zero matrix, and it does not allow any update. Indeed, for the S-SM-AP algorithm, each of the coefficients should be initialized as $|w_i(0)| > \epsilon$ for $i = 0, 1, \cdots, N$.

Relation with other algorithms

The similarities and differences between the proposed algorithm and the SM-AP algorithm were already addressed when we discussed the complexity of these algorithms. Now, one should observe that the update equation of the S-SM-AP algorithm is similar to the one of the set-membership partial update affine projection (SM-PUAP) algorithm [2], in which our matrix $\mathbf{F}_\epsilon(\mathbf{w}(k))$ is replaced by a diagonal matrix $\mathbf{C}$ also with entries equal to 1 or 0, but there is no specific form to set/select $\mathbf{C}$. Therefore, the proposed algorithm can be considered as a particular case of the SM-PUAP in which there is a mathematically defined way (based on the sparsity of the unknown system) to select the coefficients that are relevant and the ones that will be discarded. Regarding the memory requirements of the proposed algorithm, they are the same as in the AP algorithm, i.e., determined by the data-reuse factor L.

6.3.3 The Improved S-SM-AP (IS-SM-AP) algorithm

As we can observe in the update equation of the S-SM-AP algorithm, if a coefficient of the weight vector falls inside the interval $[-\epsilon, +\epsilon]$, then in the next update this coefficient does not update since it is eliminated by the discard function. On the other hand, the coefficients $w_i(k)$ inside the interval $[-\epsilon, +\epsilon]$ are close to zero, and the best intuitive approximation for them is zero (the center of the interval). Besides, making these coefficients $w_i(k)$ equal to zero implies in a reduction of computational complexity, because it reduces the number of operations required to compute the output of the adaptive filter $y(k) = \mathbf{x}^T(k)\mathbf{w}(k)$.[3] For this purpose, we multiply $\mathbf{w}(k)$ by $\mathbf{F}_\epsilon(\mathbf{w}(k))$, and obtain the Improved S-SM-AP (IS-SM-AP) algorithm as follows

$$\mathbf{w}(k+1) = \begin{cases} \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{w}(k) + \mathbf{q}(k) & \text{if } |e(k)| > \overline{\gamma}, \\ \mathbf{w}(k) & \text{otherwise.} \end{cases} \tag{6.22}$$

Table 6.3 illustrates the IS-SM-AP algorithm.

6.3.4 The S-AP and the IS-AP algorithms

By adopting the bound $\overline{\gamma} = 0$, the S-SM-AP algorithm will convert to the S-AP algorithm with unity step size. Therefore, the S-AP algorithm can be described as follows

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{X}(k)[\mathbf{X}^T(k)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{X}(k) + \delta\mathbf{I}]^{-1}\mathbf{e}(k) \tag{6.23}$$

where μ is the convergence factor.

By the same argument, we can obtain the update equation of the IS-AP algorithm

[3]This additional reduction in the number of operations becomes more important as the filter order increases. For instance, in acoustic echo cancellation systems, in which the adaptive filter has a few thousands of coefficients [112, 113], this simple strategy implies in significant computational savings.

Table 6.3: Improved simple set-membership affine projection algorithm (IS-SM-AP)

IS-SM-AP Algorithm

```
Initialization
$\mathbf{w}(0) = [1\ 1\ \cdots\ 1]^T$
choose $\overline{\gamma}$ around $\sqrt{5\sigma_n^2}$ and small constant $\delta > 0$
Do for $k > 0$
    $\mathbf{e}(k) = \mathbf{d}(k) - \mathbf{X}^T(k)\mathbf{w}(k)$
    if $|e(k)| > \overline{\gamma}$
      $\mathbf{q}(k) = \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{X}(k)[\mathbf{X}^T(k)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{X}(k) + \delta\mathbf{I}]^{-1}(\mathbf{e}(k) - \boldsymbol{\gamma}(k))$
      $\mathbf{w}(k+1) = \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{w}(k) + \mathbf{q}(k)$
    else
      $\mathbf{w}(k+1) = \mathbf{w}(k)$
    end
end
```

as below

$$\begin{aligned}\mathbf{w}(k+1) =&\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{w}(k)\\ &+ \mu\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{X}(k)[\mathbf{X}^T(k)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{X}(k) + \delta\mathbf{I}]^{-1}\mathbf{e}(k)\end{aligned} \tag{6.24}$$

where μ is the convergence factor. These algorithms are counterparts of the AP algorithm, however they can exploit the sparsity in systems.

Remark: In the previous sections, we have focused on the AP algorithms. However, the NLMS and the binormalized data-reusing LMS algorithms can be derived as special cases of the AP algorithms. Indeed, by choosing $L = 0$ and 1, the AP algorithms will be reduced to the NLMS and the binormalized data-reusing LMS algorithms, respectively.

6.4 Some issues of the S-SM-AP and the IS-SM-AP Algorithms

As we discussed in Subsection 6.3.2, the proposed S-SM-AP and the IS-SM-AP algorithms are sensitive to the initialization. In fact, the absolute value of parameters of $\mathbf{w}(0)$ have to be greater than ϵ and $w_i(0)w_{oi} > 0$ for $i = 0, \cdots, N$, i.e., $w_i(0)$ and w_{oi} must have the same sign, where w_{oi} is the i-th component of the unknown system. Moreover, when the system is time-varying, these algorithms cannot track the system.

Table 6.4: Discard set-membership affine projection algorithm (D-SM-AP)

```
D-SM-AP Algorithm

Initialization
$\mathbf{w}(0) = \mathbf{0}$ and $\mathbf{m}(0) = \mathbf{0}$
choose $\overline{\gamma}$ around $\sqrt{5\sigma_n^2}$ and small constant $\delta > 0$
Do for $k > 0$
    $\mathbf{e}(k) = \mathbf{d}(k) - \mathbf{X}^T(k)\mathbf{w}(k)$
    $\mathbf{m}(k+1) = \begin{cases} \mathbf{m}(k) + \mathbf{X}(k)[\mathbf{X}^T(k)\mathbf{X}(k) + \delta\mathbf{I}]^{-1}(\mathbf{e}(k) - \boldsymbol{\gamma}(k)) & \text{if } |e(k)| > \overline{\gamma} \\ \mathbf{m}(k) & \text{otherwise} \end{cases}$
    $\mathbf{w}(k+1) = \mathbf{F}_\epsilon(\mathbf{m}(k+1))\mathbf{m}(k+1)$
end
```

In other words, if a coefficient falls inside $[-\epsilon, \epsilon]$, then it cannot go out. Thus, in the case of time-varying systems, it means that the algorithm is unable to track the system.

To address this issue, we can use an auxiliary weight vector $\mathbf{m}(k)$ as in [114]. Through this technique, the discard function applies only to the auxiliary weight vector, and we can propose the discard SM-AP (D-SM-AP) algorithm. The D-SM-AP algorithm is presented in Table 6.4. Note that the computational burden of the D-SM-AP algorithm is higher than the IS-SM-AP and the S-SM-AP algorithms. However, it can be utilized in time-varying systems, and we can adopt any initialization $\mathbf{w}(0)$.

6.5 Recursive Least-Squares Algorithm Exploiting Sparsity

In this section, we utilize the discard function to introduce an RLS algorithm for sparse systems. In Subsection 6.5.1, we derive the S-RLS algorithm that exploits the sparsity of the estimated parameters by giving low weight to the small coefficients. For this purpose, the strategy consists in multiplying the coefficients of the sparse filter which are close to zero by a small constant. Then, in Subsection 6.5.2, we include a discussion of some characteristics of the proposed algorithm. Subsection 6.5.3 briefly describes the DS-S-RLS algorithm, the data-selective version of the S-RLS algorithm.

6.5.1 Derivation of the S-RLS algorithm

We utilize the discard vector function defined in Equation (6.13) in order to introduce the objective function of the S-RLS algorithm as follows

$$\min \xi^d(k) = \sum_{i=0}^{k} \lambda^{k-i}[d(i) - \mathbf{x}^T(i)\mathbf{f}_\epsilon(\mathbf{w}(k))]^2, \tag{6.25}$$

where the parameter λ is an exponential weighting factor that should be selected in the range $0 \ll \lambda \leq 1$.

By differentiating $\xi^d(k)$ with respect to $\mathbf{w}(k)$, we obtain

$$\frac{\partial \xi^d(k)}{\partial \mathbf{w}(k)} = -2\sum_{i=0}^{k} \lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)[d(i) - \mathbf{x}^T(i)\mathbf{f}_\epsilon(\mathbf{w}(k))], \tag{6.26}$$

where $\mathbf{F}_\epsilon(\mathbf{w}(k))$ is the Jacobian matrix of $\mathbf{f}_\epsilon(\mathbf{w}(k))$ (see (6.13)). By equating the above equation to zero, we find the optimal vector $\mathbf{w}(k)$ that solves the least-square problem, as follows

$$-\sum_{i=0}^{k} \lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)\mathbf{x}^T(i)\mathbf{f}_\epsilon(\mathbf{w}(k)) + \sum_{i=0}^{k} \lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)d(i) = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{6.27}$$

Therefore,

$$\mathbf{f}_\epsilon(\mathbf{w}(k)) = \Big[\sum_{i=0}^{k} \lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)\mathbf{x}^T(i)\Big]^{-1} \times \sum_{i=0}^{k} \lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)d(i). \tag{6.28}$$

Note that $\mathbf{F}_\epsilon(\mathbf{w}(k))$ is a diagonal matrix with diagonal entries equal to zero or one. Indeed, for the components of $\mathbf{w}(k)$ whose absolute values are larger than ϵ, their corresponding entries on the diagonal matrix $\mathbf{F}_\epsilon(\mathbf{w}(k))$ are one, whereas the remaining entries are zero. Hence,

$$\begin{aligned}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)\mathbf{x}^T(i) &= \mathbf{F}_\epsilon^2(\mathbf{w}(k))\mathbf{x}(i)\mathbf{x}^T(i) = \mathbf{F}_\epsilon(\mathbf{w}(k))(\mathbf{x}^T(i)\mathbf{F}_\epsilon(\mathbf{w}(k)))^T\mathbf{x}^T(i) \\ &= \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)\mathbf{x}^T(i)\mathbf{F}_\epsilon(\mathbf{w}(k)).\end{aligned} \tag{6.29}$$

By utilizing (6.29) in (6.28) and replacing $\mathbf{f}_\epsilon(\mathbf{w}(k))$ by $\mathbf{w}(k+1)$, we get

$$\begin{aligned}\mathbf{w}(k+1) &= \Big[\sum_{i=0}^{k}\lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)\mathbf{x}^T(i)\mathbf{F}_\epsilon(\mathbf{w}(k))\Big]^{-1} \times \sum_{i=0}^{k}\lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)d(i)\\ &= \mathbf{R}_{D,\epsilon}^{-1}(k)\mathbf{p}_{D,\epsilon}(k),\end{aligned} \tag{6.30}$$

where $\mathbf{R}_{D,\epsilon}(k)$ and $\mathbf{p}_{D,\epsilon}(k)$ are called the deterministic correlation matrix of the input signal and the deterministic cross-correlation vector between the input and the desired signals, respectively. Whenever the i-th diagonal entry of matrix $\mathbf{F}_\epsilon(\mathbf{w}(k))$ is zero, it is replaced by a small power-of-two (e.g., 2^{-5}) multiplied by the sign of the component $w_i(k)$ in order to avoid that matrix $\mathbf{R}_{D,\epsilon}(k)$ becomes ill conditioned.

If we apply the direct method to calculate the inverse of $\mathbf{R}_{D,\epsilon}(k)$, then the resulting algorithm has computational complexity of $O[N^3]$. Generally, in the traditional RLS algorithm, the inverse matrix is computed through the matrix inversion lemma [32]. In matrix inversion lemma, we have

$$[\mathbf{A}+\mathbf{BCD}]^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}[\mathbf{DA}^{-1}\mathbf{B}+\mathbf{C}^{-1}]^{-1}\mathbf{DA}^{-1}, \tag{6.31}$$

where $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$ are matrices of appropriate dimensions, and $\mathbf{A}$ and $\mathbf{C}$ are invertible. If we choose $\mathbf{A} = \lambda\mathbf{R}_{D,\epsilon}(k-1)$, $\mathbf{B} = \mathbf{D}^T = \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)$, and $\mathbf{C} = 1$ then by using the matrix inversion lemma, the inverse of the deterministic correlation matrix can be calculated in the form

$$\begin{aligned}\mathbf{S}_{D,\epsilon}(k) =&\mathbf{R}_{D,\epsilon}^{-1}(k)\\ =&\frac{1}{\lambda}\Big[\mathbf{S}_{D,\epsilon}(k-1) - \frac{\mathbf{S}_{D,\epsilon}(k-1)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)\mathbf{x}^T(k)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{S}_{D,\epsilon}(k-1)}{\lambda+\mathbf{x}^T(k)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{S}_{D,\epsilon}(k-1)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)}\Big].\end{aligned} \tag{6.32}$$

The resulting equation to compute $\mathbf{R}_{D,\epsilon}^{-1}(k)$ has computational complexity of $O[N^2]$, whereas the computational resources for the direct inversion is of order N^3. Finally,

$$\mathbf{w}(k+1) = \mathbf{S}_{D,\epsilon}(k)\mathbf{p}_{D,\epsilon}(k). \tag{6.33}$$

Table 6.5 describes the S-RLS algorithm.

We can introduce the alternative S-RLS (AS-RLS) algorithm in order to decrease the computational load of the S-RLS. Assuming $\mathbf{F}_\epsilon(\mathbf{w}(k)) \approx \mathbf{F}_\epsilon(\mathbf{w}(k-1))$, we can

Table 6.5: Recursive least-squares algorithm for sparse systems (S-RLS)

```
S-RLS Algorithm

Initialization
$\mathbf{S}_{D,\epsilon}(-1) = \delta\mathbf{I}$
where $\delta$ can be the inverse of the input signal power estimate
$\mathbf{p}_{D,\epsilon}(-1) = [0\ 0\ \cdots\ 0]^T$
$\mathbf{w}(0) = [1\ 1\ \cdots\ 1]^T$
Do for $k \geq 0$
    compute $\mathbf{S}_{D,\epsilon}(k)$ through Equation (6.32)
    $\mathbf{p}_{D,\epsilon}(k) = \lambda\mathbf{p}_{D,\epsilon}(k-1) + \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)d(k)$
    $\mathbf{w}(k+1) = \mathbf{S}_{D,\epsilon}(k)\mathbf{p}_{D,\epsilon}(k)$
end
```

rewrite Equation (6.30) as

$$\begin{aligned}
&\Big[\sum_{i=0}^{k} \lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)\mathbf{x}^T(i)\mathbf{F}_\epsilon(\mathbf{w}(k))\Big]\mathbf{w}(k+1) = \sum_{i=0}^{k} \lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)d(i) \\
&= \lambda\Big[\sum_{i=0}^{k-1} \lambda^{k-i-1}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)d(i)\Big] + \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)d(k) \\
&\approx \lambda\Big[\sum_{i=0}^{k-1} \lambda^{k-i-1}\mathbf{F}_\epsilon(\mathbf{w}(k-1))\mathbf{x}(i)d(i)\Big] + \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)d(k) \\
&= \lambda\mathbf{p}_{D,\epsilon}(k-1) + \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)d(k) \qquad (6.34)
\end{aligned}$$

By considering that $\mathbf{R}_{D,\epsilon}(k-1)\mathbf{w}(k) = \mathbf{p}_{D,\epsilon}(k-1)$, we obtain

$$\begin{aligned}
&\Big[\sum_{i=0}^{k} \lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)\mathbf{x}^T(i)\mathbf{F}_\epsilon(\mathbf{w}(k))\Big]\mathbf{w}(k+1) \\
&\approx \lambda\mathbf{R}_{D,\epsilon}(k-1)\mathbf{w}(k) + \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)d(k) \\
&= \Big[\sum_{i=0}^{k-1} \lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k-1))\mathbf{x}(i)\mathbf{x}^T(i)\mathbf{F}_\epsilon(\mathbf{w}(k-1))\Big]\mathbf{w}(k) + \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)d(k) \\
&\approx \Big[\sum_{i=0}^{k} \lambda^{k-i}\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(i)\mathbf{x}^T(i)\mathbf{F}_\epsilon(\mathbf{w}(k)) - \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)\mathbf{x}^T(k)\mathbf{F}_\epsilon(\mathbf{w}(k))\Big]\mathbf{w}(k) \\
&+ \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)d(k). \qquad (6.35)
\end{aligned}$$

Table 6.6: Alternative recursive least-squares algorithm for sparse systems

AS-RLS Algorithm
Initialization $\mathbf{S}_{D,\epsilon}(-1) = \delta \mathbf{I}$ where δ can be inverse of the input signal power estimate $\mathbf{w}(0) = [1\ 1\ \cdots\ 1]^T$ Do for $k \geq 0$ $e(k) = d(k) - \mathbf{x}^T(k)\mathbf{w}(k)$ $\psi(k) = \mathbf{S}_{D,\epsilon}(k-1)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)$ $\mathbf{S}_{D,\epsilon}(k) = \frac{1}{\lambda}\Big[\mathbf{S}_{D,\epsilon}(k-1) - \frac{\psi(k)\psi^T(k)}{\lambda+\psi^T(k)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)}\Big]$ $\mathbf{w}(k+1) = \mathbf{w}(k) + e(k)\mathbf{S}_{D,\epsilon}(k)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)$ end

Then, by using Equation (6.29) and a few manipulations, we get

$$\mathbf{w}(k+1) \approx \mathbf{w}(k) + e(k)\mathbf{S}_{D,\epsilon}(k)\mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k), \tag{6.36}$$

where $e(k) = d(k) - \mathbf{x}^T(k)\mathbf{w}(k)$. Table 6.6 illustrates the AS-RLS algorithm.

6.5.2 Discussion of the S-RLS algorithm

The update equation of the S-RLS algorithm is similar to the update equation of the RLS algorithm, but the former gives importance only to the subset of coefficients of $\mathbf{w}(k)$ whose absolute values are larger than ϵ. The matrix $\mathbf{F}_\epsilon(\mathbf{w}(k))$ defines the important coefficients of $\mathbf{w}(k)$.

6.5.3 DS-S-RLS algorithm

In this subsection, our goal is to reduce the update rate of the S-RLS algorithm. In fact, when the current weight vector is acceptable, i.e., the output estimation error is small, we can save computational resources by avoiding the new update. The data selective S-RLS (DS-S-RLS) algorithm updates whenever the output estimation error is larger than a prescribed value $\overline{\gamma}$, i.e., when $|e(k)| = |d(k) - \mathbf{w}^T(k)\mathbf{x}(k)| > \overline{\gamma}$. Therefore, the DS-S-RLS algorithm reduces the computational complexity by avoiding updates whenever the estimate is acceptable. Table 6.7 describes the DS-S-RLS algorithm.

