ADAPTIVE FILTERING ALGORITHMS AND DATA-SELECTIVE STRATEGIES FOR GRAPH SIGNAL ESTIMATION

Marcelo Jorge Mendes Spelta

Dissertação de Mestrado 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 Mestre em Engenharia Elétrica.

Orientador: Wallace Alves Martins

Rio de Janeiro
Maio de 2019

ADAPTIVE FILTERING ALGORITHMS AND DATA-SELECTIVE STRATEGIES FOR GRAPH SIGNAL ESTIMATION

Marcelo Jorge Mendes Spelta

DISSERTAÇÃO 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 MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Markus Vinícius Santos Lima, D.Sc.

Prof. Paulo Sergio Ramirez Diniz, Ph.D.

Prof. José Antonio Apolinário Junior, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
MAIO DE 2019

Spelta, Marcelo Jorge Mendes

Adaptive Filtering Algorithms and Data-selective Strategies for Graph Signal Estimation/Marcelo Jorge Mendes Spelta. – Rio de Janeiro: UFRJ/COPPE, 2019.

XXIV, 95 p.: il.; 29, 7cm.

Orientador: Wallace Alves Martins

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2019.

Referências Bibliográficas: p. 85 – 92.

1. Graph signal processing. 2. Graph signal estimation. 3. Adaptive filtering. 4. Data-selective algorithms. 5. Set-membership filtering. I. Martins, Wallace Alves. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

Agradecimentos

Agradeço à minha mãe Fátima e ao meu pai Marcelo pelo carinho, companheirismo, atenção e dedicação prestados ao longo de toda a vida e, em particular, durante este período turbulento de mestrado. Graças a vocês estive imerso num ambiente de convivência agradável e estimulante para que desenvolvesse, de forma livre, minhas habilidades e capacidades. As lições que aprendi com vocês nem sempre foram as mais fáceis ou diretas, mas foram essenciais para a formação do indivíduo que sou hoje. Da mesma maneira, agradeço ao meu irmão Lucas pela amizade, conversas e boas risadas em muitos momentos, e mesmo implicância em outros. Sobre este, fiquei bastante surpreso com a sua escolha em cursar engenharia na graduação, afinal, imaginava que ao testemunhar minha experiência pessoal haveria um esperado desestímulo a seguir o mesmo caminho. A dúvida que paira no momento é se, após a conclusão de sua longa graduação, haverá motivação prevista ou imprevista para que prossiga na área de pesquisa, como ocorreu comigo. Apesar de alguns momentos difíceis nestes últimos anos, a existência deste forte ambiente familiar fez com que eu encontrasse forças para continuar mesmo quando os obstáculos pareciam grandes demais para serem transpostos. Por estas, além de muitas outras razões, como o simples fato de suportar o meu humor instável, agradeço enormemente aos três pela paciência e pelo companheirismo.

Ainda sobre família, agradeço o incessante amor e carinho dos meus avós maternos Isolina e Francisco e paternos Eligio e Ilda, sendo que esta última infelizmente nos deixou em 2017, mas a chama de sua presença continua a arder intensamente no fundo dos nossos corações. Meus avós são muito diferentes entre si e, por conta desta complementariedade de personalidades e experiências, com cada um deles tive a oportunidade de aprender algo único: desde a vastidão deste país continental e suas fronteiras, relatada por um antigo caminhoneiro, até a importância e beleza das coisas simples, demonstrada regularmente pela pessoa mais tranquila que conheci. Ao recordar da história dos três gatinhos que perderam as suas luvinhas, faço também uma menção honrosa e agradecimento especial à tia Ligia, que desde sempre nos anima com sua alegria e nos faz ficar acima do peso com suas receitas, e merece a posição conquistada de terceira avó. Agradeço também aos demais membros da minha família dos núcleos do Rio de Janeiro e de Vitória, que tanto contribuíram e

contribuem à minha formação.

Deixando o ambiente familiar, gostaria de lembrar de alguns dos colegas e amigos que fizeram parte de toda a longa caminhada até este momento. Com cada um de vocês aprendi muito e espero que tenha deixado alguma lembrança, mesmo que esta se resuma a algo simples ou bobo. Pelos inesquecíveis anos no imperial Colégio Pedro II - Unidade Centro, agradeço a convivência com Luan, Renato, Vicente e Wallace. A graduação em Engenharia Eletrônica na UFRJ trouxe consigo anos difíceis, mas também momentos relaxados e divertidos com os colegas Caios, Carlos, Douglas, Michel, Raphael Barros, Rayssa e Zeneide. Durante o ano de intercâmbio em Glasgow, agradeço também ao colega de quarto mineiro Gustavo Augusto, que estava sempre disposto a me acompanhar no kebab. Anos mais tarde tive a oportunidade de estagiar na empresa Open Labs, quando conheci os divertidos companheiros de graduação Arthur e Gabriel, aos quais também gostaria de agradecer. Com o mestrado na UFRJ fui apresentado a alguns e conheci melhor outros dos incríveis colegas do laboratório SMT: Felipe, Gabriel, Igor, Lucas, Matheus, Rafael Chaves, Roberto, Tadeu, Vinícius e Wesley. Muito obrigado a todos vocês pela paciência e os momentos compartilhados.