Table 6.7: Data-selective recursive least-squares algorithm for sparse systems (DS-S-RLS)

```
DS-S-RLS Algorithm

Initialization
$\mathbf{S}_{D,\epsilon}(-1) = \delta\mathbf{I}$
where $\delta$ can be the inverse of the input signal power estimate
choose $\overline{\gamma}$ around $\sqrt{5\sigma_n^2}$
$\mathbf{p}_{D,\epsilon}(-1) = [0\ 0\ \cdots\ 0]^T$
$\mathbf{w}(0) = [1\ 1\ \cdots\ 1]^T$
Do for $k \geq 0$
    $e(k) = d(k) - \mathbf{w}^T(k)\mathbf{x}(k)$
    if $|e(k)| > \overline{\gamma}$
      compute $\mathbf{S}_{D,\epsilon}(k)$ through Equation (6.32)
      $\mathbf{p}_{D,\epsilon}(k) = \lambda\mathbf{p}_{D,\epsilon}(k-1) + \mathbf{F}_\epsilon(\mathbf{w}(k))\mathbf{x}(k)d(k)$
      $\mathbf{w}(k+1) = \mathbf{S}_{D,\epsilon}(k)\mathbf{p}_{D,\epsilon}(k)$
    else
      $\mathbf{w}(k+1) = \mathbf{w}(k)$
    end
end
```

6.6 l_0 Norm Recursive Least-Squares Algorithm

In the previous section, we have introduced the S-RLS algorithm for sparse systems utilizing the discard function. Another interesting approach to exploit the system sparsity can be derived by using the l_0 norm [8] leading to the l_0-RLS algorithm. However, as mentioned earlier, the resulting optimization problem of l_0 norm has difficulties due to the discontinuity of the l_0 norm. Thus, we use Equations (6.2a)-(6.2d) to approximate the l_0 norm.

Therefore, the objective function of the l_0-RLS algorithm is given by

$$\min \sum_{i=0}^{k} \lambda^{k-i}[d(i) - \mathbf{x}^T(i)\mathbf{w}(k)]^2 + \alpha\|\mathbf{w}(k)\|_0, \tag{6.37}$$

where $\alpha \in \mathbb{R}_+$ is the weight given to the l_0 norm penalty. Replacing $\|\mathbf{w}(k)\|_0$ by its approximation, we obtain

$$\min \sum_{i=0}^{k} \lambda^{k-i}[d(i) - \mathbf{x}^T(i)\mathbf{w}(k)]^2 + \alpha G_\beta(\mathbf{w}(k)). \tag{6.38}$$

Table 6.8: l_0 norm recursive least-squares algorithm for sparse systems (l_0-RLS)

l_0**-RLS Algorithm**
Initialization $\mathbf{S}_D(-1) = \delta\mathbf{I}$ where δ can be inverse of the input signal power estimate $\mathbf{p}_D(-1) = [0\ 0\ \cdots\ 0]^T$ $\mathbf{w}(-1) = [1\ 1\ \cdots\ 1]^T$ Do for $k \geq 0$ $\mathbf{S}_D(k)$ as in Equation (6.42) $\mathbf{p}_D(k) = \lambda\mathbf{p}_D(k-1) + d(k)\mathbf{x}(k)$ $\mathbf{w}(k) = \mathbf{S}_D(k)\Big(\mathbf{p}_D(k) - \frac{\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1))\Big)$ end

By differentiating the above equation with respect to $\mathbf{w}(k)$, and equating the result to zero, we get

$$\begin{aligned}\mathbf{w}(k) =&\Big[\sum_{i=0}^{k}\lambda^{k-i}\mathbf{x}(i)\mathbf{x}^T(i)\Big]^{-1} \times \Big((\sum_{i=0}^{k}\lambda^{k-i}\mathbf{x}(i)d(i)) - \frac{\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k))\Big)\\ =&\mathbf{R}_D^{-1}(k)\Big(\mathbf{p}_D(k) - \frac{\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k))\Big).\end{aligned} \tag{6.39}$$

If we adopt $\mathbf{A} = \lambda\mathbf{R}_D(k-1)$, $\mathbf{B} = \mathbf{D}^T = \mathbf{x}(k)$, and $\mathbf{C} = 1$ then by using the matrix inversion lemma, the update equation of the l_0-RLS algorithm is given as follows

$$\mathbf{w}(k) = \mathbf{S}_D(k)\Big(\mathbf{p}_D(k) - \frac{\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1))\Big), \tag{6.40}$$

where the same strategy as the PASTd (projection approximation subspace tracking with deflation) [111] is employed and $\mathbf{g}_\beta(\mathbf{w}(k))$ is replaced by $\mathbf{g}_\beta(\mathbf{w}(k-1))$ in order to form the recursion. Also, $\mathbf{p}_D(k)$ and $\mathbf{S}_D(k)$ are given as follows

$$\mathbf{p}_D(k) =\lambda\mathbf{p}_D(k-1) + d(k)\mathbf{x}(k), \tag{6.41}$$

$$\mathbf{S}_D(k) =\frac{1}{\lambda}\Big[\mathbf{S}_D(k-1) - \frac{\mathbf{S}_D(k-1)\mathbf{x}(k)\mathbf{x}^T(k)\mathbf{S}_D(k-1)}{\lambda + \mathbf{x}^T(k)\mathbf{S}_D(k-1)\mathbf{x}(k)}\Big]. \tag{6.42}$$

Table 6.8 presents the l_0-RLS algorithm.

Similarly to the AS-RLS algorithm, we can derive the alternative l_0-RLS (A-l_0-

RLS) algorithm. We can rewrite Equation (6.40) as

$$\mathbf{R}_D(k)\mathbf{w}(k) = \mathbf{p}_D(k) - \frac{\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1)) = \lambda\mathbf{p}_D(k-1) + \mathbf{x}(k)d(k) - \frac{\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1)). \tag{6.43}$$

By Equation (6.39), we have $\mathbf{R}_D(k-1)\mathbf{w}(k-1) = \mathbf{p}_D(k-1) - \frac{\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1))$, then we get

$$\begin{aligned}\mathbf{R}_D(k)\mathbf{w}(k) =& \lambda\mathbf{R}_D(k-1)\mathbf{w}(k-1) + \frac{\lambda\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1)) \\ & - \frac{\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1)) + \mathbf{x}(k)d(k) \\ =& \Big[\sum_{j=0}^{k}\lambda^{k-i}\mathbf{x}(i)\mathbf{x}^T(i) - \mathbf{x}(k)\mathbf{x}^T(k)\Big]\mathbf{w}(k-1) \\ & + \frac{(\lambda-1)\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1)) + \mathbf{x}(k)d(k).\end{aligned} \tag{6.44}$$

If we define the *a priori* error as

$$e(k) = d(k) - \mathbf{x}^T(k)\mathbf{w}(k-1), \tag{6.45}$$

we obtain

$$\mathbf{R}_D(k)\mathbf{w}(k) = \mathbf{R}_D(k)\mathbf{w}(k-1) + e(k)\mathbf{x}(k) + \frac{(\lambda-1)\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1)). \tag{6.46}$$

Therefore, the update equation of the A-l_0-RLS algorithm is given by

$$\mathbf{w}(k) = \mathbf{w}(k-1) + \mathbf{S}_D(k)[e(k)\mathbf{x}(k) + \frac{(\lambda-1)\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1))]. \tag{6.47}$$

Table 6.9 presents the A-l_0-RLS algorithm.

6.6.1 DS-l_0-RLS algorithm

In this subsection, we propose the DS-l_0-RLS algorithm to decrease the update rate of the l_0-RLS algorithm. Similarly to the discussion in Subsection 6.5.3, the DS-l_0-RLS algorithm for sparse systems can be derived by implementing an update in the l_0-RLS algorithm whenever the output estimation error is larger than a predetermined value $\overline{\gamma}$, i.e., when $|e(k)| = |d(k) - \mathbf{w}^T(k)\mathbf{x}(k)| > \overline{\gamma}$. Hence, the computational resources of the DS-l_0-RLS algorithm is lower than the l_0-RLS algorithm since it prevents unnecessary

Table 6.9: Alternative l_0 norm recursive least-squares algorithm for sparse systems

```
A-$l_0$-RLS Algorithm

Initialization
$\mathbf{S}_D(-1) = \delta\mathbf{I}$
where $\delta$ can be inverse of the input signal power estimate
$\mathbf{w}(-1) = [1\ 1\ \cdots\ 1]^T$
Do for $k \geq 0$
    $e(k) = d(k) - \mathbf{x}^T(k)\mathbf{w}(k-1)$
    $\psi(k) = \mathbf{S}_D(k-1)\mathbf{x}(k)$
    $\mathbf{S}_D(k) = \frac{1}{\lambda}\Big[\mathbf{S}_D(k-1) - \frac{\psi(k)\psi^T(k)}{\lambda+\psi^T(k)\mathbf{x}(k)}\Big]$
    $\mathbf{w}(k) = \mathbf{w}(k-1) + \mathbf{S}_D(k)[e(k)\mathbf{x}(k) + \frac{(\lambda-1)\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1))]$
end
```

Table 6.10: Data-selective l_0 norm recursive least-squares algorithm for sparse systems (DS-l_0-RLS)

```
DS-$l_0$-RLS Algorithm

Initialization
$\mathbf{S}_D(-1) = \delta\mathbf{I}$
where $\delta$ can be inverse of the input signal power estimate
choose $\overline{\gamma}$ around $\sqrt{5\sigma_n^2}$
$\mathbf{p}_D(-1) = [0\ 0\ \cdots\ 0]^T$
$\mathbf{w}(-1) = [1\ 1\ \cdots\ 1]^T$
Do for $k \geq 0$
    $e(k) = d(k) - \mathbf{w}^T(k-1)\mathbf{x}(k)$
    if $|e(k)| > \overline{\gamma}$
        $\mathbf{S}_D(k)$ as in Equation (6.42)
        $\mathbf{p}_D(k) = \lambda\mathbf{p}_D(k-1) + d(k)\mathbf{x}(k)$
        $\mathbf{w}(k) = \mathbf{S}_D(k)\Big(\mathbf{p}_D(k) - \frac{\alpha}{2}\mathbf{g}_\beta(\mathbf{w}(k-1))\Big)$
    else
        $\mathbf{w}(k) = \mathbf{w}(k-1)$
    end
end
```

updates. The DS-l_0-RLS algorithm is described in Table 6.10.

In Subsection 6.7.2, we compare the simulation results of the RLS-based algorithms with the Adaptive Sparse Variational Bayes iterative scheme based on Laplace prior

Table 6.11: Number of operations for AS-RLS, l_0-RLS, and ASVB-L algorithms

Algorithm	Addition & Subtraction	Multiplication	Division
AS-RLS	$N^2 + 3N$	$N^2 + 5N + 1$	1
A-l_0-RLS	$N^2 + 5N$	$N^2 + 9N + 1$	$N + 1$
ASVB-L	$N^2 + 7N + 6$	$2N^2 + 10N + 3$	$6N + 2$

(ASVB-L) algorithm [115–117]. Therefore, it is worthwhile to compare the computational complexity of these algorithms. Table 6.11 shows the number of real multiplications, real additions, and real divisions must be performed at each iteration by these algorithms.

6.7 Simulations

In this section, we present some numerical simulations for the proposed algorithms. In all scenarios, we deal with the system identification problem. In Subsection 6.7.1, we apply the LMS-based algorithms. The numerical results of the RLS-based algorithms are illustrated in Subsection 6.7.2.

6.7.1 Simulation results of the LMS-based algorithms

Here, we have applied the algorithms described in this chapter, the NLMS, and the AP algorithms to identify three unknown sparse systems of order 14.[4] The first one is an arbitrary sparse system $\mathbf{w}_o$, the second one is a block sparse system $\mathbf{w}'_o$, and the third one is a symmetric-block sparse system $\mathbf{w}''_o$. The coefficients of these three systems are presented in Table 6.12. The input is a binary phase-shift keying (BPSK) signal with variance $\sigma_x^2 = 1$. The signal-to-noise ratio (SNR) is set to be 20 dB, i.e., the noise variance is $\sigma_n^2 = 0.01$. The data-reuse factor is $L = 1$, the bound on the estimation error is set to be $\overline{\gamma} = \sqrt{5\sigma_n^2}$, and the threshold bound vector $\boldsymbol{\gamma}(k)$ is selected as the *simple-choice constraint vector* [8] which is defined as $\gamma_0(k) = \frac{\overline{\gamma}e(k)}{|e(k)|}$ and $\gamma_i(k) = d(k-i) - \mathbf{w}^T(k)\mathbf{x}(k-i)$, for $i = 1, \cdots, L$. The initial vector $\mathbf{w}(0)$ and the regularization factor are $10^{-3} \times [1, \cdots, 1]^T$ and 10^{-12}, respectively. The learning curves are the results of averaging of the outcomes of 500 trials.

[4]The results for the S-SM-AP algorithm are not shown here because they are almost identical to the results of the IS-SM-AP algorithm, but the latter has the advantage of requiring fewer computations.

Table 6.12: The coefficients of unknown systems $\mathbf{w}_o$, $\mathbf{w}'_o$, and $\mathbf{w}''_o$.

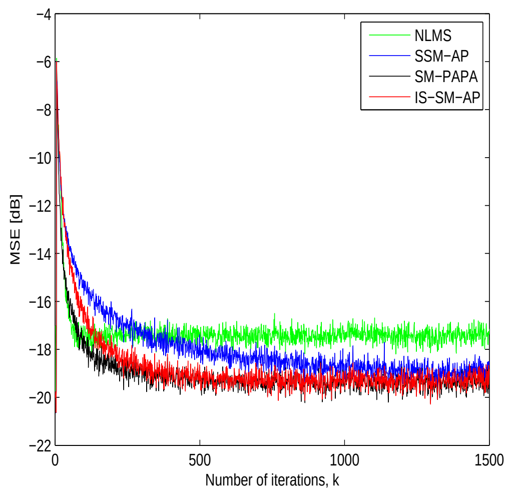
$\mathbf{w}_o$	$\mathbf{w}'_o$	$\mathbf{w}''_o$
24e-2	2e-7	2e-8
2e-8	-21e-10	-1e-9
-23e-2	17e-8	1e-7
-3e-7	21e-8	-3e-7
5e-1	-3e-7	**-64e-3**
-1e-9	**24e-2**	**2e-1**
2e-1	**7e-1**	**5e-1**
1e-7	**2e-1**	**2e-1**
-5e-8	**33e-2**	**-64e-3**
12e-6	**-6e-1**	-5e-5
1e-8	-5e-7	12e-6
-5e-6	18e-9	1e-8
4e-6	-5e-7	-5e-6
-1e-7	21e-8	4e-6
-2e-1	-11e-8	-1e-5

Scenario 1

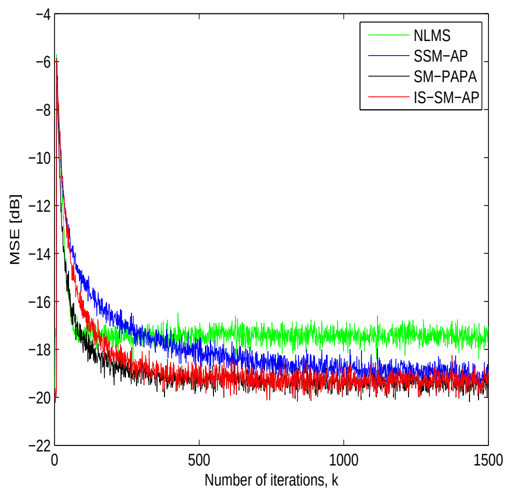
In this scenario, we have implemented the IS-SM-AP, the SSM-AP, the SM-PAPA, and the NLMS algorithms to identify the three unknown sparse systems in Table 6.12. The convergence factor of the NLMS algorithm is $\mu = 0.9$. The constant ϵ in the IS-SM-AP algorithm is chosen as 2×10^{-4}; that is, on average, 5 out of 15 coefficients (boldface coefficients in $\mathbf{w}_o$, $\mathbf{w}'_o$, and $\mathbf{w}''_o$) are updated at each iteration. We have selected $\alpha = 5 \times 10^{-3}$, $\beta = 5$, and $\varepsilon = 100$ for the SM-PAPA and the SSM-AP algorithms. In the SSM-AP algorithm, we have used the GMF as the approximation of the l_0 norm.

Figures 6.4(a), 6.4(b), and 6.4(c) depict the learning curves for the IS-SM-AP, the SM-PAPA, the SSM-AP, and the NLMS algorithms to identify the unknown systems $\mathbf{w}_o$, $\mathbf{w}'_o$, and $\mathbf{w}''_o$, respectively. The average number of updates implemented by the IS-SM-AP, the SM-PAPA, and the SSM-AP algorithms are given in columns 2 to 4 of Table 6.13.

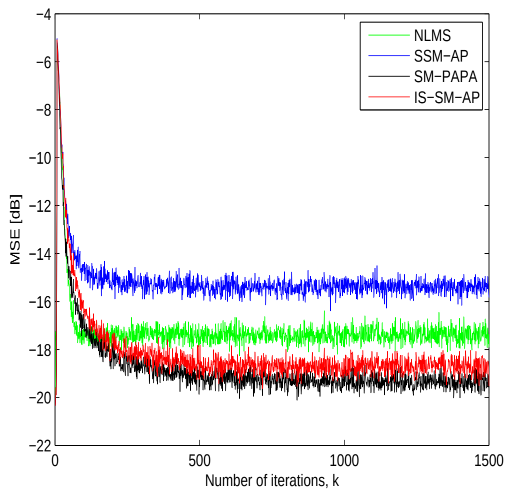
In addition, we have applied all the aforementioned algorithms in this scenario, using the parameters that were already defined in the previous paragraph, but changing the input signal model to an autoregressive (AR) process in order to identify the unknown system $\mathbf{w}_o$. The new input signal is generated as a first-order AR process



(a)



(b)



(c)

Figure 6.4: The learning curves of the SM-PAPA, the SSM-AP, the IS-SM-AP, and the NLMS algorithms applied on: (a) $\mathbf{w}_o$; (b) $\mathbf{w}'_o$; (c) $\mathbf{w}''_o$.

defined as $x(k) = 0.95x(k-1)+n(k)$. In this case, the learning curves of the algorithms are shown in Figure 6.5, and the average number of updates performed by the IS-SM-AP, the SM-PAPA, and the SSM-AP algorithms are presented in the fifth column of Table 6.13. Also, the number of arithmetic operations required by the IS-SM-AP, the SM-PAPA, and the SSM-AP algorithms in whole iterations are 41635, 110835, and 84396, respectively.