Quanto aos docentes, agradeço a todos os professores do Colégio Pedro II - Unidade Centro, do Curso Técnico de Eletrônica do CEFET/RJ, do Departamento de Engenharia Eletrônica da UFRJ e do Programa de Engenharia Elétrica da COPPE/UFRJ, por contribuírem direta e indiretamente com a minha formação acadêmica, profissional e pessoal. Dentre estes, agradeço especialmente aos professores Carlos José Ribas D'Ávila, Eduardo Vieira Leão Nunes e Wallace Alves Martins. O professor Carlos José me ajudou bastante no início da graduação e passou a me acompanhar desde então, estando sempre disponível para sanar as mais diversas dúvidas ou resolver qualquer pendência de forma bem eficiente. Enquanto isso, o professor Eduardo Nunes foi um grande amigo que me orientou durante o projeto final da graduação e foi o responsável por instigar em mim o desejo de pesquisar e iniciar o mestrado acadêmico. A opção por mudar de ênfase na pós-graduação, migrando de controle para processamento de sinais, foi importante para que assimilasse novos conhecimentos e permitiu que eu me aproximasse e conhecesse melhor meu orientador de mestrado, e grande amigo, Wallace Martins. O professor Wallace confiou em meu potencial desde a primeira troca de mensagens e merece um enorme agradecimento pela excelente orientação durante o desenvolvimento deste trabalho de mestrado. A ajuda dele fez com que eu valorizasse a minha capacidade de produção, que foi refinada através de uma filtragem de conteúdo inicial e valiosas sugestões de redação e síntese que foram incorporadas a artigos submetidos e aceitos em congressos. Além da notável paciência com os longos manuscritos enviados para revisão, o professor Wallace demonstra características inspiradoras a seus alunos,

como o grande apreço pelo conhecimento. Por essas e outras razões, eu o agradeço por este período de amizade e grande aprendizado. Além destes, gostaria de mencionar outros professores de relevância na minha formação: Carlos Teodósio, Jomar Gozzi, Luiz Wagner, Markus Lima, Paulo Diniz, e Sérgio Lima Netto. Agradeço a vocês pelos ensinamentos e o vasto conhecimento transmitido.

Agradeço também a atenção e disponibilidade dos professores Markus Vinícius Santos Lima, Paulo Sergio Ramirez Diniz e José Antonio Apolinário Junior, que gentilmente aceitaram o convite para compor a banca avaliadora deste trabalho.

Por fim, agradeço ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro fornecido durante a produção desta dissertação e ao Programa de Engenharia Elétrica (PEE) da COPPE/UFRJ pela ajuda de custo na participação de congressos acadêmicos.

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

ALGORITMOS DE FILTRAGEM ADAPTATIVA E ESTRATÉGIAS DATA-SELECTIVE PARA ESTIMAÇÃO DE SINAIS EM GRAFOS

Marcelo Jorge Mendes Spelta

Maio/2019

Orientador: Wallace Alves Martins

Programa: Engenharia Elétrica

Dado o potencial do processamento de sinais em grafos (GSP em inglês), um campo de pesquisa recente que estende o processamento de sinais clássico a sinais definidos sobre grafos, esta dissertação explora e propõe novos algoritmos para um problema de GSP que foi recentemente reformulado considerando estratégias de filtragem adaptativa. Após apresentar uma visão geral individualizada de filtragem adaptativa e GSP, este trabalho ressalta a fusão destas áreas quando algoritmos baseados nos métodos *least-mean-square* (LMS) e *recursive least-squares* (RLS) são empregados para estimação em tempo real de sinais em grafos limitados em banda com utilização de um número reduzido de medições ruidosas.

Com a extensão desta ideia, esta dissertação propõe o algoritmo *normalized least-mean-square* (NLMS) para o mesmo contexto de GSP. Conforme a filtragem adaptativa clássica, a técnica NLMS obtida para estimação de sinais em grafos converge mais rapidamente que o algoritmo LMS enquanto é menos complexa que o algoritmo RLS. Análises detalhadas do erro e desvio médio quadráticos em estado estacionário são fornecidas para o algoritmo NLMS proposto, sendo estas também empregadas para complementar análises prévias dos algoritmos LMS e RLS para GSP. Adicionalmente, duas estratégias diferentes de seletividade de dados (DS em inglês) são propostas neste trabalho para reduzir a complexidade computacional geral ao somente calcular atualizações do algoritmo quando o sinal de entrada contém inovação suficiente. Escolhas adequadas de parâmetros de restrição são sugeridas com base na análise destas estratégias de DS, e fórmulas fechadas são derivadas para o cálculo estimado da probabilidade de atualização quando utilizados diferentes algoritmos adaptativos. Por fim, este trabalho apresenta diversas simulações numéricas que corroboram, com elevada acurácia, os resultados teóricos previstos.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

ADAPTIVE FILTERING ALGORITHMS AND DATA-SELECTIVE STRATEGIES FOR GRAPH SIGNAL ESTIMATION

Marcelo Jorge Mendes Spelta

May/2019

Advisor: Wallace Alves Martins

Department: Electrical Engineering

Considering the potential of graph signal processing (GSP), a recent research field that extends classical signal processing to signals defined over graph structures, this dissertation explores and proposes new algorithms to a GSP problem that has been lately recast within the adaptive filtering framework. After presenting an overview of both adaptive filtering and GSP, this work highlights the merging of these areas when algorithms based on the least-mean-square (LMS) and recursive least-squares (RLS) methods are used for the online estimation of bandlimited graph signals (GS) using a reduced number of noisy measurements.

Extending this idea, this dissertation proposes a normalized least-mean-square (NLMS) algorithm for the same GSP context. As in the classical adaptive filtering framework, the resulting NLMS GS estimation technique is faster than the LMS algorithm while being less complex than the RLS algorithm. Detailed steady-state mean-squared error and deviation analyses are provided for the proposed NLMS algorithm, and are also employed to complement previous results on the LMS and RLS algorithms. Additionally, two different data-selective (DS) strategies are proposed to reduce the overall computational complexity by only performing updates when the input signal brings enough innovation. Proper definitions of constraint parameters are given based on the analysis of these DS strategies, and closed formulas are derived for an estimate of the update probability when using different adaptive algorithms. At last, this work presents many numerical simulations corroborating, with high accuracy, the theoretical results predicted.

Contents

List of Figures

List of Tables

List of Symbols

$\widehat{v_n v_m}$	Edge linking graph vertices v_n and v_m, p. 40
$C_{\text{opt}}(\cdot)$	Cost-function for computing the optimal CV for the SM-PAPA, p. 32
$C_{\text{r}}(\cdot)$	Cost-function of the classical RLS algorithm, p. 12
L	Number of previous data, besides the current one, considered in data reuse strategy, p. 16
M	Order of the adaptive FIR filter, p. 9
N	Number of nodes in a graph structure, p. 40
N'	Number of graph nodes to be sampled, p. 50
P	Number of frequency components used in the GSP estimate of the bandlimited graph signal, p. 74
P_{up}	Algorithm probability of update or update rate, p. 22
T	Threshold value for indicating the existence of edge between graph nodes based on the corresponding edge weight in graph structure inference, p. 43
$\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}$	Difference between estimator $\hat{\boldsymbol{s}}_{\mathcal{F}}$ and original GFT $\mathbf{s}_{\mathcal{F}}$, p. 55
$\Delta\hat{\boldsymbol{x}}_{\text{o}}$	Difference between estimator $\hat{\boldsymbol{x}}_{\text{o}}$ and original GS $\mathbf{x}_{\text{o}}$, p. 55
$\Delta\tilde{e}^2$	Difference between *a posteriori* and *a priori* error vectors for deriving the GSP NLMS algorithm, p. 59
$\Gamma(\cdot)$	Standard gamma function, p. 70
$\Gamma_{\text{i}}(\cdot)$	Upper incomplete gamma function, p. 70
Θ	*Feasibility set* from set-membership filtering, p. 25

$\|\cdot\|_2$	ℓ_2-norm, p. 10
$\|\cdot\|_{\mathbf{G}^{-1}[k]}$	Quadratic norm with respect to the positive definite matrix $\mathbf{G}[k]$, p. 20
α	Step-length parameter given by a line search scheme, p. 33
$\bar{\boldsymbol{\gamma}}_{\mathrm{DS}}$	Threshold vector used in the GSP CW-EC strategy, p. 67
$\bar{\gamma}$	Positive threshold for performing internal updates in algorithms with data selection, p. 22
$\bar{\gamma}_{\mathrm{DS}}$	Threshold value used in the GSP ℓ_2N-EC strategy, p. 67
$\bar{\mathbf{e}}$	Normalized error vector, p. 69
$\bar{\mathbf{x}}_{\mathrm{o}}^{P}$	Estimate using only the P-largest frequency components of the original bandlimited graph signal $\mathbf{x}_{\mathrm{o}}$, p. 74
β_{R}	Forgetting factor of the GSP RLS algorithm, p. 53
β_{r}	Forgetting factor of classical RLS algorithm, p. 12
$\boldsymbol{\gamma}$	Constraint vector, p. 27
$\boldsymbol{\gamma}^{+}$	Constraint-vector update value at the end of an internal iteration of an algorithm using the GP method, p. 33
$\boldsymbol{\gamma}^{\mathrm{c}}$	Cauchy point constraint-vector value at an internal iteration of an algorithm using the gradient projection method, p. 33
$\boldsymbol{\gamma}_0$	Constraint-vector value at the begin of an internal iteration of an algorithm using the gradient projection method, p. 33
$\boldsymbol{\gamma}_{\mathrm{opt}}$	Optimal constraint-vector for the SM-PAPA, p. 31
$\boldsymbol{\gamma}_{\mathrm{sc}}$	Simple-choice constraint-vector, p. 28
$\boldsymbol{\gamma}_{\mathrm{tc}}$	Trivial-choice constraint-vector, p. 27
$\boldsymbol{\lambda}_{\mathrm{N}}$	Lagrange multipliers for solving the GSP NLMS convex problem, p. 94
$\boldsymbol{\lambda}_{\mathrm{ap}}$	Lagrange multipliers for solving the classical AP convex problem, p. 16
$\boldsymbol{\lambda}_{\mathrm{papa}}$	Lagrange multipliers for solving the classical PAPA convex problem, p. 20

$\boldsymbol{\lambda}_{\text{sm}}$ Lagrange multipliers for solving the classical SM-PAPA convex problem, p. 93

$\boldsymbol{\varepsilon}$ *A posteriori* error vector for the adaptive GS estimation context, p. 58

$\boldsymbol{\varepsilon}_{\text{dr}}$ *A posteriori* error signal vector in traditional adaptive filtering context using data reuse, p. 16

$\boldsymbol{\alpha}$ Frequency-domain vector assigned to the GSP error $\mathbf{e}$, p. 63

$\boldsymbol{\alpha}_{\mathcal{F}}^{\text{T}}$ Elements of $\boldsymbol{\alpha}$ indexed by the frequency set $\mathcal{F}$, p. 63

$\boldsymbol{\psi}$ *Exact membership set* from set-membership filtering, p. 24

$\boldsymbol{\psi}^{L}$ *Exact membership set* that only considers the current and L-most recent previous data entries, p. 26