Observe that, in every scenario we tested, the IS-SM-AP algorithm performed as well as the other state-of-the-art sparsity-aware algorithms, but this algorithm has the

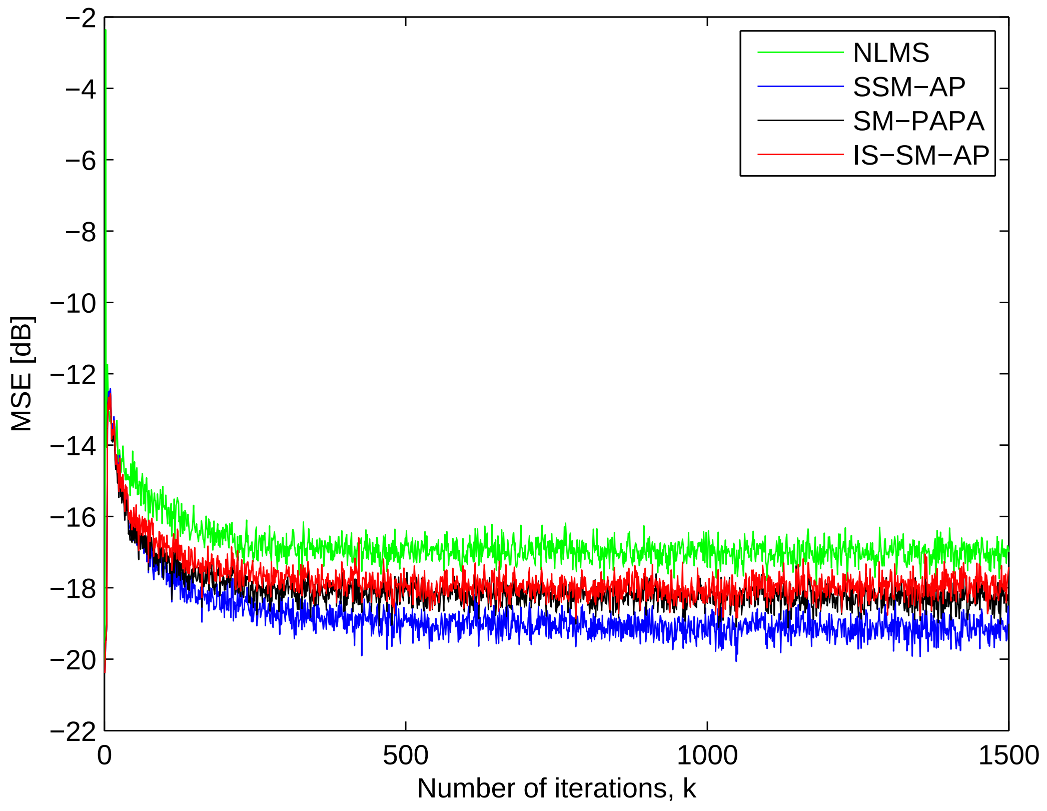


Figure 6.5: The learning curves of the SM-PAPA, the SSM-AP, the IS-SM-AP, and the NLMS algorithms applied on $\mathbf{w}_o$ using AR input signal.

advantage of requiring fewer computations since at each iteration in which an update occurs only a subset (on average, one third) of the coefficients is updated. Another interesting observation is that the SM-PAPA algorithm works better with BPSK input signal, whereas the SSM-AP algorithm is slightly better when a correlated input signal is used.

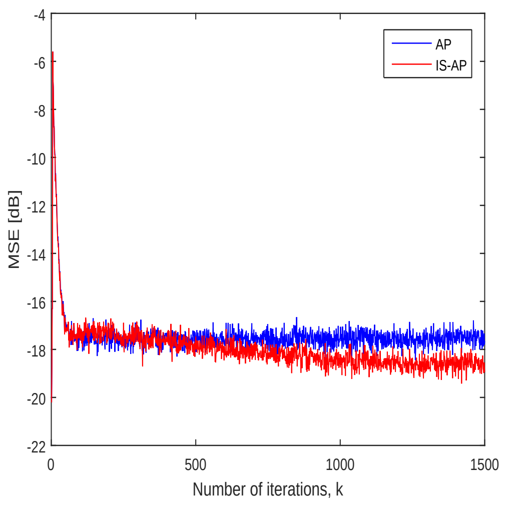
Scenario 2

In this scenario, we have applied the AP and the IS-AP algorithms to identify the three unknown sparse systems in Table 6.12. To identify $\mathbf{w}_o$ and $\mathbf{w}'_o$ we choose the convergence factor $\mu = 0.6$ and to identify $\mathbf{w}''_o$ we adopt $\mu = 0.1$. Figures 6.6(a), 6.6(b), and 6.6(c) show the learning curves for the AP and the IS-AP algorithms to identify the unknown systems $\mathbf{w}_o$, $\mathbf{w}'_o$, and $\mathbf{w}''_o$, respectively.

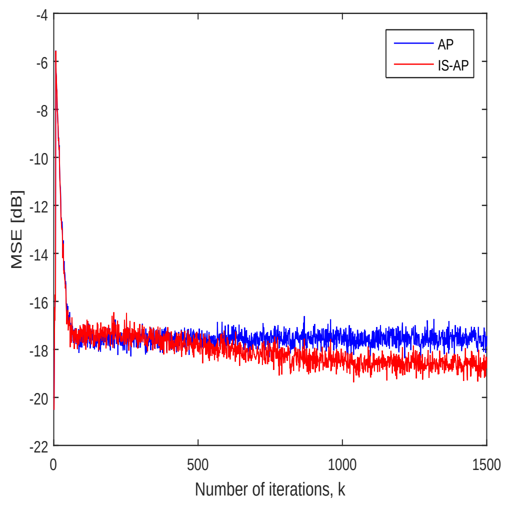
Moreover, we have applied the AP and the IS-AP algorithms in this scenario, with same parameters, but changing the input signal model to an autoregressive (AR) as

Table 6.13: The average number of updates implemented by the IS-SM-AP, the SM-PAPA, and the SSM-AP algorithms

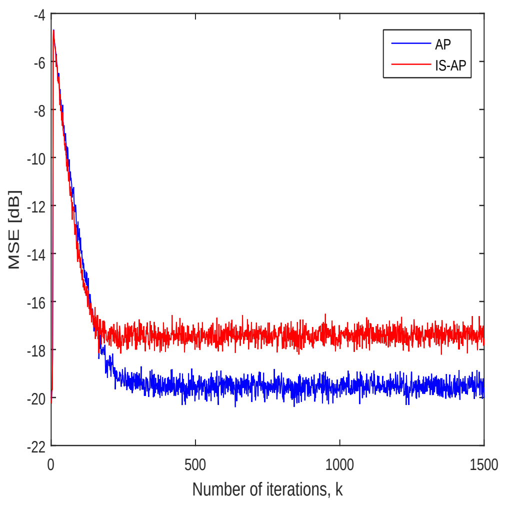
Algorithm	$\mathbf{w}_o$ BPSK input	$\mathbf{w}'_o$ BPSK input	$\mathbf{w}''_o$ BPSK input	$\mathbf{w}''_o$ AR input
IS-SM-AP	6.3%	6.3%	7.6%	8.4%
SM-PAPA	5.3%	5.3%	5.9%	7.7%
SSM-AP	8.9%	8.9%	20.5%	5.6%



(a)



(b)



(c)

Figure 6.6: The learning curves of the AP and the IS-AP algorithms applied on: (a) $\mathbf{w}_o$; (b) $\mathbf{w}'_o$; (c) $\mathbf{w}''_o$.

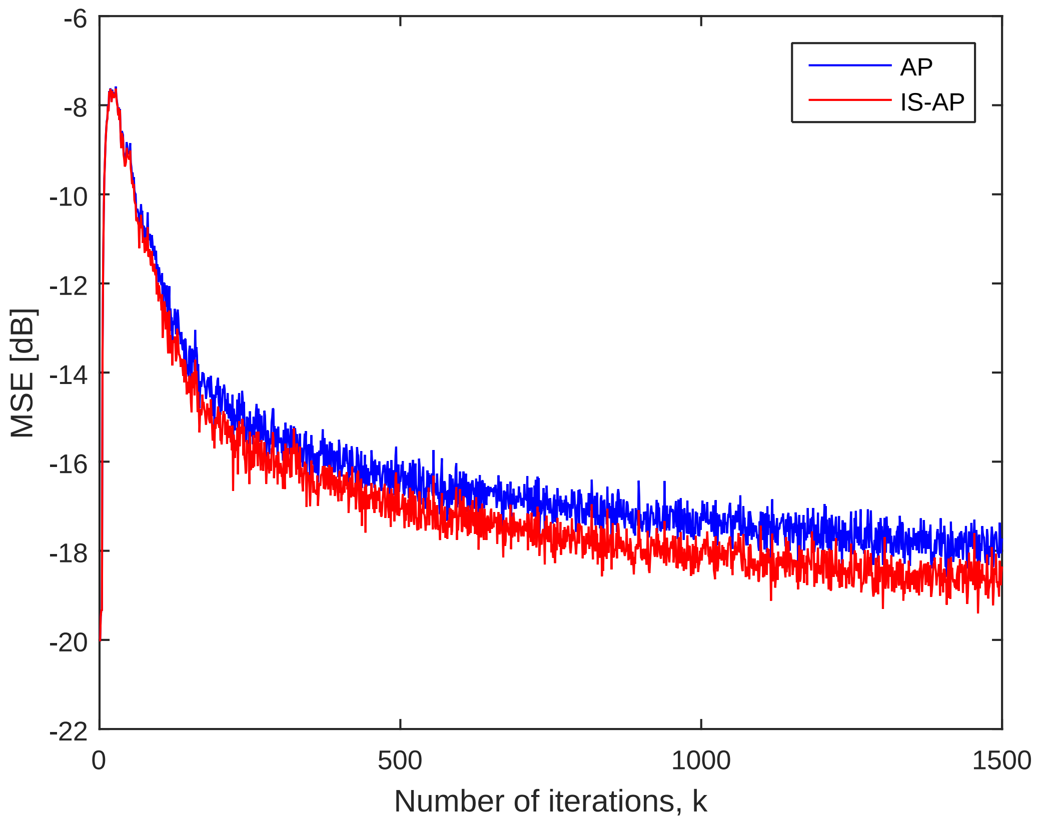
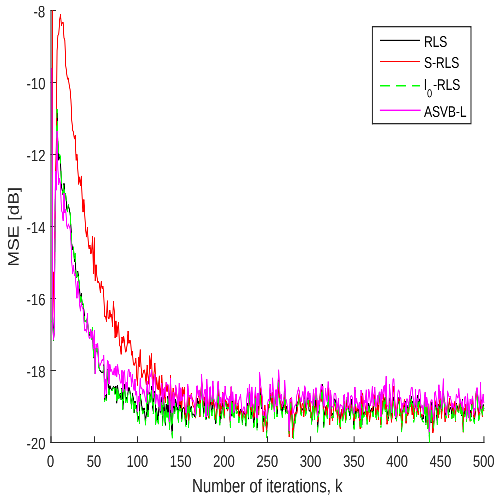


Figure 6.7: The learning curves of the AP and the IS-AP algorithms applied on $\mathbf{w}_o$ using AR input signal.

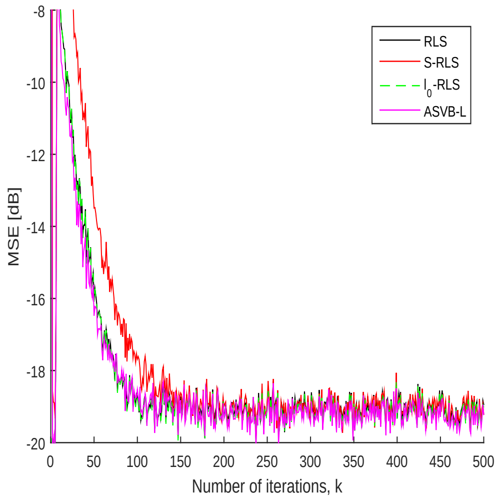
Scenario 1 to identify the unknown system $\mathbf{w}_o$. The convergence factor μ is equal to 0.6. Their learning curves are shown in Figure 6.7. By comparing Figures 6.5 and 6.7 we can observe the value of set-membership filtering. In fact, by utilizing the SMF approach not only we have a lower number of arithmetic operations, but also we improve the steady state performance. Note that, we have obtained better MSE in all figures of Scenario 1 compared to their corresponding figures in Scenario 2.

6.7.2 Simulation results of the RLS-based algorithms

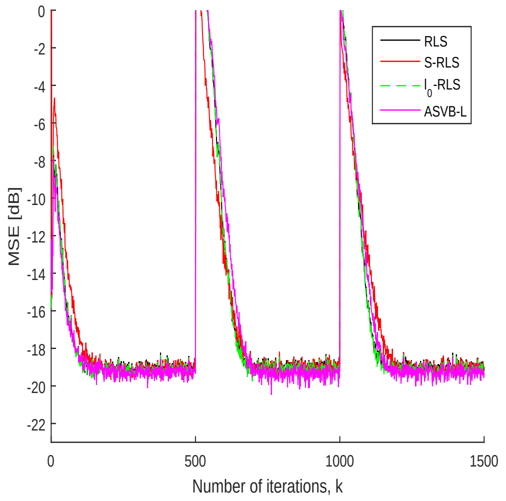
Here, the RLS, the S-RLS, the AS-RLS, the l_0-RLS, the A-l_0-RLS, the ASVB-L [115–117], the DS-S-RLS, the DS-l_0-RLS, and the data-selective ASVB-L (DS-ASVB-L) algorithms are tested to identify three unknown sparse systems of order 14. The first model is an arbitrary sparse system $\mathbf{w}_o$, the second model is a block sparse system $\mathbf{w}'_o$, and the third model, $\mathbf{w}'''_o$, is a sparse system which its coefficients changes at 500th and



(a)



(b)



(c)

Figure 6.8: The learning curves of the RLS, the S-RLS, the l_0-RLS, and the ASVB-L algorithms applied to identify: (a) $\mathbf{w}_o$; (b) $\mathbf{w}'_o$; (c) $\mathbf{w}'''_o$.

$1000th$ iterations. The coefficients of $\mathbf{w}_o$ and $\mathbf{w}'_o$ are listed in Table 6.12. The input is an autoregressive signal generated by $x(k) = 0.95x(k-1) + n(k-1)$. The signal-to-noise ratio (SNR) is set to be 20 dB, meaning that the noise variance is $\sigma_n^2 = 0.01$. The bound on the estimation error is set to be $\overline{\gamma} = \sqrt{5\sigma_n^2}$. The initial vector $\mathbf{w}(0)$ and λ are $[1, \cdots, 1]^T$ and 0.97, respectively. The parameter δ is 0.2 and the constant ϵ is chosen as 0.015. For the DS-l_0-RLS and the l_0-RLS algorithms, the parameters α and β are chosen as 0.005 and 5, respectively. We have chosen the GMF as the approximation of the l_0 norm. The depicted learning curves represent the results of

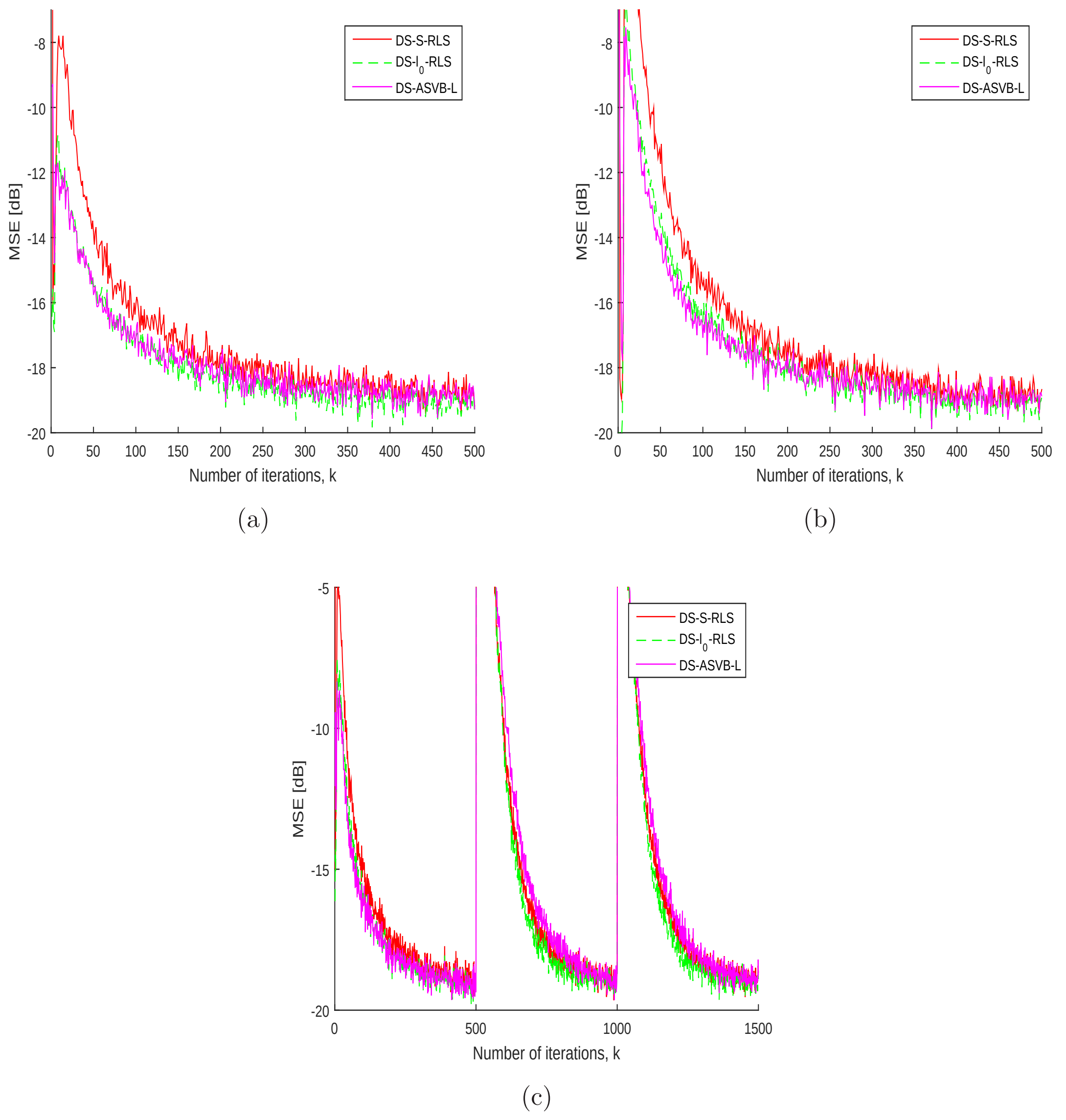


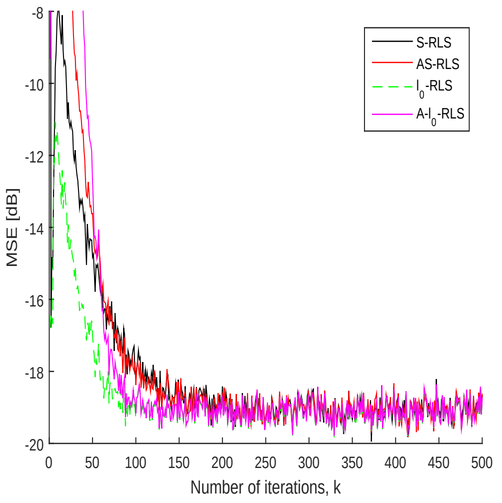
Figure 6.9: The learning curves of the DS-S-RLS, the DS-l_0-RLS, and the DS-ASVB-L algorithms applied to identify: (a) $\mathbf{w}_o$; (b) $\mathbf{w}'_o$; (c) $\mathbf{w}'''_o$.

averaging of the outcomes of 500 trials.

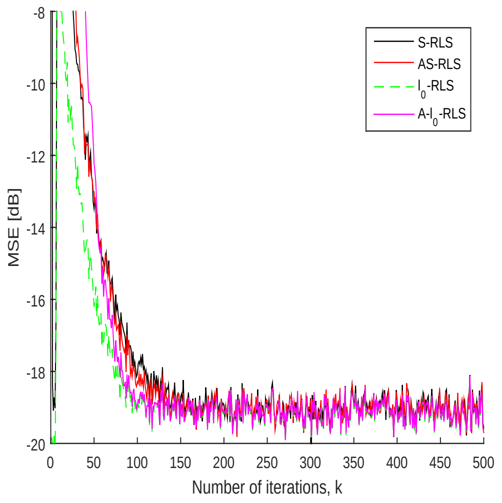
Figures 6.8(a), 6.8(b), and 6.8(c) show the learning curves for the RLS, the S-RLS, the l_0-RLS, and the ASVB-L algorithms to identify the unknown systems $\mathbf{w}_o$, $\mathbf{w}'_o$, and $\mathbf{w}'''_o$, respectively. Figures 6.9(a), 6.9(b), and 6.9(c) illustrate the learning curves for the DS-S-RLS, the DS-l_0-RLS, and the DS-ASVB-L algorithms to identify the unknown systems $\mathbf{w}_o$, $\mathbf{w}'_o$, and $\mathbf{w}'''_o$, respectively. The average number of updates implemented by the DS-S-RLS, the DS-l_0-RLS, and the DS-ASVB-L algorithms are presented in columns 2 to 4 of Table 6.14.

Table 6.14: The average number of updates implemented by the DS-S-RLS, the DS-l_0-RLS, and the DS-ASVB-L algorithms

Algorithm	$\mathbf{w}_o$	$\mathbf{w}'_o$	$\mathbf{w}'''_o$
DS-S-RLS	11.95%	14.13%	19.40%
DS-l_0-RLS	8.72%	10.90%	17.74%
DS-ASVB-L	9.18%	10.53%	19.69%



(a)



(b)

Figure 6.10: The learning curves of the S-RLS, the AS-RLS, the l_0-RLS, and the A-l_0-RLS algorithms applied to identify: (a) $\mathbf{w}_o$; (b) $\mathbf{w}'_o$.

Observe that, in every scenario we tested, the S-RLS and the l_0-RLS algorithms performed as well as the RLS algorithm. The S-RLS algorithm has lower computational complexity compared to the l_0-RLS algorithm. As can be seen, the performances of the S-RLS and the DS-S-RLS algorithms are close to the ASVB-L and the DS-ASVB-L algorithms, respectively, while the former ones require lower computational resources.

Finally, Figures 6.10(a) and 6.10(b) depict the learning curves of the S-RLS, the AS-RLS, the l_0-RLS, and the A-l_0-RLS algorithms, when they are applied to identify the unknown systems $\mathbf{w}_o$ and $\mathbf{w}'_o$, respectively. As can be seen, the performances of the AS-RLS and the A-l_0-RLS algorithms are similar to the S-RLS and the l_0-RLS algorithms, respectively.

6.8 Conclusions

In this chapter, we have proposed the S-SM-AP and the IS-SM-AP algorithms to take advantage of sparsity in the signal models while attaining low computational complexity. To reach this target, we have derived a simple update equation which only updates the filter coefficients whose magnitudes are greater than a predetermined value. Also, this method is jointly applied with the well-known set-membership approach aiming at obtaining even lower computational complexity and better convergence rate. The simulation results have shown the excellent performance of the algorithm and lower computational complexity as compared to some other sparsity-aware data-selective adaptive filters. Indeed, the IS-SM-AP algorithm performed as well as the SM-PAPA algorithm while requiring fewer arithmetic operations (for the scenarios in Section 6.7, it entailed about 38% of the operations spent by the SM-PAPA). Also, the numerical results in Section 6.7 confirm the importance of SMF technique for the proposed algorithm.

Moreover, we have used the discard function and the l_0 norm in order to propose the S-RLS and the l_0-RLS algorithms, respectively, to exploit the sparsity in the involved signal models. Also, we have employed the data-selective strategy to implement an update when the output estimation error is greater than a pre-described positive value leading to reduced update rate and lower computational complexity. The simulation results have shown the excellent performance of the proposed algorithms as compared to the standard RLS algorithm being competitive with the new proposed state-of-the-art ASVB-L algorithm which requires much more computations. It is worthy to mention that there are many RLS-based algorithms to exploit sparsity in signal and system models [118–120]; however, their update equation is entirely different from the algorithms proposed in this chapter. Therefore, we avoid comparing the RLS-based algorithms proposed here with other RLS-based algorithms in the literature.

Chapter 7

Feature LMS algorithms

Among the adaptive filtering algorithms, the popular least-mean-square (LMS) algorithm, first introduced in 1960 [121, 122], has been widely considered as the most used in the field. Elaborate studies of the LMS algorithm were presented in [2, 123]. Also, the LMS and its variants solve real problems including active noise control [124], digital equalization [125], continuous-time filter tuning [126], system identification [127], among others.

In the previous chapter, some adaptive filtering algorithms exploiting the sparsity in the system parameters were proposed. Also, a number of adaptive filtering algorithms exploiting the sparsity in the model coefficients has been introduced by imposing some constraints in the cost function [8, 100, 128, 129]. This strategy relies on the attraction of some coefficient values to zero enabling the detection of nonrelevant parameters of the model.

In this chapter, we introduce the feature LMS (F-LMS) family of algorithms inducing simple sparsity properties hidden in the parameters. The type of feature to seek determines the structure of the feature matrix $\mathbf{F}(k)$ to be applied in the constraints of the F-LMS algorithm. In fact, a plethora of featured algorithms is possible to be defined by applying smart combinations of feature matrices to the coefficient vector. In this work, some simple cases are discussed whereas many more advanced solutions will be exploited in future publications. Moreover, by introducing *feature function*, we propose the low-complexity F-LMS (LCF-LMS) algorithm to reduce the computational complexity of the F-LMS algorithms. The LCF-LMS algorithm implements less multiplication in calculating the output signal.

The content of this chapter was partially published in [130]. This chapter is organized as follows. Section 7.1 proposes the F-LMS family of algorithms. Some examples of F-LMS algorithms for systems with lowpass and highpass spectrum are introduced in Section 7.2. The LCF-LMS and the alternative LCF-LMS (ALCF-LMS) algorithms are derived in Sections 7.3 and 7.4, respectively. The matrix representation of the feature function is explained in Section 7.5. Simulation results are presented in Section 7.6 and the conclusions are drawn in Section 7.7.

7.1 The Feature LMS algorithms

Feature LMS (F-LMS) refers to a family of LMS-type algorithms capable of exploiting the features inherent to the unknown systems to be identified. These algorithms minimize the general objective function

$$\xi_{\text{F-LMS}}(k) = \underbrace{\frac{1}{2}|e(k)|^2}_{\text{standard LMS term}} + \underbrace{\alpha\mathcal{P}\left(\mathbf{F}(k)\mathbf{w}(k)\right)}_{\text{feature-inducing term}}, \tag{7.1}$$

where $\alpha \in \mathbb{R}_+$ stands for the weight given to the *sparsity-promoting penalty function* $\mathcal{P}$, which maps a vector to the nonnegative reals $\mathbb{R}_+$, and $\mathbf{F}(k)$ is the so-called *feature matrix* responsible for revealing the hidden sparsity, i.e., the result of applying $\mathbf{F}(k)$ to $\mathbf{w}(k)$ should be a sparse vector (in the sense that most entries of the vector $\mathbf{F}(k)\mathbf{w}(k)$ should be close or equal to zero).

The penalty function $\mathcal{P}$ can be any sparsity-promoting penalty function that is almost everywhere differentiable in order to allow for gradient-based methods. Examples of suitable functions are: (i) vector norms, especially the widely used l_1 norm [100, 128]; (ii) vector norms combined with shrinking strategies [109]; (iii) a function that approximates the l_0 norm [8, 105].

The feature matrix $\mathbf{F}(k)$ can vary at each iteration and it represents any linear combination that when applied to $\mathbf{w}(k)$ results in a sparse vector. In practice, $\mathbf{F}(k)$ should be chosen based on some previous knowledge about the unknown system $\mathbf{w}_o$. For instance, $\mathbf{w}_o$ can represent a lowpass or a highpass filter, it can have linear phase, it can be an upsampled or downsampled signal, etc. All these features can be exploited by the F-LMS algorithm in order to accelerate convergence and/or achieve lower mean-squared error (MSE).

The resulting gradient-based algorithms using the objective function given in (7.1) are known as F-LMS algorithms, and their recursions have the general form

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}(k) - \mu\alpha\mathbf{p}(k), \tag{7.2}$$

where $\mu \in \mathbb{R}_+$ is the step size, which should be small enough to ensure convergence [2], and $\mathbf{p}(k) \in \mathbb{R}^{N+1}$ is the gradient of function $\mathcal{P}\left(\mathbf{F}(k)\mathbf{w}(k)\right)$.

7.2 Examples of F-LMS algorithms

From Section 7.1, it is clear that the F-LMS family contains infinitely many algorithms. So, in this section we introduce some of these algorithms in order to illustrate how some specific features of the unknown system can be exploited. For the sake of clarity, we focus on simple algorithms and, therefore, we choose function $\mathcal{P}$ to be the l_1 norm and the feature matrix to be time-invariant $\mathbf{F}$ so that the cost function in (7.1) simplifies to

$$\xi_{\text{F-LMS}}(k) = \frac{1}{2}|e(k)|^2 + \alpha\|\mathbf{F}\mathbf{w}(k)\|_1, \tag{7.3}$$

where $\|\cdot\|_1$ denotes the l_1-norm and for a vector $\mathbf{w} \in \mathbb{R}^{N+1}$ it is given by $\|\mathbf{w}\|_1 = \sum_{i=0}^{N}|w_i|$. As a consequence, the reader will notice that the computational complexity of the algorithms proposed in this section is only slightly superior to the complexity of the LMS algorithm, as the computation of $\mathbf{p}(k)$ required in (7.2) is very simple (does not involve multiplication or division).

7.2.1 The F-LMS algorithm for lowpass systems

Most systems found in practice have their energy concentrated mainly in the low frequencies. If the unknown system has lowpass narrowband spectrum, then its impulse response $\mathbf{w}_o$ is smooth, meaning that the difference between adjacent coefficients is small (probably close to zero).

The adaptive filtering algorithm can take advantage of this feature present in the unknown system by selecting the feature matrix properly. Indeed, by selecting $\mathbf{F}$ as

$\mathbf{F}_l$, where $\mathbf{F}_l$ is a $N \times N+1$ matrix defined as

$$\mathbf{F}_l = \begin{bmatrix} 1 & -1 & 0 & \cdots & 0 \\ 0 & 1 & -1 & \cdots & 0 \\ \vdots & & \ddots & \ddots & \\ 0 & 0 & \cdots & 1 & -1 \end{bmatrix}, \tag{7.4}$$

and $\|\mathbf{F}_l \mathbf{w}(k)\|_1 = \sum_{i=0}^{N-1} |w_i(k) - w_{i+1}(k)|$, the optimization problem in (7.3) can be interpreted as: we seek for $\mathbf{w}(k)$ that minimizes both the squared error (LMS term) and the distances between adjacent coefficients of $\mathbf{w}(k)$. In other words, the F-LMS algorithm for lowpass systems acts like the LMS algorithm, but enforcing $\mathbf{w}(k)$ to be a lowpass system. It is worth mentioning that if $\mathbf{w}_o$ is indeed a lowpass system, then matrix $\mathbf{F}_l$ yields a sparse vector $\mathbf{F}_l\mathbf{w}(k)$.[1]

Thus, the F-LMS algorithm for lowpass systems is defined by the recursion given in (7.2), but replacing vector $\mathbf{p}(k)$ with $\mathbf{p}_l(k)$ defined as

$$\begin{cases} p_{l,i}(k) = \text{sgn}(w_0(k) - w_1(k)) & \text{if } i = 0, \\ p_{l,i}(k) = -\text{sgn}(w_{i-1}(k) - w_i(k)) + \text{sgn}(w_i(k) - w_{i+1}(k)) & \text{if } i = 1, \cdots, N-1, \\ p_{l,i}(k) = -\text{sgn}(w_{N-1}(k) - w_N(k)) & \text{if } i = N, \end{cases} \tag{7.5}$$

where sgn$(\cdot)$ denotes the sign function.

As previously explained, the F-LMS algorithm above tries to reduce the distances between consecutive coefficients of $\mathbf{w}(k)$, i.e., matrix $\mathbf{F}_l$ can be understood as the process of windowing $\mathbf{w}(k)$ with a window of length 2 (i.e., two coefficients are considered at a time). We can increase the window length, in order to make a smoothing considering more coefficients simultaneously, by nesting linear combinations as follows

$$\mathbf{F}_l^{M-\text{nested}} = \prod_{m=1}^{M} \mathbf{F}_l^{(m)} \mathbf{F}_l, \tag{7.6}$$

where $\mathbf{F}_l^{(m)}$ has the same structure given in (7.4), but losing m rows and m columns in relation to the dimensions of $\mathbf{F}_l$.

In addition to the previous examples, suppose that the unknown system is the

[1]A matrix similar to the $\mathbf{F}_l$ in (7.4) is already known by the statisticians working on a field called *trend filtering* [131].

result of upsampling a lowpass system by a factor of L. In this case, we should use matrix $\mathbf{F}_l^*$, whose rows have $L-1$ zeros between the ± 1 entries, in (7.3). For $L = 2$, we have the following matrix

$$\mathbf{F}_l^* = \begin{bmatrix} 1 & 0 & -1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & -1 & \cdots & 0 \\ \vdots & & \ddots & \ddots & \ddots & \\ 0 & 0 & \cdots & 1 & 0 & -1 \end{bmatrix}, \tag{7.7}$$

and $\|\mathbf{F}_l^*\mathbf{w}(k)\|_1 = \sum_{i=0}^{N-2} |w_i(k) - w_{i+2}(k)|$.

Next the F-LMS algorithm using such $\mathbf{F}_l^*$ has the update rule given in (7.2), but replacing $\mathbf{p}(k)$ with $\mathbf{p}_l^*(k)$ defined as

$$\begin{cases} p_{l,i}^*(k) = \text{sgn}(w_i(k) - w_{i+2}(k)) & \text{if } i = 0, 1, \\ p_{l,i}^*(k) = -\text{sgn}(w_{i-2}(k) - w_i(k)) + \text{sgn}(w_i(k) - w_{i+2}(k)) & \text{if } i = 2, \cdots, N-2, \\ p_{l,i}^*(k) = -\text{sgn}(w_{i-2}(k) - w_i(k)) & \text{if } i = N-1, N. \end{cases} \tag{7.8}$$

7.2.2 The F-LMS algorithm for highpass systems

If the unknown system $\mathbf{w}_o$ has a highpass narrowband spectrum, then adjacent coefficients tend to have similar absolute values, but with opposite signs. Therefore, the sum of two consecutive coefficients is close to zero and we can exploit this feature in the learning process by minimizing the sum of adjacent coefficients of $\mathbf{w}(k)$. This can be accomplished by selecting $\mathbf{F}$ as $\mathbf{F}_h$, where $\mathbf{F}_h$ is an $N \times N+1$ feature matrix defined as

$$\mathbf{F}_h = \begin{bmatrix} 1 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 1 & \cdots & 0 \\ \vdots & & \ddots & \ddots & \\ 0 & 0 & \cdots & 1 & 1 \end{bmatrix}, \tag{7.9}$$

such that $\|\mathbf{F}_h\mathbf{w}(k)\|_1 = \sum_{i=0}^{N-1} |w_i(k) + w_{i+1}(k)|$.

The F-LMS algorithm for highpass systems is characterized by the recursion given in (7.2), but replacing $\mathbf{p}(k)$ with $\mathbf{p}_h(k)$, which is defined as

$$\begin{cases} p_{h,i}(k) = \mathrm{sgn}(w_0(k) + w_1(k)) & \text{if } i = 0, \\ p_{h,i}(k) = \mathrm{sgn}(w_{i-1}(k) + w_i(k)) + \mathrm{sgn}(w_i(k) + w_{i+1}(k)) & \text{if } i = 1, \cdots, N-1, \\ p_{h,i}(k) = \mathrm{sgn}(w_{N-1}(k) + w_N(k)) & \text{if } i = N. \end{cases} \tag{7.10}$$

Similar to the lowpass case, let us consider that the unknown system is the result of interpolating a highpass system by a factor $L = 2$. The set of interpolated highpass systems leads to a notch filter with zeros at $z = \pm \jmath$. In this case, we can utilize $\mathbf{F}_h^*$ in the objective function (7.3), where $\mathbf{F}_h^*$ is described by

$$\mathbf{F}_h^* = \begin{bmatrix} 1 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 1 & \cdots & 0 \\ \vdots & & \ddots & \ddots & \ddots & \\ 0 & 0 & \cdots & 1 & 0 & 1 \end{bmatrix}, \tag{7.11}$$

and $\|\mathbf{F}_h^* \mathbf{w}(k)\|_1 = \sum_{i=0}^{N-2} |w_i(k) + w_{i+2}(k)|$.

Using $\mathbf{F}_h^*$, the F-LMS recursion in (7.2) should substitute $\mathbf{p}(k)$ by $\mathbf{p}_h^*(k)$ defined as

$$\begin{cases} p_{h,i}^*(k) = \mathrm{sgn}(w_i(k) + w_{i+2}(k)) & \text{if } i = 0, 1, \\ p_{h,i}^*(k) = \mathrm{sgn}(w_{i-2}(k) + w_i(k)) + \mathrm{sgn}(w_i(k) + w_{i+2}(k)) & \text{if } i = 2, \cdots, N-2, \\ p_{h,i}^*(k) = \mathrm{sgn}(w_{i-2}(k) + w_i(k)) & \text{if } i = N-1, N. \end{cases} \tag{7.12}$$

7.3 Low-complexity F-LMS Algorithms

In this section, we derive the low-complexity feature LMS (LCF-LMS) algorithm to exploit sparsity in the linear combination of the parameters, as the F-LMS algorithms do, while also reducing the computational cost of calculating the output signal.

Here, the idea is to reduce the number of multiplications required for computing the output signal when there is a strong relation between neighboring coefficients. In systems with lowpass frequency content, for example, neighboring coefficients vary smoothly. Therefore, when the input signal is highly correlated, we can fix the value of the neighboring coefficients where the distances (the absolute value of their differences) between any two consecutive coefficients are less than a small constant $\epsilon > 0$. As a result, we reduce the number of multiplications in the calculation of $y(k) \triangleq \mathbf{w}^T(k)\mathbf{x}(k)$. For instance, if for nonnegative integers m and j, where $m, j < N$, the discrepancies between the coefficients with indexes m to $m+j$ are less than ϵ, then we can use the mth coefficient as a reference. Mathematically, if the value of $|w_{m+i+1}(k)-w_{m+i}(k)| \leq \epsilon$ for $i = 0, 1, 2, \cdots, j-1$, then in the calculation of the output signal instead of computing

$$y(k) = w_m(k)x_m(k) + \cdots + w_{m+j}(k)x_{m+j}(k), \tag{7.13}$$

we can approximate $y(k)$ as

$$\hat{y}(k) \triangleq \underbrace{w_m(k)x_m(k) + \cdots + w_m(k)x_m(k)}_{(j+1)-\text{times}}. \tag{7.14}$$

As a result, we decrease the number of multiplications from $j+1$ to one. Hence, for a block of coefficients in which the distance between any two consecutive coefficients is less than ϵ, we can use the first parameter of the block as the reference parameter. As soon as the distance between two consecutive coefficients becomes greater than ϵ, we will use the new one as a reference for the new block of coefficients.