$\boldsymbol{w}$ Noise vector, p. 49

$\boldsymbol{x}_{w}$ Noisy graph signal, p. 49

χ_{k}^{2} Chi-squared distribution with k degrees of freedom, p. 69

δ_{R} Small positive number for preventing numerical instabilities in the GSP RLS algorithm, p. 53

δ_{ap} Small positive number for preventing numerical instabilities in the classical AP algorithm, p. 17

δ_{n} Small positive number for preventing numerical instabilities in the classical NLMS algorithm, p. 15

δ_{papa} Small positive number for preventing numerical instabilities in the classical proportionate AP algorithm (PAPA), p. 19

δ_{r} Small positive number for initializing the $\mathbf{R}_{\text{r}}$ matrix in traditional RLS algorithm evaluation, p. 13

δ_{sm} Small positive number for preventing numerical instabilities in the classical SM-PAPA algorithm, p. 27

$\hat{\mathbf{h}}$ Adaptive Filter coefficients vector in traditional adaptive filtering context, p. 9

$\hat{\mathbf{s}}_{\mathcal{F}}$ Adaptive estimate of $\mathbf{s}_{\mathcal{F}}$, p. 52

$\hat{\mathbf{x}}_{\text{o}}$	Adaptive estimate of bandlimited graph signal $\mathbf{x}_{\text{o}}$, p. 52
κ	Update factor for defining the update rate of GSP data-selective strategies, p. 68
λ_i^{lon}	Longitude of node v_i, p. 73
$\lambda_{\min}^{+}(\cdot)$	Minimum non-negative eigenvalue, p. 51
$\mathbb{E}(\cdot)$	Expected value operation, p. 10
$\mathbb{R}$	Real numbers set, p. 9
$\mathcal{A}(\cdot)$	Active set containing the indices where the vector position is saturated, p. 34
$\mathcal{E}$	Edges set, p. 40
$\mathcal{F}$	Support or frequency set, p. 47
$\mathcal{G}(\mathcal{V}, \mathcal{E})$	Graph structure $\mathcal{G}$ defined by the vertices set $\mathcal{V}$ and edges set $\mathcal{E}$, p. 40
$\mathcal{H}$	*Constraint set* from set-membership filtering, p. 24
$\mathcal{I}$	Set containing all possible input pairs $(\mathbf{x}, d)$, defined for the SMF context, p. 25
$\mathcal{L}_{\text{N}}(\cdot)$	Lagrangian function for solving the GSP NLMS convex problem, p. 94
$\mathcal{L}_{\text{ap}}(\cdot)$	Lagrangian function for solving the classical AP convex problem, p. 16
$\mathcal{L}_{\text{papa}}(\cdot)$	Lagrangian function for solving the classical PAPA convex problem, p. 20
$\mathcal{L}_{\text{sm}}(\cdot)$	Lagrangian function for solving the classical SM-PAPA convex problem, p. 93
$\mathcal{N}$	Set containing integers from 1 up to N ($\{1, 2, ..., N\}$), p. 46
$\mathcal{P}(\cdot)$	Projection function, p. 33
$\mathcal{S}$	Sampling set, p. 49
$\mathcal{V}$	Vertices set, p. 40

μ_{L} Convergence factor of the GSP LMS algorithm, p. 52

μ_{N} Convergence factor of the GSP NLMS algorithm, p. 59

μ_{ap} Convergence factor for the traditional AP algorithm, p. 17

$\mu_{\mathrm{l}_{\mathrm{max}}}$ Maximum value of μ_{l} that results in a convergent traditional LMS algorithm, p. 12

μ_{l} Convergence factor for the traditional LMS algorithm, p. 12

μ_{n} Convergence factor for the traditional NLMS algorithm, p. 15

μ_{papa} Convergence factor for the classical proportionate AP algorithm (PAPA), p. 19

$\nabla_{\hat{\mathbf{h}}}(\cdot)$ Gradient operation with respect to the vector $\hat{\mathbf{h}}$, p. 12

ν_{sm} Ancillary variable of the SM-PAPA computing the proportionate elements g_i, p. 36

$\sigma^2_{e_n}$ Variance of the n^{th} GSP error component e_n, p. 56

$\sigma^2_{w_a}$ Variance for scaling the common noise effect $\mathbf{1}$ in the general covariance matrix $\mathbf{C}_w$ definition, p. 75

$\sigma^2_{w_b}$ Variance for scaling the random noise effect $\mathbf{r}_w$ in the general covariance matrix $\mathbf{C}_w$ definition, p. 75

$\sigma^2_{w_n}$ Variance of the n^{th} element of the noise vector $\boldsymbol{w}$, p. 63

τ Internal parameter of the SM-PAPA implementation for choosing the proportionate elements g_i, p. 35

θ Variance parameter of the Gaussian kernel function for graph structure inference, p. 44

ε *A posteriori* error signal in traditional adaptive filtering context, p. 14

φ_i^{lat} Latitude of node v_i, p. 73

$\mathbf{0}$ Vector of zero components, p. 13

$\mathbf{1}$ Vector with all components equal to 1, p. 75

$\mathbf{A}$ Adjacency matrix, p. 41

$\mathbf{B}_{\mathrm{N}}$ Ancillary matrix for practical implementation of the GSP NLMS algorithm, p. 59

$\mathbf{B}_{\mathrm{R}}$ Ancillary matrix for practical implementation of the GSP RLS algorithm, p. 55

$\mathbf{C}_{w}$ Covariance matrix related to the noise vector $\boldsymbol{w}$, p. 49

$\mathbf{D}_{\mathcal{S}}$ Sampling matrix for sampling the positions indicated by $\mathcal{S}$, p. 48