To this end, for each block of coefficients in which the distance of any two consecutive coefficients is less than ϵ, we have to preserve the first coefficient of the block, and the rest of them will be replaced by zero. Furthermore, when the absolute value of a coefficient is less than ϵ, we can replace it with zero to avoid additional multiplication [109, 110]. Therefore, two subsets of parameters will be replaced by zero: (I) the coefficients whose absolute values are less than ϵ, and (II) the coefficients whose distances from their antecessor are less than ϵ.

The above reasoning can be implemented by means of the *feature function*, $\mathbb{F}_\epsilon : \mathbb{R}^{N+1} \to \mathbb{R}^{N+1}$, applied to the weigh vector of the adaptive filter. The ith element of the feature function, for $i = 0, 1, \cdots, N$, is defined as

$$\mathbb{F}_{\epsilon,i}(\mathbf{w}(k)) \triangleq \begin{cases} f_\epsilon(w_0(k)) & \text{if } i = 0, \\ f_\epsilon(w_i(k)) & \text{if } |w_i(k) - w_{i-1}(k)| > \epsilon \ \& \ i \neq 0, \\ 0 & \text{if } |w_i(k) - w_{i-1}(k)| \leq \epsilon \ \& \ i \neq 0, \end{cases} \tag{7.15}$$

where f_ϵ is the discard function defined in (6.12). As can be observed, the feature function replaces the subsets (I) and (II) of the coefficients of $\mathbf{w}(k)$ with zero. Let us define $\mathbf{w}_s(k) \triangleq \mathbb{F}_\epsilon(\mathbf{w}(k))$. Figure 7.1 shows an example for the impulse response

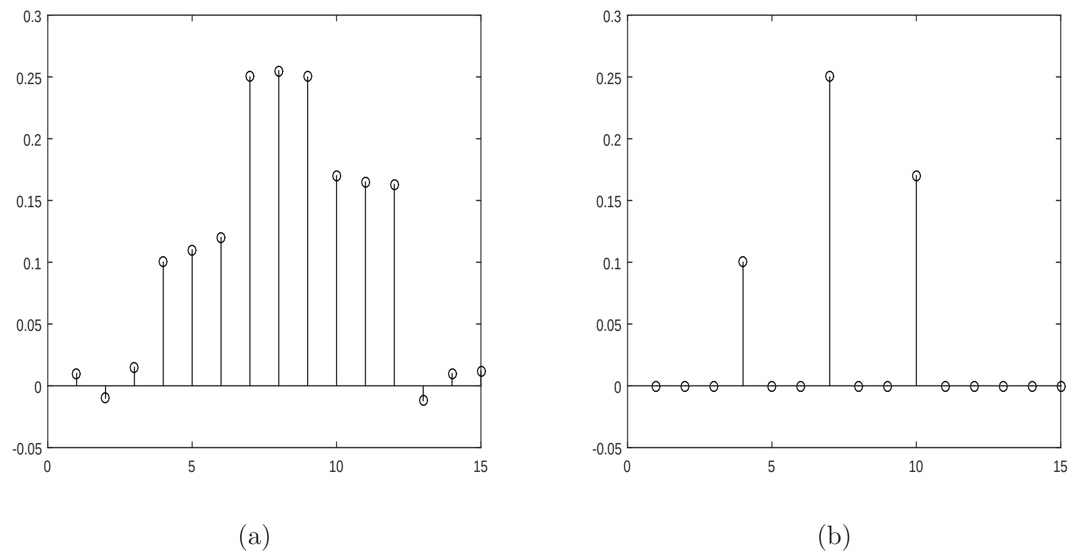


Figure 7.1: The impulse response of (a) $\mathbf{w}(k)$; (b) $\mathbf{w}_s(k) = \mathbb{F}_\epsilon(\mathbf{w}(k))$ for $\epsilon = 0.02$.

of $\mathbf{w}(k)$ and $\mathbf{w}_s(k)$ when $\epsilon = 0.02$. As can be observed, $\mathbf{w}(k)$ has fifteen nonzero coefficients, and after using the feature function twelve of them are replaced by zero.

Our goal is to utilize $\mathbf{w}_s(k) = \mathbb{F}_\epsilon(\mathbf{w}(k))$ in the calculation of the output signal. However, we must determine from which subset of coefficients of $\mathbf{w}(k)$ the zero elements of $\mathbf{w}_s(k)$ came, i.e., subsets (I) or (II). In fact, for some i, $w_{s_i}(k)$ is zero if and only if $w_i(k)$ belongs to the subsets (I) or (II). If $w_i(k)$ belongs to the subset (I), then we can directly apply $w_{s_i}(k)$ to calculate the output signal, i.e., we use $w_{s_i}(k)x_i(k) = 0$. However, if $w_i(k)$ belongs to the subset (II), then we must apply the last nonzero coefficient of $\mathbf{w}_s(k)$ before the ith index to compute the output signal. Assume that this nonzero coefficient has index m, then we use $w_{s_m}(k)$ instead of $w_i(k)$ since their values are close to each other. Hence, in the calculation of the output signal, we use $w_{s_m}(k)x_m(k)$ instead of $w_{s_i}(k)x_i(k)$.

In order to determine the background of the zero coefficients in $\mathbf{w}_s(k)$, we define a binary vector $\mathbf{b}(k) \in \{0,1\}^{N+1}$ as $\mathbf{b}(k) = \mathbf{f}_\epsilon(\mathbf{w}(k))$, where $\mathbf{f}_\epsilon$ is the discard vector function. Then, for some i, if $w_{s_i}(k)$ and $b_i(k)$ are zero, we infer that $w_i(k)$ belongs to the subset (I). However, if $w_{s_i}(k) = 0$ and $b_i(k) = 1$, then we conclude that $w_i(k)$ belongs to the subset (II).

Finally, we can present the LCF-LMS algorithm in Table 7.1. This algorithm implements less multiplication as compared to the LMS algorithm.

As mentioned earlier, for proposing the LCF-LMS algorithm, we assumed that the input signal is highly correlated. This assumption restricts the use of the LCF-LMS

Table 7.1: Low-complexity feature LMS algorithm

LCF-LMS Algorithm

```
Initialization
$\mathbf{w}_s(0) = \mathbf{b}(0) = \mathbf{w}(0) = [0 \ \cdots \ 0]^T$
choose $\mu$ in the range $0 < \mu \ll 1$
choose small constant $\epsilon > 0$
Do for $k \geq 0$
  temp = 0, $y(k) = 0$
  for $i = 0$ to $N$
    if $w_{s_i}(k) \neq 0$
      temp $= w_{s_i}(k)x_i(k)$
      $y(k) = y(k) +$ temp
    else
      $y(k) = y(k) + ($temp $\times b_i(k))$
    end
  end
  $e(k) = d(k) - y(k)$
  $\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}(k)$
  $\mathbf{w}_s(k+1) = \mathbb{F}_\epsilon(\mathbf{w}(k+1))$
  $\mathbf{b}(k+1) = \mathbf{f}_\epsilon(\mathbf{w}(k+1))$
end
```

algorithm. To avoid this assumption, instead of approximating $y(k)$ by (7.14), we can approximate $y(k)$ as

$$\hat{y}(k) \triangleq w_m(k)(x_m(k) + x_{m+1}(k) + \cdots + x_{m+j}(k)). \tag{7.16}$$

In other words, when $w_m(k)$ represents a block of coefficients of length $j+1$, the LCF-LMS algorithm sums $j+1$ copies of $w_m(k)x_m(k)$; however, in Equation (7.16), we multiply $w_m(k)$ by the sum of the input signal components corresponding to the coefficients represented by $w_m(k)$. Note that the number of required arithmetic operations in (7.16) and (7.14) are identical; i.e., both equations implement one multiplication and j additions. The algorithm using Equation (7.16) in calculating output signal is called the improved LCF-LMS (I-LCF-LMS) algorithm, and its application is not limited to cases with correlated input signals. The I-LCF-LMS algorithm is presented in Table 7.2.

Table 7.2: Improved low-complexity feature LMS algorithm

I-LCF-LMS Algorithm

```
Initialization
$\mathbf{w}_s(0) = \mathbf{b}(0) = \mathbf{w}(0) = [0 \ \cdots \ 0]^T$
choose $\mu$ in the range $0 < \mu \ll 1$
choose small constant $\epsilon > 0$
Do for $k \geq 0$
  $\text{temp}_x = 0$, $\text{temp}_w = 0$, $y(k) = 0$
  for $i = 0$ to $N$
   if $w_{s_i}(k) \neq 0$
    $y(k) = y(k) + (\text{temp}_w \times \text{temp}_x)$
    $\text{temp}_w = w_{s_i}(k)$
    $\text{temp}_x = x_i(k)$
   else
    $\text{temp}_x = \text{temp}_x + (x_i(k) \times b_i(k))$
   end
  end
  $y(k) = y(k) + (\text{temp}_w \times \text{temp}_x)$
  $e(k) = d(k) - y(k)$
  $\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}(k)$
  $\mathbf{w}_s(k+1) = \mathbb{F}_\epsilon(\mathbf{w}(k+1))$
  $\mathbf{b}(k+1) = \mathbf{f}_\epsilon(\mathbf{w}(k+1))$
end
```

7.4 Alternative LCF-LMS Algorithm

In the LCF-LMS algorithm, when $\mathbf{w}(k)$ contains a long sequence of coefficients with almost similar absolute values, then $\mathbf{w}_s(k)$ contains a long block of zeros. Therefore, when calculating the output signal, all parameters of this block are represented by the first element of the block. As a result, since we are using a fixed coefficient to represent many ones, we could have an accumulated error. In this section, we introduce the alternative LCF-LMS (ALCF-LMS) algorithm to address this problem.

To avoid accumulated error because of many adjacent zeros in $\mathbf{w}_s(k)$, for some natural number $p < N$, we can force the feature function to keep every p coefficients of $\mathbf{w}(k)$ in $\mathbf{w}_s(k)$ if the absolute value of the coefficient is greater than ϵ. In other words, no parameter can represent a block of coefficients with more than p elements. The only exception is the case when the parameters of the block have absolute values smaller than ϵ (i.e., they are really close to zero; therefore, they must be replaced by zero). Let us denote by $\mathbb{F}^a_\epsilon : \mathbb{R}^{N+1} \rightarrow \mathbb{R}^{N+1}$ the new feature function, and it is called the *alternative feature function*. The ith element of $\mathbb{F}_\epsilon^a$, for $i = 0, 1, \cdots, N$, is defined by

$$\mathbb{F}_{\epsilon,i}^a(\mathbf{w}(k)) \triangleq \begin{cases} f_\epsilon(w_i(k)) & \text{if } \mathrm{mod}(i,p) = 0, \\ f_\epsilon(w_i(k)) & \text{if } |w_i(k) - w_{i-1}(k)| > \epsilon \ \& \ \mathrm{mod}(i,p) \neq 0, \\ 0 & \text{if } |w_i(k) - w_{i-1}(k)| \leq \epsilon \ \& \ \mathrm{mod}(i,p) \neq 0, \end{cases} \tag{7.17}$$

where $\mathrm{mod}(i,p)$ stands for the remainder of $\frac{i}{p}$. Therefore, the ALCF-LMS algorithm is similar to the LCF-LMS one in Table 7.1, but the feature function is replaced by the alternative feature function (i.e., $\mathbf{w}_s(k+1) = \mathbb{F}_\epsilon^a(\mathbf{w}(k+1))$).

By using the same argument, we can propose the alternative I-LCF-LMS (AI-LCF-LMS) algorithm. Indeed, if we replace the feature function in Table 7.2 with the alternative feature function, then we obtain the AI-LCF-LMS algorithm.

7.5 Matrix Representation of the Feature Function

In this section, we show how to generate $\mathbf{w}_s(k)$ through matrix operations. Indeed, presenting $\mathbf{w}_s(k)$ through matrix operations is helpful for future mathematical analysis.

To generate $\mathbf{w}_s(k)$, we use quantization matrices $\mathbf{Q}_t(k)$ for $t = 1, 2, 3$, and two feature matrices $\mathbf{F}_1$ and $\mathbf{F}_2(k)$, all matrices belong to $\mathbb{R}^{(N+1)\times(N+1)}$. The matrices $\mathbf{F}_1$ and $\mathbf{F}_2(k)$ are responsible for exploiting the sparsity in the linear combination of the parameters and reconstructing the weight vector after exploiting the sparsity, respectively. Therefore, to exploit the hidden sparsity in the parameters of $\mathbf{w}(k)$ and their linear combinations, we introduce $\mathbf{w}_s(k)$ as follows

$$\mathbf{w}_s(k) \triangleq \mathbf{Q}_3(k)\mathbf{F}_2(k)\mathbf{Q}_2(k)\mathbf{F}_1\mathbf{Q}_1(k)\mathbf{w}(k). \tag{7.18}$$

In the following, we describe the matrices and justify their actions. We define the quantization matrix $\mathbf{Q}_1(k)$ as the Jacobian matrix of $\mathbf{f}_\epsilon(\mathbf{w}(k))$. Therefore, $\mathbf{Q}_1(k)$ is a diagonal matrix whose entries are zero or one. For the coefficients of $\mathbf{w}(k)$ where their absolute values are less than ϵ, the corresponding entries on the diagonal of $\mathbf{Q}_1(k)$ are zero, otherwise they are one. Similarly, the matrices $\mathbf{Q}_2(k)$ and $\mathbf{Q}_3(k)$ are defined as the Jacobian matrices of $\mathbf{f}_\epsilon(\mathbf{F}_1\mathbf{Q}_1(k)\mathbf{w}(k))$ and $\mathbf{f}_\epsilon(\mathbf{F}_2(k)\mathbf{Q}_2(k)\mathbf{F}_1\mathbf{Q}_1(k)\mathbf{w}(k))$, respectively. Thus $\mathbf{Q}_2(k)$ is a diagonal matrix with zero and one. Its diagonal entries are zero (one) for the corresponding elements of $\mathbf{F}_1\mathbf{Q}_1(k)\mathbf{w}(k)$ with the absolute value lower (greater) than ϵ. Also, $\mathbf{Q}_3(k)$ is a diagonal matrix similar to $\mathbf{Q}_2(k)$; however, it is derived from the vector $\mathbf{F}_2(k)\mathbf{Q}_2(k)\mathbf{F}_1\mathbf{Q}_1(k)\mathbf{w}(k)$. The diagonal entries of $\mathbf{Q}_3(k)$ are one for the corresponding elements of $\mathbf{F}_2(k)\mathbf{Q}_2(k)\mathbf{F}_1\mathbf{Q}_1(k)\mathbf{w}(k)$ with absolute value greater than ϵ, and zero for the others.

The feature matrix $\mathbf{F}_1$ has to find the difference between the coefficients of the vector $\mathbf{Q}_1(k)\mathbf{w}(k)$. In fact, it keeps the first parameter unchanged, and for other coefficients replaces them with the differences between them and the previous one. Thus, it can be represented as

$$\mathbf{F}_1 \triangleq \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ -1 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 1 & 0 & \cdots & 0 \\ \vdots & 0 & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & -1 & 1 & 0 \\ 0 & 0 & \cdots & 0 & -1 & 1 \end{bmatrix}. \tag{7.19}$$

The function of the feature matrix $\mathbf{F}_2(k)$ is to reconstruct the weight vector from the vector $\mathbf{r}(k) \triangleq \mathbf{Q}_2(k)\mathbf{F}_1\mathbf{Q}_1(k)\mathbf{w}(k)$. The structure of $\mathbf{F}_2(k)$ is a little complicated. In the following steps, we explain how to construct $\mathbf{F}_2(k)$:

1. Assume that the first nonzero element of $\mathbf{r}(k)$ is $r_{i_1}(k)$, thus all rows of $\mathbf{F}_2(k)$ before the i_1th row are zero vectors.

2. For i_1th row, the element corresponding to the $r_{i_1}(k)$ is one, and other entries of this row are zero.

3. If the next element of $\mathbf{r}(k)$ is nonzero, then the next row of $\mathbf{F}_2(k)$ contains one more nonzero entry equal to one corresponding to these nonzero coefficients of $\mathbf{r}(k)$. We repeat this step as far as a zero element appears in $\mathbf{r}(k)$.

4. As soon as a zero element appears in $\mathbf{r}(k)$, we look for the next nonzero element, and assume that it is $r_{i_2}(k)$. Then the next row of $\mathbf{F}_2(k)$ is similar to the previous row, but the element corresponding to $r_{i_2}(k)$ must be equal to one.

5. Suppose that the first nonzero element of $\mathbf{r}(k)$ after $r_{i_2}(k)$ is $r_{i_3}(k)$. Then next rows of $\mathbf{F}_2(k)$ until the (i_3-1)th row are identical to the last constructed row. Note that if it does not exist some nonzero element as $r_{i_3}(k)$, the remaining rows of $\mathbf{F}_2(k)$ are identical to the last constructed row.

6. The i_3th row of $\mathbf{F}_2(k)$ contains only one nonzero element equal to one, and it must be placed on column i_3. This row is similar to the i_1th row (step 2); however, the position of one is different. Now, we go back to the step 3 and repeat the same process to construct the next rows of $\mathbf{F}_2(k)$.

In Equation (7.18), the matrix $\mathbf{Q}_1(k)$ replaces the coefficients of $\mathbf{w}(k)$ which has absolute value lower than ϵ with zero. Then matrix $\mathbf{F}_1$ keeps the first coefficient unchanged. For the other components, this matrix subtracts the previous component from each of them. Hence, for the resulting vector, the matrix $\mathbf{Q}_2(k)$ changes the elements with an absolute value lower than ϵ to zero. Afterwards, the matrix $\mathbf{F}_2(k)$ reconstructs the weight vector and, in some sense, it inverts the effect of $\mathbf{F}_1$. Finally, for the resulting vector, the matrix $\mathbf{Q}_3(k)$ replaces the coefficients inside $[-\epsilon, \epsilon]$ with zero. The final result is identical to $\mathbb{F}_\epsilon(\mathbf{w}(k))$.