$\mathbf{G}$ Proportionate matrix used in classical proportionate adaptive filtering algorithms, p. 19

$\mathbf{I}$ Identity matrix, p. 13

$\mathbf{L}$ Laplacian matrix, p. 41

$\mathbf{M}_{\mathrm{L}}$ Extension of the GSP LMS convergence factor μ_{L} for deriving the GSP NLMS algorithm, p. 58

$\mathbf{M}_{\mathrm{N}}$ Ancillary matrix used for simplifying the representation of the GSP NLMS algorithm update expression, p. 60

$\mathbf{M}_{\mathrm{R}}$ Ancillary matrix used for simplifying the GSP RLS algorithm update expression, p. 54

$\mathbf{P}$ Ancillary matrix for analyzing GSP LMS performance, p. 65

$\mathbf{Q}$ Ancillary matrix for analyzing GSP LMS performance, p. 65

$\mathbf{R}_{\mathrm{R}}$ Ancillary matrix for computing GSP RLS update, p. 54

$\mathbf{R}_{\mathrm{r}}$ Ancillary matrix for computing traditional RLS update, p. 13

$\mathbf{S}_{\mathrm{L}}^{*}$ Stationary value of $\mathbf{S}_{\mathrm{L}}$, p. 65

$\mathbf{S}_{\mathrm{N}}$ $\mathbb{E}[\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k]\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}^{\mathrm{T}}[k]]$ when using NLMS update expression, p. 61

$\mathbf{S}_{\mathrm{N}}^{*}$ Stationary value of $\mathbf{S}_{\mathrm{N}}$, p. 61

$\mathbf{S}_{\mathrm{R}}^{*}$ Stationary value of $\mathbf{S}_{\mathrm{R}}$, p. 66

$\mathbf{S}_{\mathrm{sm}}$ Ancillary matrix used for evaluating the optimal CV for the SM-PAPA, p. 31

$\mathbf{U}$ Orthonormal eigenvectors matrix from either $\mathbf{A}$ or $\mathbf{L}$, p. 46

$\mathbf{U}_{\mathcal{F}}$	Matrix of eigenvectors $\mathbf{u}$ indexed by frequency set $\mathcal{F}$, p. 47
$\mathbf{X}_{\mathrm{dr}}$	Input signal matrix in traditional adaptive filtering context using data reuse, p. 16
$\mathbf{Z}$	General positive definite matrix, p. 30
$\mathbf{\Lambda}$	Diagonal eigenvalues matrix from either $\mathbf{A}$ or $\mathbf{L}$, p. 46
$\mathbf{\Phi}$	Interpolation matrix, p. 48
$\mathbf{\Pi}$	Regularization matrix used in the GSP RLS algorithm, p. 53
$\mathbf{b}$	General vector, p. 30
$\mathbf{d}_{\mathrm{dr}}$	Desired signal vector in traditional adaptive filtering context using data reuse, p. 16
$\mathbf{e}$	*A priori* error vector for the adaptive GS estimation context, p. 52
$\mathbf{e}_{\mathrm{dr}}$	*A priori* error signal vector in traditional adaptive filtering context using data reuse, p. 16
$\mathbf{h}$	Coefficients vector of a FIR real system to be identified in traditional adaptive filtering context, p. 10
$\mathbf{p}_{\mathrm{R}}$	Ancillary vector for computing GSP RLS update, p. 54
$\mathbf{p}_{\mathrm{r}}$	Ancillary vector for computing traditional RLS update, p. 13
$\mathbf{r}_w$	Realization of a random vector whose entries follow a uniform distribution between $[0, 1]$, p. 75
$\mathbf{s}$	GFT of a graph signal $\mathbf{x}_{\mathrm{G}}$, p. 46
$\mathbf{s}_{\mathcal{F}}$	Vector formed by the positions of the GFT $\mathbf{s}$ indexed by the frequency set $\mathcal{F}$, p. 47
$\mathbf{u}$	Orthonormal eigenvector of either $\mathbf{A}$ or $\mathbf{L}$, p. 46
$\mathbf{u}_{n_{\mathcal{F}}}^{\mathrm{T}}$	n^{th} row of $\mathbf{U}_{\mathcal{F}}$, p. 55
$\mathbf{w}$	Realization of the noise vector $\boldsymbol{w}$, p. 49
$\mathbf{x}$	Input vector in traditional adaptive filtering context, p. 9
$\mathbf{x}_w$	Realization of the noisy graph signal $\boldsymbol{x}_w$, p. 49

$\mathbf{x}_{\mathcal{S}}$	Graph signal sampled at indices determined by $\mathcal{S}$, p. 48
$\mathbf{x}_{\mathrm{G}}$	Graph signal, p. 42
$\mathbf{x}_{\mathrm{o}}$	Bandlimited graph signal, p. 47
$\mathbf{y}$	Filter output in traditional adaptive filtering context, p. 9
$\mathbf{y}_{\mathrm{dr}}$	Filter output vector in traditional adaptive filtering context using data reuse, p. 16
a_{nm}	Weight indicating the strength of the link between nodes v_n and v_m, p. 41
d	Desired signal in traditional adaptive filtering context, p. 9
$d_{\mathrm{E}}(v_i, v_j)$	Euclidean distance between vertices v_i and v_j, p. 42
$d_{\mathrm{H}}(v_i, v_j)$	Haversine distance between nodes v_i and v_j, p. 73
e	*A priori* error signal in traditional adaptive filtering context, p. 9
e_n	n^{th} component of the GSP error $\mathbf{e}$, p. 55
g_m	Elements of the matrix $\mathbf{G}$ used in classical proportionate adaptive filtering algorithms, p. 18
r_{E}	Approximate Earth radius used in Haversine formula, p. 73
v	Graph vertex, p. 40
$\binom{N}{N'}$	Combination operation of choosing N' elements from a total of N, p. 50
$\mathrm{erf}(\cdot)$	Error function, p. 68