To clarify the process above, we describe the details for $\mathbf{w}(k) = [0\ 0.5\ 0.51\ 0.01\ 0.6\ 0.7\ 0.8\ 0.81\ 0\ -0.01]^T$, as an example, when $\epsilon = 0.02$. $\mathbf{Q}_1(k)$ is a diagonal matrix, where its diagonal is $[0\ 1\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 0]^T$. Therefore, $\mathbf{Q}_1(k)\mathbf{w}(k) = [0\ 0.5\ 0.51\ 0\ 0.6\ 0.7\ 0.8\ 0.81\ 0\ 0]^T$. Then $\mathbf{F}_1\mathbf{Q}_1(k)\mathbf{w}(k) = [0\ 0.5\ 0.01\ -0.51\ 0.6\ 0.1\ 0.1\ 0.01\ -0.81\ 0]^T$. The diagonal of $\mathbf{Q}_2(k)$ is $[0\ 1\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 0]^T$, and $\mathbf{Q}_2(k)\mathbf{F}_1\mathbf{Q}_1(k)\mathbf{w}(k) = [0\ 0.5\ 0\ -0.51\ 0.6\ 0.1\ 0.1\ 0\ -0.81\ 0]^T$. Following the procedure explained to construct $\mathbf{F}_2(k)$, we obtain the matrix $\mathbf{F}_2(k)$ as follows

$$\mathbf{F}_2(k) = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0\\ 0&1&0&1&0&0&0&0&0&0\\ 0&1&0&1&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&1&1&0&0&0&0\\ 0&0&0&0&1&1&1&0&0&0\\ 0&0&0&0&1&1&1&0&1&0\\ 0&0&0&0&1&1&1&0&1&0\\ 0&0&0&0&1&1&1&0&1&0 \end{bmatrix}. \tag{7.20}$$

Then $\mathbf{F}_2(k)\mathbf{Q}_2(k)\mathbf{F}_1\mathbf{Q}_1(k)\mathbf{w}(k) = [0\ 0.5\ -0.01\ -0.01\ 0.6\ 0.7\ 0.8\ -0.01\ -0.01\ -0.01]^T$. The diagonal of $\mathbf{Q}_3(k)$ is $[0\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0]^T$. Hence, $\mathbf{w}_s(k) = \mathbf{Q}_3(k)\mathbf{F}_2(k)\mathbf{Q}_2(k)\mathbf{F}_1\mathbf{Q}_1(k)\mathbf{w}(k) = [0\ 0.5\ 0\ 0\ 0.6\ 0.7\ 0.8\ 0\ 0\ 0]^T$. Also, if we use the feature function with $\epsilon = 0.02$, then we obtain $\mathbf{w}_s(k) = \mathbb{F}_\epsilon(\mathbf{w}(k)) =$

[0 0.5 0 0 0.6 0.7 0.8 0 0 0]^T.

7.6 Simulations

In this section, we apply the LMS, the F-LMS, the LCF-LMS, and the ALCF-LMS algorithms to system identification problems. In scenario 1, we utilize the LMS and the F-LMS algorithms. Then, in scenario 2, we use the LMS, the LCF-LMS, and the ALCF-LMS algorithms.

In both scenarios, the order of all the unknown systems is 39, i.e., they have 40 coefficients. The signal-to-noise ratio (SNR) is chosen as 20 dB. For all algorithms, the initial vector is $\mathbf{w}(0) = [0 \ \cdots \ 0]^T$, and the MSE learning curves are computed by averaging the outcomes of 200 independent trials.

7.6.1 Scenario 1

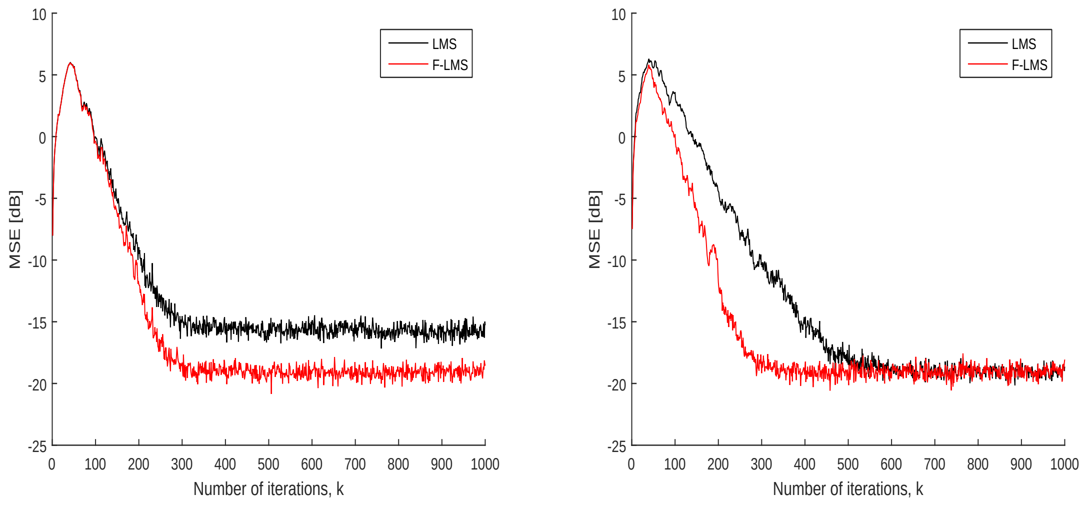
In this scenario, we apply the LMS and the F-LMS algorithms to identify some unknown lowpass and highpass systems. The first example considers predominantly lowpass and highpass systems defined as $\mathbf{w}_{o,l} = [0.4, \cdots, 0.4]^T$ and $\mathbf{w}_{o,h} = [0.4, -0.4, 0.4, \cdots, -0.4]^T$, respectively. The second example uses the interpolated models $\mathbf{w}'_{o,l} = [0.4, 0, 0.4, \cdots, 0, 0.4, 0]^T$ and $\mathbf{w}'_{o,h} = [0.4, 0, -0.4, 0, 0.4, \cdots, 0]^T$. The third example uses block-sparse lowpass and block-sparse highpass models, $\mathbf{w}''_{o,l}$ and $\mathbf{w}''_{o,h}$, whose entries are defined in (7.21) and (7.22), respectively.

$$w''_{o,l_i} = \begin{cases} 0 & \text{if } 0 \leq i \leq 9, \\ 0.05(i-9) & \text{if } 10 \leq i \leq 14, \\ 0.3 & \text{if } 15 \leq i \leq 24, \\ 0.3 - 0.05(i-24) & \text{if } 25 \leq i \leq 29, \\ 0 & \text{if } 30 \leq i \leq 39, \end{cases} \tag{7.21}$$

$$w''_{o,h_i} = (-1)^{i+1} w''_{o,l_i}. \tag{7.22}$$

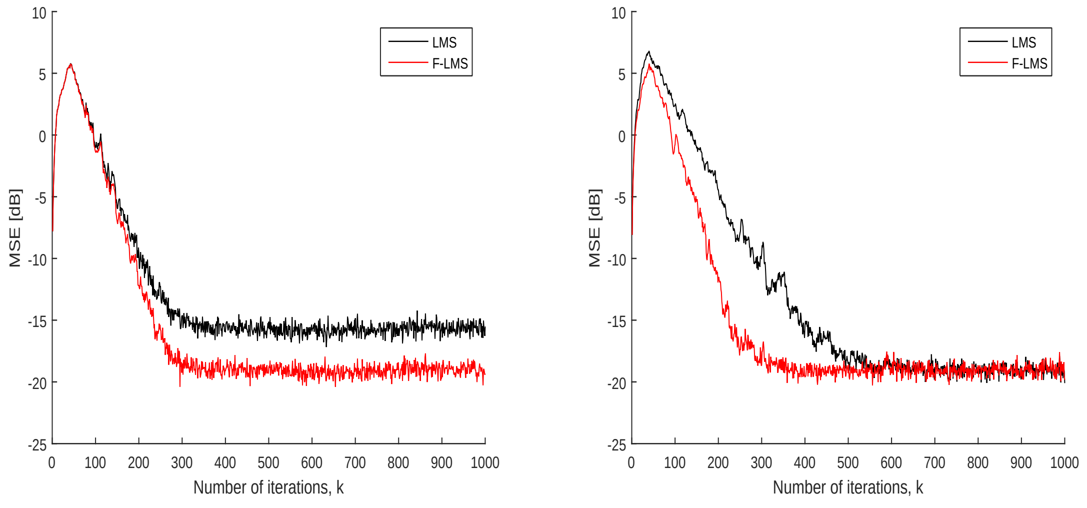
The input signal is a zero-mean white Gaussian noise with unit variance. The value of α for the F-LMS algorithm is chosen as 0.05. The values of the step size μ are informed later for each simulated scenario. The MSE learning curves of the LMS and the F-LMS algorithms are depicted in Figures 7.2 to 7.5.

Figure 7.2 depicts the MSE learning curves of the LMS and the F-LMS algorithms



(a) (b)

Figure 7.2: MSE learning curves of the LMS and F-LMS algorithms considering $\mathbf{w}_{o,l}$: (a) both algorithms with the same step size: $\mu = 0.03$; (b) LMS and F-LMS with step sizes equal to 0.01 and 0.03, respectively.



(a) (b)

Figure 7.3: MSE learning curves of the LMS and F-LMS algorithms considering $\mathbf{w}_{o,h}$: (a) both algorithms with the same step size: $\mu = 0.03$; (b) LMS and F-LMS with step sizes equal to 0.01 and 0.03, respectively.

considering the lowpass system $\mathbf{w}_{o,l}$. In Figure 7.2(a), both algorithms use the same step size $\mu = 0.03$ so that they exhibit similar convergence speeds. In this figure, we can observe that the F-LMS algorithm achieved a steady-state MSE which is more than 3 dB lower than the MSE results of the LMS algorithm. In Figure 7.2(b), the

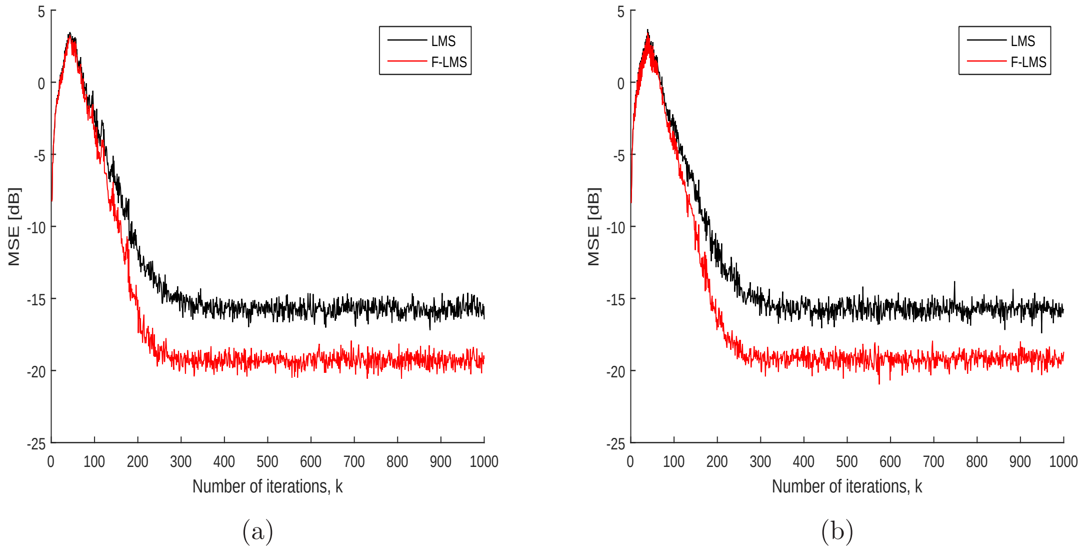


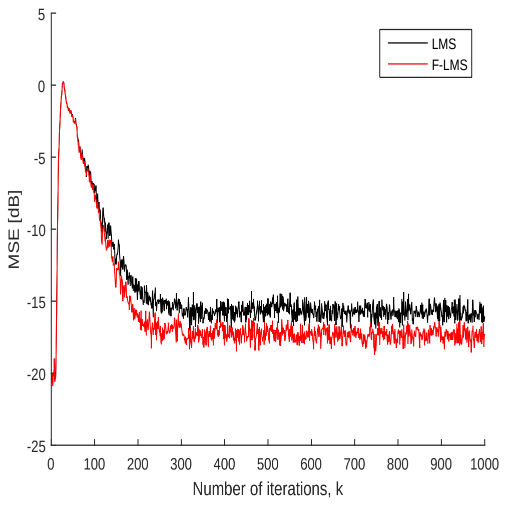
Figure 7.4: MSE learning curves of the LMS and F-LMS algorithms, both with step size $\mu = 0.03$, considering the unknown systems: (a) $\mathbf{w}'_{o,l}$ and (b) $\mathbf{w}'_{o,h}$.

steady-state MSE of the algorithms are fixed in order to compare their convergence speeds. Thus, we set the step sizes of the LMS and the F-LMS algorithms as 0.01 and 0.03, respectively. We can observe, in this figure, that the F-LMS algorithm converged much faster than the LMS algorithm.

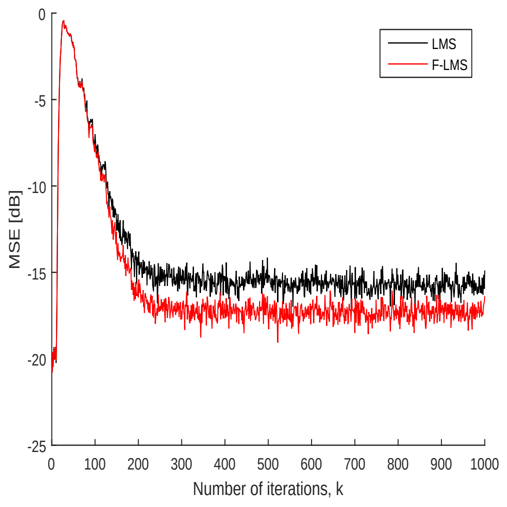
In Figure 7.3, we present results equivalent to the ones presented in Figure 7.2, but considering the highpass system $\mathbf{w}_{o,h}$. Once again, when the step sizes of both algorithms are the same ($\mu = 0.03$), refer to Figure 7.3(a), the F-LMS algorithm achieved lower steady-state MSE; whereas the F-LMS algorithm (with $\mu = 0.03$) converged much faster than the LMS algorithm (with $\mu = 0.01$) when their steady-state MSEs are fixed, as illustrated in Figure 7.3(b).

Figures 7.4(a) and 7.4(b) depict the MSE learning curves of the LMS and the F-LMS algorithms, both using $\mu = 0.03$, considering the interpolated systems $\mathbf{w}'_{o,l}$ and $\mathbf{w}'_{o,h}$, respectively. Notice, in both figures, that the F-LMS algorithm achieved lower steady-state MSE, thus outperforming the LMS algorithm.

Figures 7.5(a) and 7.5(b) depict the MSE learning curves of the LMS and the F-LMS algorithms, both using $\mu = 0.03$, considering the block-sparse systems $\mathbf{w}''_{o,l}$ and $\mathbf{w}''_{o,h}$, respectively. In both cases, the F-LMS algorithm achieved lower steady-state MSE, thus outperforming the LMS algorithm.



(a)



(b)

Figure 7.5: MSE learning curves of the LMS and F-LMS algorithms, both with step size $\mu = 0.03$, considering the unknown systems: (a) $\mathbf{w}''_{o,l}$ and (b) $\mathbf{w}''_{o,h}$.

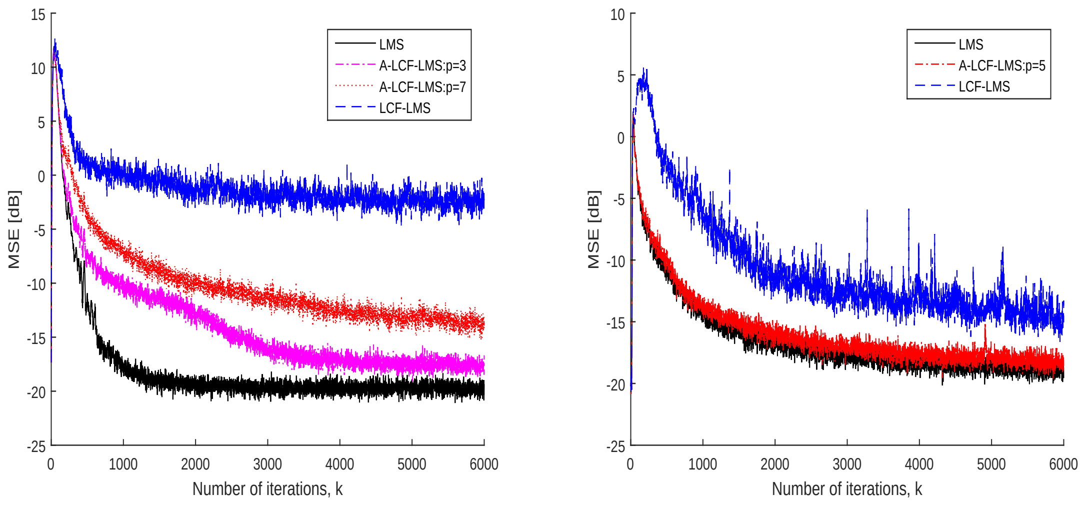
7.6.2 Scenario 2

In this scenario, we apply the LMS, the LCF-LMS, the ALCF-LMS, the I-LCF-LMS, and the AI-LCF-LMS algorithms to identify two unknown systems. The first unknown system is the predominantly lowpass system $\mathbf{w}_{o,l}$. The second unknown model is a block-sparse model, $\mathbf{w}'''_{o,l}$, defined as follows

$$w'''_{o,l_i} = \begin{cases} 0 & \text{if } 0 \leq i \leq 9, \\ 0.04 + 0.01(i-9) & \text{if } 10 \leq i \leq 17, \\ 0.5 & \text{if } 18 \leq i \leq 21, \\ 0.13 - 0.01(i-21) & \text{if } 22 \leq i \leq 29, \\ 0 & \text{if } 30 \leq i \leq 39. \end{cases} \tag{7.23}$$

In the case of the LCF-LMS and the ALCF-LMS algorithms, the input signal is an autoregressive signal generated by $x(k) = 0.99x(k-1) + n(k-1)$. However, we do not have any restrictions on the input signal when utilizing the I-LCF-LMS and the AI-LCF-LMS algorithms. Thus, we use a zero-mean white Gaussian noise with unit variance as the input signal when implementing the I-LCF-LMS and the AI-LCF-LMS algorithms. The step size μ for the all algorithms is 0.003. Also, we adopt ϵ equal to 0.02.

Figures 7.6(a) and 7.6(b) show the MSE learning curves of the LMS, the LCF-LMS,



(a) (b)

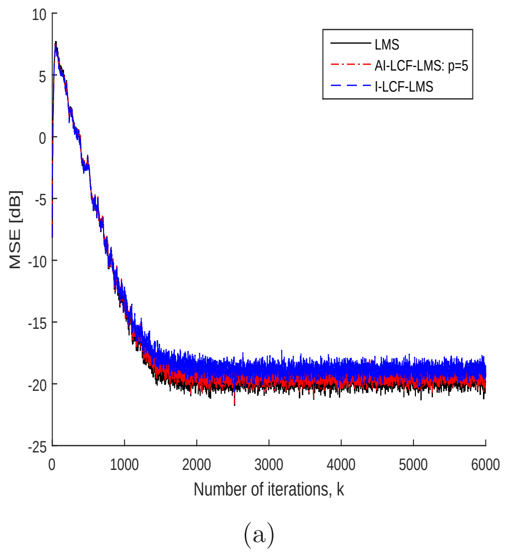
Figure 7.6: MSE learning curves of the LMS, the LCF-LMS, and the ALCF-LMS algorithms considering the unknown systems: (a) $\mathbf{w}_{o,l}$ and (b) $\mathbf{w}'''_{o,l}$.

and the ALCF-LMS algorithms. Furthermore, the MSE learning curves of the LMS, the I-LCF-LMS, and the AI-LCF-LMS algorithms are illustrated in Figures 7.7(a) and 7.7(b).

Figure 7.6(a) shows the learning curves of the mentioned algorithms when they are applied to identify the predominantly lowpass unknown system $\mathbf{w}_{o,l}$. We can observe that the LCF-LMS algorithm, the blue curve, has high MSE but it has the lowest computational complexity. In the steady-state environment, it implements only one multiplication to calculate the error signal. However, the LMS algorithms, the black curve, requires forty multiplication to compute the error signal, and it has the highest computational burden. The ALCF-LMS algorithms have acceptable performances and, using $p = 3$ and 7, they need thirteen and six multiplication to calculate the error signal, respectively.

Figure 7.6(b) depicts the learning curves of the algorithms, when they are applied to identify the block-sparse lowpass unknown model $\mathbf{w}'''_{o,l}$. As can be seen, the LCF-LMS algorithm, the blue curve, has the highest MSE but it executes only three multiplication to compute the error signal. The red curve illustrates the remarkable performance of the ALCF-LMS algorithm. Indeed, its learning curve is extremely close to the learning curve of the LMS algorithm. However, in the steady-state environment, it implements only six multiplication to calculate the error signal.

Figure 7.7(a) illustrates the learning curves of the LMS, the I-LCF-LMS, and the



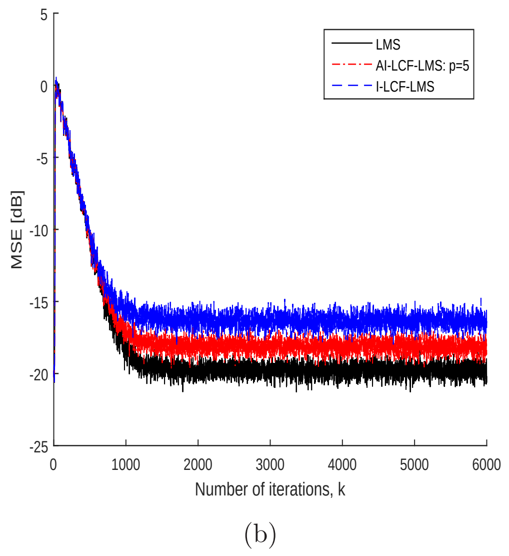


Figure 7.7: MSE learning curves of the LMS, the I-LCF-LMS, and the AI-LCF-LMS algorithms considering the unknown systems: (a) $\mathbf{w}_{o,l}$ and (b) $\mathbf{w}'''_{o,l}$.

AI-LCF-LMS algorithms when they are utilized in the identification of the predominantly lowpass unknown system $\mathbf{w}_{o,l}$. The three algorithms have the same convergence rate; however, the LMS algorithm has the best MSE, followed by the AI-LCF-LMS and the I-LCF-LMS algorithms. As can be seen, the superiority of the MSE of the LMS algorithm to the MSE of the other two algorithms is not remarkable but the LMS algorithm has higher computational load. In the steady-state environment, for the calculation of the error signal, the LMS algorithm implements 40 multiplication, whereas the I-LCF-LMS and the AI-LCF-LMS algorithms execute one and eight multiplication, respectively.

The MSE learning curves of the LMS, the I-LCF-LMS, and the AI-LCF-LMS algorithms, when they are applied to identify the block-sparse unknown system $\mathbf{w}'''_{o,l}$, are presented in Figure 7.7(b). The curves shown in this figure indicate that the LMS algorithm has the best misadjustment, followed by the AI-LCF-LMS and the I-LCF-LMS algorithms. Moreover, we can observe that the three algorithms have similar convergence speed. We must note that the computational complexity of the LMS algorithm is higher than that of the I-LCF-LMS and of the AI-LCF-LMS algorithms. In other words, to compute the error signal in the steady-state environment, the LMS algorithm requires 40 multiplication; however, the I-LCF-LMS and the AI-LCF-LMS algorithms need three and six multiplication, respectively.

As can be seen, in Scenario 1, the learning curves of the F-LMS algorithm are lower than that of the LMS algorithm. However, in Scenario 2, the learning curves of the LCF-LMS, the ALCF-LMS, the I-LCF-LMS, and the AI-LCF-LMS algorithms are higher than that of the LMS algorithm. It is worthwhile to mention that the computational complexity of the F-LMS algorithm is higher than that of the LMS algorithm, whereas the LCF-LMS, the ALCF-LMS, the I-LCF-LMS, and the AI-LCF-LMS algorithms require lower computational resources as compared to the LMS algorithm. Therefore, higher MSE in the performance of the low-complexity F-LMS algorithms is compensated by their lower computational complexity.

7.7 Conclusions

In this chapter, we have proposed a family of algorithms called Feature LMS (F-LMS). The F-LMS algorithms are capable of exploiting specific features of the unknown system to be identified in order to accelerate convergence speed and/or reduce steady-state MSE, obtaining a more accurate estimate. The main idea is to apply a sparsity-promoting function to a linear combination of the parameters, in which this linear combination should reveal the sparsity hidden in the parameters, i.e., the linear combination exploits the specific structure/feature in order to generate a sparse vector. Some examples of the F-LMS algorithms having low computational complexity and exploiting the lowpass and highpass characteristics of unknown systems were introduced. Simulation results confirmed the superior performance of the F-LMS algorithm in comparison with the LMS algorithm.

Furthermore, we have introduced the low-complexity F-LMS (LCF-LMS) and the alternative LCF-LMS (ALCF-LMS) algorithms in order to exploit hidden sparsity in the parameter with low computational cost. For this purpose, we have defined the feature function. The proposed algorithms have lower computational burden compared to the LMS algorithm; however, they have competitive performance. Also, we have introduced the improved versions of the LCF-LMS and the ALCF-LMS algorithms. Numerical results showed the competitive performance of the AI-LCF-LMS algorithm while requiring less multiplication to compute the error signal.

In future works, we intend to investigate other choices for the sparsity-promoting penalty function and the feature matrix. Also, we want to analyze the stability and MSE of the F-LMS and the LCF-LMS algorithms.

Chapter 8

Conclusions, and Future Works

In this thesis, we have investigated a number of data-selective adaptive filtering algorithms. It is generally accepted that data selection is an effective strategy to reduce the computational resources of the adaptive algorithms. To benefit from data selection in adaptive filtering algorithms, we have utilized the set-membership filtering (SMF) approach.

In set-membership (SM) adaptive filtering algorithms, the inclusion of *a priori* information, such as the noise bound, into the objective function leads to some noticeable advantages. The SM adaptive algorithms evaluate, choose, and process data at each iteration of their learning process. These algorithms have the potential to outperform the conventional adaptive filtering algorithms. Indeed, they retain the advantages of their traditional counterparts; however, they are more accurate, more robust against noise, and have lower computational load.

Moreover, we incorporate some sparsity-aware techniques into the SM adaptive algorithms. Thus, we introduced some sparsity-aware set-membership adaptive filtering algorithms. In order to exploit the sparsity in system models, we utilized the l_0 norm approximation, the discard function, and the feature matrices. The l_0 norm approximation and the discard function exploit the sparsity in coefficients close to zero; however, the feature matrices exploit the sparsity in linear combination of the parameters.

8.1 Contributions

The thesis started by reviewing the classical adaptive filtering algorithms. Also, we have introduced the SM normalized least-mean-square (SM-NLMS) and the SM affine projection (SM-AP) algorithms briefly. Then we have analyzed the robustness (in the sense of l_2 stability) of the SM-NLMS and the SM-AP algorithms. One of the major drawbacks of adopting the conventional algorithms is that one cannot guarantee the convergence of the algorithm independent of the choice of the parameters. However, when the additional noise is bounded, we have proved that the SM algorithms never diverge.

Moreover, the SMF approach has been generalized to trinion and quaternion numbers. Whenever the problem at hand suits both the quaternion and trinion solutions, the trinion algorithms clearly have an advantage over the quaternion ones in terms of computational burden. Furthermore, we have derived a new set-membership partial-update affine projection algorithm. This algorithm can improve the convergence rate significantly, particularly in a nonstationary environment.

In addition, some data-selective adaptive filtering algorithms have been proposed in order to exploit sparsity in systems with low computational cost. The key idea is to apply the discard function and the l_0 norm approximation. In particular, the use of discard function can effectively decrease the computational complexity. Finally, we have derived some feature least-mean-square (F-LMS) algorithms to exploit hidden sparsity in models when adjacent coefficients have a strong relation. To this end, the feature matrices and the feature function play fundamental roles.

8.2 Future Works

In this section, we list our future works. Indeed, research into studying and analyzing the F-LMS and the low-complexity (LCF-LMS) algorithms is already in progress. We are investigating some mathematical properties, such as the stability and MSE, of the F-LMS and the LCF-LMS algorithms. Also, we are currently in the process of investigating other choices for the sparsity-promoting penalty function and the feature matrix.

A possible topic for research is to employ distinct feature matrices in an online basis aiming at verifying the best one for a given iteration. It is also possible to derive a multitude for feature matrices inspired by previous knowledge of the spectral content of the unknown system model.

Another future work will concentrate on proposing some set-membership quaternion-valued adaptive filtering algorithms to exploit sparsity in system models. Also, further works need to be performed in order to analyze the performance of the proposed trinion- and quaternion-valued and partial-update adaptive algorithms.

Bibliography

[1] LIMA, M. V. S. *Energy-efficient adaptive filters.* D.Sc. Thesis, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, December 2013.

[2] DINIZ, P. S. R. *Adaptive Filtering: Algorithms and Practical Implementation.* 4th ed. New York, USA, Springer, 2013.

[3] JIANG, M. *Quaternion-valued adaptive signal processing and its application to adaptive beamforming and wind profile prediction.* D.Sc. Thesis, University of Sheffield, 2016.

[4] BERBERIDIS, D., KEKATOS, V., GIANNAKIS, G. B. "Online censoring for large-scale regressions with application to streaming big data", *IEEE Transactions on Signal Processing*, v. 64, n. 15, pp. 3854–3867, August 2016.

[5] WANG, G., BERBERIDIS, D., KEKATOS, V., et al. "Online reconstruction from big data via compressive censoring". In: *IEEE Global Conference on Signal and Information Processing (GlobalSIP 2014)*, pp. 326–330, Atlanta, GA, USA, December 2014.

[6] YAZDANPANAH, H., LIMA, M. V. S., DINIZ, P. S. R. "On the robustness of the set-membership NLMS algorithm". In: *9th IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM 2016)*, pp. 1–5, Rio de Janeiro, Brazil, July 2016.

[7] YAZDANPANAH, H., LIMA, M. V. S., DINIZ, P. S. R. "On the robustness of set-membership adaptive filtering algorithms", *EURASIP Journal on Advances in Signal Processing*, v. 72, December 2017.

[8] LIMA, M. V. S., FERREIRA, T. N., MARTINS, W. A., et al. "Sparsity-aware data-selective adaptive filters", *IEEE Transactions on Signal Processing*, v. 62, n. 17, pp. 4557–4572, September 2014.

[9] GOLLAMUDI, S., NAGARAJ, S., KAPOOR, S., et al. "Set-membership filtering and a set-membership normalized LMS algorithm with an adaptive step size", *IEEE Signal Processing Letters*, v. 5, n. 5, pp. 111–114, May 1998.

[10] SAYED, A. H. *Adaptive Filters.* New York, USA, Wiley-IEEE, 2008.

[11] HAYKIN, S. *Adaptive Filter Theory.* 4th ed. Englewood Cliffs, NJ, Prentice Hall, 2002.

[12] CLASER, R., NASCIMENTO, V. H., ZAKHAROV, Y. V. "A low-complexity RLS-DCD algorithm for volterra system identification". In: *24th European Signal Processing Conference (EUSIPCO 2016)*, pp. 6–10, Budapest, Hungary, September 2016.

[13] DINIZ, P. S. R., LIMA, M. V. S., MARTINS, W. A. "Semi-blind data-selective algorithms for channel equalization". In: *IEEE International Symposium on Circuits and Systems (ISCAS 2008)*, pp. 53–56, Seattle, WA, USA, May 2008.

[14] ANDERSEN, K. T., MOONEN, M. "Adaptive time-frequency analysis for noise reduction in an audio filter bank with low delay", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, v. 24, n. 4, pp. 784–795, April 2016.

[15] AZPICUETA-RUIZ, L. A., ARENAS-GARCÍA, J., NASCIMENTO, V. H., et al. "Reduced-cost combination of adaptive filters for acoustic echo cancellation". In: *International Telecommunications Symposium (ITS 2014)*, pp. 1–5, São Paulo, Brazil, August 2014.

[16] DE LAMARE, R. C. "Adaptive and iterative multi-branch MMSE decision feedback detection algorithms for multi-antenna systems", *IEEE Transactions on Wireless Communications*, v. 12, n. 10, pp. 5294–5308, October 2013.

[17] YAZDANPANAH, H., DINIZ, P. S. R. "New trinion and quaternion set-membership affine projection algorithms", *IEEE Transactions on Circuits and Systems II: Express Briefs*, v. 64, n. 2, pp. 216–220, February 2017.

[18] EHRENFRIED, K., KOOP, L. "Comparison of iterative deconvolution algorithms for the mapping of acoustic sources", *AIAA Journal*, v. 45(7), pp. 1584–1595, July 2007.

[19] ZHENG, X., CHEN, B. M. “Identification of market forces in the financial system adaptation framework”. In: *IEEE International Conference on Control and Automation (ICCA 2010)*, pp. 103–108, Xiamen, China, June 2010.

[20] LEHMANN, E. L., CASELLA, G. *Theory of Point Estimation.* 2nd ed. New York, USA, Springer, 2003.

[21] THEODORIDIS, S., KOUTROUMBAS, K. *Pattern Recognition.* 4th ed. , Academic Press, 2008.

[22] BISHOP, C. M. *Pattern Recognition and Machine Learning.* 2nd ed. , Springer, 2011.

[23] COMBETTES, P. L. “The foundations of set theoretic estimation”, *Proceedings of the IEEE*, v. 81, n. 2, pp. 182–208, February 1993.

[24] LIMA, M. V. S., DINIZ, P. S. R. “Fast learning set theoretic estimation”. In: *21st European Signal Processing Conference (EUSIPCO 2013)*, pp. 1–5, Marrakech, Moroccol, September 2013.

[25] COMBETTES, P. L., TRUSSELL, H. J. “The use of noise properties in set theoretic estimation”, *IEEE Transactions on Signal Processing*, v. 39, n. 7, pp. 1630–1641, July 1991.

[26] FOGEL, E., HUANG, Y.-F. “On the value of information in system identification–bounded noise case”, *Automatica*, v. 18, n. 2, pp. 229–238, March 1982.

[27] DELLER, J. R. “Set-membership identification in digital signal processing”, *IEEE ASSP Magazine*, v. 6, n. 4, pp. 4–20, October 1989.

[28] GOLLAMUDI, S., KAPOOR, S., NAGARAJ, S., et al. “Set-membership adaptive equalization and updator-shared implementation for multiple channel communications systems”, *IEEE Transactions on Signal Processing*, v. 46, n. 9, pp. 2372–2385, September 1998.

[29] NAGARAJ, S., GOLLAMUDI, S., KAPOOR, S., et al. “BEACON: an adaptive set-membership filtering technique with sparse updates”, *IEEE Transactions on Signal Processing*, v. 47, n. 11, pp. 2928–2941, November 1999.

[30] DINIZ, P. S. R., WERNER, S. "Set-membership binormalized data reusing LMS algorithms", *IEEE Transactions on Signal Processing*, v. 51, n. 1, pp. 124–134, January 2003.

[31] WERNER, S., DINIZ, P. S. R. "Set-membership affine projection algorithm", *IEEE Signal Processing Letters*, v. 8, n. 8, pp. 231–235, August 2001.

[32] GOODWIN, G. C., PAYNE, R. L. *Dynamic System Identification: Experiment Design and Data Analysis*. New York, USA, Academic, 1977.

[33] GALDINO, J. F., APOLINÁRIO, JR., J. A., DE CAMPOS, M. L. R. "A set-membership NLMS algorithm with time-varying error bound". In: *IEEE International Symposium on Circuits and Systems (ISCAS 2006)*, pp. 277–280, Island of Kos, Greece, May 2006.

[34] OZEKI, K., UMEDA, T. "An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties", *Electronics and Communications in Japan*, v. 67-A, n. 5, pp. 19–27, 1984.

[35] DINIZ, P. S. R., YAZDANPANAH, H. "Data censoring with set-membership algorithms". In: *IEEE Global Conference on Signal and Information Processing (GlobalSIP 2017)*, Montreal, Canada, November 2017.

[36] LIMA, M. V. S., DINIZ, P. S. R. "Steady-state analysis of the set-membership affine projection algorithm". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2010)*, pp. 3802–3805, Dallas, USA, March 2010.

[37] LIMA, M. V. S., DINIZ, P. S. R. "Steady-state MSE performance of the set-membership affine projection algorithm", *Circuits, Systems, and Signal Processing*, v. 32, n. 4, pp. 1811–1837, January 2013.

[38] ARABLOUEI, R., DOĞANÇAY, K. "Tracking performance analysis of the set-membership NLMS adaptive filtering algorithm". In: *Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC 2012)*, pp. 1–6, Hollywood, CA, USA, December 2012.

[39] CARINI, A., SICURANZA, G. L. "Analysis of a multichannel filtered-x set-membership affine projection algorithm". In: *IEEE International Conference on Acoustics, Speech and Signal Processing Proceedings*, Toulouse, France, May 2006.

[40] GUO, L., HUANG, Y.-F. "Frequency-domain set-membership filtering and its applications", *IEEE Transactions on Signal Processing*, v. 55, n. 4, pp. 1326–1338, April 2007.

[41] WERNER, S., APOLINÁRIO, JR., J. A., DINIZ, P. S. R. "Set-membership proportionate affine projection algorithms", *EURASIP Journal on Audio, Speech, and Music Processing*, v. 2007, n. 1, pp. 1–10, January 2007.

[42] MARTINS, W. A., LIMA, M. V. S., DINIZ, P. S. R. "Semi-blind data-selective equalizers for QAM". In: *IEEE 9th Workshop on Signal Processing Advances in Wireless Communications (SPAWC 2008)*, pp. 501–505, Recife, Brazil, July 2008.

[43] BHOTTO, M. Z. A., ANTONIOU, A. "Robust set-membership affine-projection adaptive-filtering algorithm", *IEEE Transactions on Signal Processing*, v. 60, n. 1, pp. 73–81, January 2012.

[44] ZHANG, S., ZHANG, J. "Set-membership NLMS algorithm with robust error bound", *IEEE Transactions on Circuits and Systems II: Express Briefs*, v. 61, n. 7, pp. 536–540, July 2014.

[45] MAO, W. L. "Robust set-membership filtering techniques on GPS sensor jamming mitigation", *IEEE Sensors Journal*, v. 17, n. 6, pp. 1810–1818, March 2017.