List of Abbreviations

ℓ_2N-EC	ℓ_2-norm error constraint, p. 67
GSP_A	GSP framework based on ASP, p. 45
GSP_L	GSP framework based on the eigendecomposition of graph characteristic matrices, p. 45
MSD_G	Mean-squared deviation for the GSP context, p. 49
SD_G	Squared deviation for the GSP context, p. 50
AC-LMS	Adaptive censoring least-mean-square, p. 22
AC-RLS	Adaptive censoring recursive least-squares, p. 22
AP	Affine projection, p. 15
ARE	Average reconstruction error, p. 74
AR	Auto-regressive, p. 15
ASP	Algebraic signal processing, p. 39
CDF	Cumulative distribution function, p. 70
CV	Constraint vector, p. 27
CW-EC	Component-wise error constraint, p. 67
DR	Data reuse, p. 15
DS	Data-selective, p. 3
FIR	Finite-duration impulse response, p. 8
FLOPs	Floating-point operations, p. 61
FoM	Figure of merit, p. 10
FoMs	Figures of merit, p. 3

GFT	Graph Fourier transform, p. 46
GP	Gradient projection, p. 32
GSP	Graph signal processing, p. 2
GS	Graph signal, p. 2
IGFT	Inverse graph Fourier transform, p. 46
IIR	Infinite-duration impulse response, p. 8
IPAP	Improved proportionate affine projection, p. 19
IPNLMS	Improved proportionate normalized least-mean-square, p. 18
IP	Interior-points, p. 32
IoT	Internet of things, p. 1
KKT	Karush–Kuhn–Tucker, p. 35
LMS	Least-mean-square, p. 2
MSD	Mean-squared deviation, p. 3
MSE	Mean-squared error, p. 3
NLMS	Normalized least-mean-square, p. 3
PAPA	Proportionate affine projection algorithm, p. 19
PAP	Proportionate affine projection, p. 18
PNLMS	Proportionate normalized least-mean-square, p. 18
QP	Quadratic programming, p. 32
RE	Relative error, p. 78
RLS	Recursive least-squares, p. 2
RV	Random variable, p. 68
SC-CV	Simple-choice constraint vector, p. 28
SM-AP	Set-membership affine projection, p. 25
SM-NLMS	Set-membership normalized least-mean-square, p. 25

SM-PAPA	Set-membership proportionate affine projection algorithm, p. 22
SM-PAP	Set-membership proportionate affine projection, p. 26
SM-PNLMS	Set-membership proportionate normalized least-mean-square, p. 26
SMF	Set-membership filtering, p. 23
SM	Set-membership, p. 22
TC-CV	Trivial-choice constraint vector, p. 27
WLS	Weighted least-squares, p. 12

Chapter 1

Introduction

1.1 The Inspiration for Graph Signal Processing

Although classical signal processing [1] provides a useful framework for handling data defined on regular domains, such as time series and image grids, some modern problems are better represented by more general structures. For instance, many irregular datasets arise from popular research fields, such as internet of things (IoT) and big data applications, and usually require performing processing tasks on their irregular domains [2, 3]. Moreover, nowadays many aspects of human life are being recorded at all levels: from personal data through health monitoring devices to financial and banking data, social networks, shopping preferences, mobility patterns, among an extensive list of applications that is constantly increasing [4]. Though this huge amount of data might be processed individually based on standard tools for revealing hidden information, neglecting their dependence results possibly in a narrow understanding of a larger process, which suggests that the implementation of a more general approach has the potential to provide a wider knowledge about the considered data. While most of the scientific community adopts the generic machine learning framework for handling these scenarios, in order to model properly these data points and their complex interactions in lower dimensional cases, the signal processing community has recently turned its attention to a general mathematical structure called graph [5], which is commonly used for representing networks, and has been trying to extend some classical signal processing results to this irregular-structure domain [6, 7].

As a simple example, by considering any generic social network environment [4], a possible model for it consists in taking this network users as nodes of a graph structure, while their connections are represented by edges which may present different weights depending on the closeness of these individuals. The attributes of an individual in a social network, such as its political preferences or academic level,

or of a geographical location in a weather station network, like the temperature or solar irradiance, can be represented as signals on the underlying graph structure, where the final goal is to extract useful information from it. From a signal processing perspective [1], doing so requires extending classical concepts and tools, such as Fourier transform [6, 7], filtering [6, 8], sampling [9–12], reconstruction [13–21], and frequency response [8], to data residing on graphs, which leads us to the promising research field of graph signal processing (GSP) [4, 6, 7].

The most natural application of GSP, which is further studied in this dissertation, is in the sensor network context where the graph structure represents the relative distance between distributed sensors and the processing goals include denoising, compression, and reconstruction of sensor data. Since measurements from neighboring nodes usually lead to smooth (low-pass) graph signals, based on a GSP framework outliers or abnormal values can be detected by employing a simple procedure of high-pass filtering and thresholding. Moreover, this smooth feature is also explored for providing compact representations of graph signals, which leads to useful savings in data storage and energy resources when transmitting data. Besides other interesting applications in sensor networks, such as using wavelets on graphs to detect disruptive traffic events like congestion [22], GSP has reached alternative research areas [4], finding application in biological systems known to have a network structure, like the human brain [23], providing a more general approach to the compression of regular structures in image processing [24], and suggesting a comprehension on how information propagates in a network [25].

1.2 Adaptive Filtering Ideas for a GSP Problem

In particular, an interesting GSP application for sensor networks is the online reconstruction method proposed in [15], suggesting the use of a procedure based on the least-mean-square (LMS) algorithm [26–29] for a bandlimited graph signal (GS) estimation context, which represents the first attempt to merge the GSP field with the well-established adaptive filtering area [28, 29]. This blending work is further extended in [16–21], where the authors present centralized and distributed GS reconstruction strategies, consolidating the use of adaptive methods in GSP. Besides the LMS algorithm, centralized and distributed versions of the recursive least-squares (RLS) algorithm [28, 30] are also proposed for the same reconstruction scenarios, suggesting that the adoption of other adaptive filtering algorithms might provide satisfying results. For a better understanding of the adaptive filtering ideas that inspired previous algorithms for graph signal estimation, and most of the contributions of this dissertation, two supporting chapters about this subject are included in this work.

As expected, the centralized GSP LMS and RLS-based algorithms proposed so far present a clear resemblance to their original implementations, which comes from the similarity of their mathematical derivation, and produce a well-known behavior in terms of convergence speed and computational complexity. Like their classical counterparts [28, 29], the LMS algorithm from [15] is much slower than the centralized RLS in [18, 19], as verified in [31] for both static and time-varying reference graph signals. On the other hand, the RLS main drawback is its large computational complexity, which imposes a practical problem in applications with many nodes. Considering these characteristics and the algorithms evolution in the adaptive filtering area [29], one expects that a normalized LMS (NLMS)-based method might provide an interesting trade-off between the current algorithms. Thus, aiming at proposing an adaptive technique for GS estimation that has faster convergence speed than the LMS, while inducing a lower computational burden than the RLS, this dissertation proposes a GSP NLMS algorithm. In particular, this method assumes a simpler form than its classical version [28, 29, 32, 33] due to some particularities of the GS estimation context, resulting in an algorithm whose convergence is faster than the LMS in [15], but with the very same complexity.

Besides its derivation, this work provides a detailed theoretical analysis on the steady-state behavior of the proposed NLMS algorithm, presenting a range of values for the underlying convergence factor that guarantee the algorithm stability and ability to provide asymptotically unbiased GS estimates, as well as closed-form expressions for estimating the corresponding mean-squared error (MSE) and mean-squared deviation (MSD) . Moreover, based on the NLMS study, we employ the same methodology and further complement the LMS and RLS analyses in [19] by obtaining more general steady-state expressions for the related figures of merit (FoMs).

Additionally, an important concern in several practical applications is power consumption. As some classical adaptive filtering algorithms implement the data-selection idea in order to obtain power savings [34–36], this dissertation also discusses adapting this idea for the GSP context. Although some data-selection adaptive algorithm families like the set-membership [34, 35] perform more complex computations involving the estimation of a solution set, in this first approach we propose the use of a simpler family called data-selective (DS) [36–38], which involves a more direct implementation based on a point update. Since the GS estimation problem presents some differences in comparison to the classical adaptive filtering problem, the novelty test at each algorithm iteration uses an error vector instead of a scalar value as in [36]. Thus, two novelty tests are proposed: one based on the individual error component values, namely the component-wise error constraint strategy, and another one that uses the vector squared ℓ_2-norm as a reference, the so-called ℓ_2-norm

error constraint strategy. Constraint parameters and update factors are suggested for both DS strategies in order to provide accurate estimates of the update probability when using not only the proposed GSP NLMS, but also the GSP LMS and RLS algorithms [19].

1.3 Contributions

The main contributions of this dissertation are:

- Proposing a GSP NLMS algorithm, along with a complete analysis that guarantees the algorithm convergence to an unbiased solution, verifies its reduced computational complexity, and provides formulas for predicting the steady-state behavior of useful error metrics;
- Complementing stationary results previously published in the literature for the GSP LMS and RLS algorithms by extending the proposed GSP NLMS steady-state analysis;
- Proposing data-selective strategies to be implemented along the GSP adaptive algorithms in order to reduce their overall computational complexity. Based on the predicted stationary error metrics for the GSP LMS, RLS, and NLMS algorithms, this dissertation suggests particular parameters that result in a simple relation between an update factor and the obtained update rate.

Additionally, some minor contributions of this work are:

- Proposing an optimal procedure for evaluating the constraint vector of the set-membership proportionate affine projection algorithm. Additionally, a faster method for computing this optimal variable is suggested [39];
- Introducing the emergent area of graph signal processing and illustrating some of its concepts and framework tools based on didactic examples. Presenting the scenario and inspiration for merging the adaptive filtering and GSP areas, along with the adaptive GSP strategies proposed so far in the literature;
- Proposing a simple and useful GSP dataset [40] based on temperature measurements obtained from Brazilian weather stations [41] and verifying that these graph signals are approximately bandlimited [31];
- Sharing all MATLAB scripts and results obtained in the simulation scenarios presented in this work. This material is available at [42].

1.4 Publications

This section summarizes the published works and submitted papers which resulted from the contributions of this dissertation.

- Spelta, M. J. M., Martins, W. A.: Optimal Constraint Vectors for Set-Membership Proportionate Affine Projection Algorithms. In: "*2018 IEEE Statistical Signal Processing Workshop (SSP)*", pp. 523-527. Freiburg, Germany (2018).