[46] LIMA, M. V. S., DINIZ, P. S. R. "On the steady-state MSE performance of the set-membership NLMS algorithm". In: *7th International Symposium on Wireless Communication Systems (ISWCS 2010)*, pp. 389–393, York, UK, September 2010.

[47] TAKAHASHI, N., YAMADA, I. "Steady-state mean-square performance analysis of a relaxed set-membership NLMS algorithm by the energy conservation argument", *IEEE Transactions on Signal Processing*, v. 57, n. 9, pp. 3361–3372, September 2009.

[48] DINIZ, P. S. R. "Convergence performance of the simplified set-membership affine projection algorithm", *Circuits, Systems, and Signal Processing*, v. 30, n. 2, pp. 439–462, April 2011.

[49] RUPP, M. "Pseudo affine projection algorithms revisited: robustness and stability analysis", *IEEE Transactions on Signal Processing*, v. 59, n. 5, pp. 2017–2023, May 2011.

[50] PROAKIS, J. G. *Digital Communications*. USA, McGraw-Hill, 1995.

[51] MARTINS, W. A., LIMA, M. V. S., DINIZ, P. S. R., et al. "Optimal constraint vectors for set-membership affine projection algorithms", *Signal Processing*, v. 134, pp. 285–294, May 2017.

[52] HAMILTON, W. R. "On quaternions, or a new system of imaginaries in algebra", *Philosophical Magazine*, v. 25, n. 3, pp. 489–495, 1844.

[53] PEI, S. C., CHANG, J. H., DING, J. J. "Commutative reduced biquaternions and their Fourier transform for signal and image processing applications", *IEEE Transactions on Signal Processing*, v. 52, n. 7, pp. 2012–2031, July 2004.

[54] GUO, L. Q., ZHU, M., GE, X. H. "Reduced biquaternion canonical transform, convolution and correlation", *Signal Processing*, v. 91, n. 8, pp. 2147–2153, August 2011.

[55] TOOK, C. C., STRBAC, G., AIHARA, K., et al. "Quaternion-valued short term joint forecasting of three-dimensional wind and atmospheric parameters", *Renewable Energy*, v. 36, n. 6, pp. 1754–1760, June 2011.

[56] BARTHÉLEMY, Q., LARUE, A., MARS, J. I. "About QLMS derivations", *IEEE Signal Processing Letters*, v. 21, n. 2, pp. 240–243, February 2014.

[57] JIANG, M. D., LIU, W., LI, Y. "A general quaternion-valued gradient operator and its applications to computational fluid dynamics and adaptive beamforming". In: *International Conference on Digital Signal Processing (DSP 2014)*, pp. 821–826, Hong Kong, China, August 2014.

[58] ZHANG, X. R., LIU, W., XU, Y. G., et al. "Quaternion-valued robust adaptive beamformer for electromagnetic vector-sensor arrays with worst-case constraint", *Signal Processing*, v. 104, pp. 274–283, November 2014.

[59] UJANG, B. C., TOOK, C. C., MANDIC, D. P. "Quaternion-valued nonlinear adaptive filtering", *IEEE Transactions on Neural Networks*, v. 22, n. 8, pp. 1193–1206, August 2011.

[60] TOOK, C. C., MANDIC, D. P. "The quaternion LMS algorithm for adaptive filtering of hypercomplex processes", *IEEE Transactions on Signal Processing*, v. 57, n. 4, pp. 1316–1327, April 2009.

[61] TOOK, C. C., MANDIC, D., BENESTY, J. “Study of the quaternion LMS and four-channel LMS algorithms”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2009)*, pp. 3109–3112, Taipei, Taiwan, April 2009.

[62] NETO, F. G. A., NASCIMENTO, V. H. “A novel reduced-complexity widely linear QLMS algorithm”. In: *IEEE Statistical Signal Processing Workshop (SSP 2011)*, pp. 81–84, Nice, France, June 2011.

[63] PEI, S.-C., CHENG, C.-M. “Color image processing by using binary quaternion-moment-preserving thresholding technique”, *IEEE Transactions on Image Processing*, v. 8, n. 5, pp. 614–628, May 1999.

[64] BIHAN, N. L., SANGWINE, S. J. “Quaternion principal component analysis of color images”. In: *International Conference on Image Processing (ICIP 2003)*, pp. I–809–12 vol. 1, Barcelona, Spain, Spain, September 2003.

[65] CAMPA, R., CAMARILLO, K., ARIAS, L. “Kinematic modeling and control of robot manipulators via unit quaternions: application to a spherical wrist”. In: *IEEE Conference on Decision and Control*, pp. 6474–6479, San Diego, CA, USA, December 2006.

[66] GUO, X., LIU, Z., LIU, W., et al. “Three-dimensional wind profile prediction with trinion-valued adaptive algorithms”. In: *International Conference on Digital Signal Processing (DSP 2015)*, pp. 566–569, Singapore, July 2015.

[67] JAHANCHAHI, C., TOOK, C. C., MANDIC, D. P. “A class of quaternion valued affine projection algorithms”, *Signal Processing*, v. 93, pp. 1712–1723, 2013.

[68] ELL, T. A., SANGWINE, S. J. “Quaternion involutions and anti-involutions”, *Computers & Mathematics with Applications*, v. 53, n. 1, pp. 137–143, 2007.

[69] MANDIC, D. P., JAHANCHAHI, C., TOOK, C. C. “A quaternion gradient operator and its applications”, *Signal Processing Letters*, v. 18, n. 1, pp. 47–50, 2011.

[70] BRANDWOOD, D. H. “A complex gradient operator and its application in adaptive array theory”. In: *IEE Proceedings F - Communications, Radar and Signal Processing*, pp. 11–16, February 1983.

[71] ASSEFA, D., MANSINHA, L., TIAMPO, K. F., et al. "The trinion Fourier transform of color images", *Signal Processing*, v. 91, n. 8, pp. 1887–1900, August 2011.

[72] VAN DEN BOS, A. "Complex gradient and Hessian", *IEE Proceedings Vision Image Signal Process*, v. 141, n. 6, pp. 380–383, December 1994.

[73] GOU, X., XU, Y., LIU, Z., et al. "Quaternion-capon beamformer using crossed-dipole arrays". In: *IEEE 4th International Symposium on Microwave, Antenna, Propagation, and EMC Technologies for Wireless Communications (MAPE 2011)*, pp. 34–37, Beijing, China, November 2011.

[74] TAO, J. W., CHANG, W. X. "A novel combined beamformer based on hypercomplex processes", *IEEE Transactions on Aerospace and Electronic Systems*, v. 49, n. 2, pp. 1276–1289, April 2013.

[75] TAO, J. W., CHANG, W. X. "Adaptive beamforming based on complex quaternion processes", *Mathematical Problems in Engineering*, v. 2014, 2014.

[76] COMPTON, R. "On the performance of a polarization sensitive adaptive array", *IEEE Transactions on Antennas and Propagation*, v. 29, n. 5, pp. 718–725, September 1981.

[77] LI, J., COMPTON, R. T. "Angle and polarization estimation using ESPRIT with a polarization sensitive array", *IEEE Transactions on Antennas and Propagation*, v. 39, n. 9, pp. 1376–1383, September 1991.

[78] GOOGLE. "RE<C: surface level wind data collection", *Google Code 2011*, [Online]. Available: http://code.google.com/p/google-rec-csp/ .

[79] DOĞANÇAY, K. *Partial-Update Adaptive Signal Processing: Design, Analysis and Implementation.* Academic Press, Elsevier, 2008.

[80] DOUGLAS, S. C. "Adaptive filters employing partial updates", *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, v. 44, n. 3, pp. 209–216, March 1997.

[81] ABOULNASR, T., MAYYAS, K. "Complexity reduction of the NLMS algorithm via selective coefficient updating", *IEEE Transactions on Signal Processing*, v. 47, n. 5, pp. 1421–1427, May 1999.

[82] DOĞANÇAY, K., TANRIKULU, O. "Adaptive filtering algorithms with selective partial updates", *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, v. 48, n. 8, pp. 762–769, August 2001.

[83] WERNER, S., DE CAMPOS, M. L. R., DINIZ, P. S. R. "Mean-squared analysis of the partial-update NLMS algorithm", *Brazilian Telecommunications Journal - SBrT*, v. 18, pp. 77–85, June 2003.

[84] WERNER, S., DE CAMPOS, M. L. R., DINIZ, P. S. R. "Partial-update NLMS algorithms with data-selective updating", *IEEE Transactions on Signal Processing*, v. 52, n. 4, pp. 938–949, April 2004.

[85] GODAVARTI, M., III, A. O. H. "Partial update LMS algorithms", *IEEE Transactions on Signal Processing*, v. 53, n. 7, pp. 2384–2399, July 2005.

[86] GRIRA, M., CHAMBERS, J. A. "Adaptive partial update channel shortening in impulsive noise environments". In: *15th International Conference on Digital Signal Processing*, pp. 555–558, Cardiff, July 2007.

[87] ARABLOUEI, R., DOĞANÇAY, K., PERREAU, S. "Partial-update adaptive decision-feedback equalization". In: *19th European Signal Processing Conference*, pp. 2205–2209, Barcelona, Spain, August 2011.

[88] DINIZ, P. S. R., PINTO, G. O., HJORUNGNES, A. "Data selective partial-update affine projection algorithm". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2008)*, pp. 3833–3836, Las Vegas, USA, April 2008.

[89] TANDON, A., SWAMY, M. N. S., AHMAD, M. O. "Partial-update L^∞-norm based algorithms", *IEEE Transactions on Circuits and Systems I: Regular Papers*, v. 54, n. 2, pp. 411–419, February 2007.

[90] BHOTTO, M. Z. A., ANTONIOU, A. "A new partial-update NLMS adaptive-filtering algorithm". In: *IEEE 27th Canadian Conference on Electrical and Computer Engineering (CCECE 2014)*, pp. 1–5, Toronto, Canada, May 2014.

[91] DENG, G. "Partial update and sparse adaptive filters", *IET Signal Processing*, v. 1, n. 1, pp. 9–17, March 2007.

[92] ARABLOUEI, R., DOĞANÇAY, K. "Set-membership recursive least-squares adaptive filtering algorithm". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2012)*, pp. 3765–3768, Kyoto, Japan, March 2012.

[93] BHOTTO, M. Z. A., ANTONIOU, A. "A robust constrained set-membership affine-projection adaptive-filtering algorithm". In: *5th International Symposium on Communications, Control and Signal Processing (ISCCSP 2012)*, pp. 1–4, Rome, Italy, May 2012.

[94] ABADI, M. S. E., HUSOY, J. H. "Set-membership subband adaptive filters". In: *3rd International Symposium on Communications, Control and Signal Processing (ISCCSP 2008)*, pp. 193–196, St. Julian's, Malta, March 2008.

[95] DINIZ, P. S. R., YAZDANPANAH, H. "Improved set-membership partial-update affine projection algorithm". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2016)*, pp. 4174–4178, Shanghai, China, March 2016.

[96] DUTTWEILER, D. L. "Proportionate normalized least-mean-squares adaptation in echo cancelers", *IEEE Transactions on Speech and Audio Processing*, v. 8, n. 5, pp. 508–518, September 2000.

[97] BENESTY, J., GAY, S. L. "An improved PNLMS algorithm". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2002)*, pp. 1881–1884, Dallas, USA, May 2002.

[98] GAY, S. L. "An efficient, fast converging adaptive filter for network echo cancellation". In: *Thirty-Second Asilomar Conference on Signals, Systems amp, Computers (ACSSC 1998)*, pp. 394–398, Pacific Grove, CA, USA, November 1998.

[99] PALEOLOGU, C., CIOCHINA, S., BENESTY, J. "An efficient proportionate affine projection algorithm for echo cancellation", *IEEE Signal Processing Letters*, v. 17, n. 2, pp. 165–168, February 2010.

[100] MENG, R., DE LAMARE, R. C., NASCIMENTO, V. H. "Sparsity-aware affine projection adaptive algorithms for system identification". In: *Sensor Signal Processing for Defence (SSPD 2011)*, pp. 1–5, London, U.K., September 2011.

[101] KOPSINIS, Y., SLAVAKIS, K., THEODORIDIS, S. "Online sparse system identification and signal reconstruction using projections onto weighted l_1 balls", *IEEE Transactions on Signal Processing*, v. 59, n. 3, pp. 936–952, March 2011.

[102] CHEN, Y., GU, Y., HERO, A. O. "Sparse LMS for system identification". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2009)*, pp. 3125–3128, Taipei, Taiwan, April 2009.

[103] BABADI, B., KALOUPTSIDIS, N., TAROKH, V. "SPARLS: the sparse RLS algorithm", *IEEE Transactions on Signal Processing*, v. 58, n. 8, pp. 4013–4025, August 2010.

[104] LIMA, M. V. S., SOBRON, I., MARTINS, W. A., et al. "Stability and MSE analyses of affine projection algorithms for sparse system identification". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2014)*, pp. 6399–6403, Florence, Italy, May 2014.

[105] LIMA, M. V. S., MARTINS, W. A., DINIZ, P. S. R. "Affine projection algorithms for sparse system identification". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2013)*, pp. 5666–5670, Vancouver, Canada, May 2013.

[106] GU, Y., JIN, J., MEI, S. "l_0 norm constraint LMS algorithm for sparse system identification", *IEEE Signal Processing Letters*, v. 16, n. 9, pp. 774–777, September 2009.

[107] PELEKANAKIS, K., CHITRE, M. "New sparse adaptive algorithms based on the natural gradient and the l_0-norm", *IEEE Journal of Oceanic Engineeering*, v. 38, n. 2, pp. 323–332, April 2013.

[108] FERREIRA, T. N., LIMA, M. V. S., DINIZ, P. S. R., et al. "Low-complexity proportionate algorithms with sparsity-promoting penalties". In: *IEEE International Symposium on Circuits and Systems (ISCAS 2016)*, Canada, May 2016.

[109] YAZDANPANAH, H., DINIZ, P. S. R., LIMA, M. V. S. "A simple set-membership affine projection algorithm for sparse system modeling". In: *24th European Signal Processing Conference (EUSIPCO 2016)*, pp. 1798–1802, Budapest, Hungary, September 2016.

[110] YAZDANPANAH, H., DINIZ, P. S. R. “Recursive least-squares algorithms for sparse system modeling”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2017)*, pp. 3879–3883, New Orleans, LA, USA, March 2017.

[111] WANG, X., POOR, H. V. *Wireless Communication Systems: Advanced Techniques for Signal Reception*. Upper Saddle River, NJ, Prentice Hall, 2004.

[112] HANSLER, E., SCHMIDT, G. *Acoustic Echo and Noise Control: A Practical Approach*. Hoboken, NJ, USA, Wiley, 2004.

[113] BENESTY, J., GANSLER, T., MORGAN, D. R., et al. *Advances in Network and Acoustic Echo Cancellation*. Berlin Heidelberg, Germany, Springer, 2010.

[114] HU, T., CHKLOVSKII, D. B. “Sparse LMS via online linearized Bregman iteration”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2014)*, pp. 7213–7217, Florence, Italy, May 2014.

[115] THEMELIS, K. E., RONTOGIANNIS, A. A., KOUTROUMBAS, K. D. “A variational Bayes framework for sparse adaptive estimation”, *IEEE Transactions on Signal Processing*, v. 62, n. 18, pp. 4723–4736, September 2014.

[116] GIAMPOURAS, P. V., RONTOGIANNINS, A. A., THEMELIS, K. E., et al. “Online Bayesian low-rank subspace learning from partial observations”. In: *European Signal Processing Conference (EUSIPCO 2015)*, pp. 2526–2530, Nice, France, September 2015.

[117] THEMELIS, K. E., RONTOGIANNINS, A. A., KOUTROUMBAS, K. D. “Online Bayesian group sparse parameter estimation using a generalized inverse Gaussian Markov chain”. In: *European Signal Processing Conference (EUSIPCO 2015)*, pp. 1686–1690, Nice, France, September 2015.

[118] ANGELOSANTE, D., BAZERQUE, J. A., GIANNAKIS, G. B. “Online adaptive estimation of sparse signals: where RLS meets the l_1-norm”, *IEEE Transactions on Signal Processing*, v. 58, n. 7, pp. 3436–3447, March 2010.

[119] ANGELOSANTE, D., GIANNAKIS, G. B. “RLS-weighted Lasso for adaptive estimation of sparse signals”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2009)*, pp. 3245–3248, Taipei, Taiwan, April 2009.

[120] VALDMAN, C., DE CAMPOS, M. L. R., APOLINÁRIO, JR., J. A. “RLS sparse system identification using LAR-based situational awareness”. In: *22nd European Signal Processing Conference (EUSIPCO 2014)*, pp. 726–730, Lisbon, Portugal, September 2014.

[121] WIDROW, B., HOFF, M. E. “Adaptive switching circuits”, *IRE WESCON Convention Record*, v. 4, pp. 96–104, 1960.

[122] DINIZ, P. S. R., WIDROW, B. “The History of Adaptive Filters”. In: Maloberti, F., Davies, A. C. (Eds.), *A Short History of Circuits and Systems*, River Publishers, cap. 5, pp. 85–90, Denmark, 2016. ISBN: 978-87-93379-71-8.

[123] WIDROW, B., STEARNS, S. D. *Adaptive Signal Processing*. Englewood Cliffs, NJ, Prentice Hall, 1985.

[124] RUPP, M., HAUSBERG, F. “LMS algorithmic variants in active noise and vibration control”. In: *22th European Signal Processing Conference (EUSIPCO 2014)*, pp. 691–695, Lisbon, Portugal, September 2014.

[125] REBHI, S., BARRAK, R., MENIF, M. “LMS-based digital pre-equalizer for cognitive RoF system”. In: *18th International Conference on Transparent Optical Networks (ICTON 2016)*, pp. 1–4, Trento, Italy, July 2016.

[126] WESTWICK, D. T., MAUNDY, B., SALMEH, R. “Approximate LMS tuning of continuous time filters: convergence and sensitivity analysis”, *IEE Proceedings - Circuits, Devices and Systems*, v. 152, n. 1, pp. 1–6, February 2005.

[127] CIOCHINĂ, S., PALEOLOGU, C., BENESTY, J., et al. “A family of optimized LMS-based algorithms for system identification”. In: *24th European Signal Processing Conference (EUSIPCO 2016)*, pp. 1803–1807, Budapest, Hungary, September 2016.

[128] CANDES, E. J., WAKIN, M. B., BOYD, S. P. “Enhancing sparsity by reweighted l_1 minimization”, *Journal of Fourier Analysis and Applications*, v. 14, n. 5, pp. 877–905, December 2008.

[129] GASSO, G., RAKOTOMAMONJY, A., CANU, S. “Recovering sparse signals with a certain family of nonconvex penalties and DC programming”, *IEEE Transactions on Signal Processing*, v. 57, n. 12, pp. 4686–4698, December 2009.

[130] DINIZ, P. S. R., YAZDANPANAH, H., LIMA, M. V. S. “Feature LMS algorithms”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2018)*, Calgary, Alberta, Canada, April 2018.

[131] WANG, Y.-X., SHARPNACK, J., SMOLA, A., et al. “Trend filtering on graphs”, *Journal of Machine Learning Research*, v. 17, pp. 1–41, April 2016.