- Spelta, M. J. M., Martins, W. A.: Online Temperature Estimation using Graph Signals. In: "*XXXVI Simpósio Brasileiro de Telecomunicações e Processamento de Sinais - SBrT2018*", pp. 154-158. Campina Grande, Brazil (2018).

- Submitted journal paper containing the contributions of Chapter 5, which are the proposition and analysis of the NLMS algorithm and the DS strategies for estimating bandlimited graph signals.

1.5 Organization

The text is organized as follows: Chapter 2 presents a general overview of classical adaptive filtering, focusing on the inspiration behind the main adaptive algorithms and their corresponding mathematical derivations. Besides the popular LMS, RLS, and NLMS algorithms, this chapter also covers methods including the data reuse idea, which intends to improve the convergence speed by considering previous data in correlated input scenarios, such as the AP algorithm and its proportionate version (PAP algorithm). In particular, the PAP algorithm aims to increase even further the convergence speed of the data reuse implementation by exploring the approximately sparse characteristic, i.e., the dominance of a few large components in a coefficients vector, in some system identification applications.

Since most classical adaptive filtering algorithms require evaluating a new update at each iteration, and this procedure might be useless when the input data does not bring enough novelty to the current system state, Chapter 3 delves into two data selection approaches for reducing the overall complexity of adaptive algorithms: the data-selective and the set-membership strategies. While the simple DS approach is further explored in later chapters, when the data selection idea is implemented along GSP adaptive algorithms, the SM approach is described for presenting the particular SM-PAPA. Based on a recent work, at the end of Chapter 3 this dissertation proposes a consistent method for obtaining the optimal constraint vector for the SM-PAPA

[39], which is conveniently computed by implementing a fast algorithm relying on the gradient projection method.

Chapter 4 starts the discussion about the emergent field of graph signal processing by defining basic concepts, highlighting its potential, and using didactic examples to illustrate some framework tools and properties, such as the graph Fourier transform and the graph frequency representation, respectively. This chapter also introduces the relevant problem of online estimating bandlimited graph signals, which combines the areas of adaptive filtering and GSP, and presents the GSP LMS and RLS algorithms, recently suggested in the literature.

Inspired by the ideas and derivations discussed in previous chapters, the main contributions of this dissertation are presented in Chapter 5, where the GSP NLMS algorithm and two data-selective strategies for GSP adaptive algorithms are proposed. Besides its mathematical derivation, we verify the necessary conditions for guaranteeing the algorithm convergence, compare the proposed GSP NLMS with the GSP LMS and RLS methods in terms of computational complexity, and describe a complete steady-state analysis that provides closed formulas for evaluating useful error metrics. This analysis is successfully extended in this work to cover the previously described GSP LMS and GSP RLS algorithms, and complements results published in the literature. Moreover, besides proposing the data-selective strategies for reducing the overall computational complexity of GSP adaptive algorithms, Chapter 5 also suggests particular choices of the strategy internal parameters in order to obtain reasonable estimates of the algorithm update rate after reaching its stationary state.

Based on temperature measurements acquired from hundreds of weather stations distributed around a large area, Chapter 6 provides a useful dataset for performing GSP numerical experiments and verifies that, as expected due to their spatial correlation, the presented temperature graph signals are approximately bandlimited. By using these values, many numerical simulations are performed in Chapter 6 corroborating the theoretical results predicted in Chapter 5 for different noise scenarios.

At last, Chapter 7 concludes this work by drawing some final remarks and suggesting potential future research directions.

1.6 Notation

The notation used throughout this dissertation is as follows: vectors and matrices are represented in boldface with lowercase and uppercase letters, such as in $\mathbf{b}$ and $\mathbf{Z}$, respectively. The notation $\text{diag}(\mathbf{b})$ stands for the diagonal matrix composed by the elements of $\mathbf{b}$, $\hat{\mathbf{b}}$ corresponds to an estimate of vector $\mathbf{b}$, $(\cdot)^{\mathrm{T}}$ denotes the transpose operation, and $\mathbf{1}$ indicates a vector where all elements are equal to one.

The representations $\|\cdot\|_{\mathbf{Z}}$, $\|\cdot\|_2$, and $\|\cdot\|_\infty$ stand for the quadratic norm with respect to the positive definite matrix $\mathbf{Z}$ [43], ℓ_2-norm, and infinite-norm, respectively.

The symbols $\mathbb{N}$ and $\mathbb{R}$ denote the set of natural and real numbers, respectively, while the stylized letters, like $\mathcal{V}$ and $\mathcal{E}$, usually represent sets. The exceptions to the latter notation are the letters $\mathcal{G}$, employed for representing graph structures, and $\mathcal{L}$ and $\mathcal{P}$, used for describing the Lagrangian and projection functions, respectively. The cardinality of a generic set $\mathcal{A}$, i.e., the number of elements of this set, is written as $|\mathcal{A}|$. For stochastic processes, the vector $\boldsymbol{b}$ (in italic boldface) stands for a random vector, whose realization is given by $\mathbf{b}$ and the notation $\mathbb{E}(\cdot)$ represents the expected value operation.

Chapter 2

Adaptive Filtering

In the signal processing context, filtering is an operation for extracting and manipulating the original information from an input signal, and mapping it into an output signal [29]. This operation is implemented by a system called *filter*, which is usually parameterized by a set of coefficients, or parameters, that are selected in order to fulfill the designated processing task, usually given as prescribed specifications. As many scenarios assume static and well-defined systems, for meeting a set of constant specifications the literature suggests many design techniques [1] which result in filters with fixed (time-invariant) coefficients. However, in some situations the specifications are not known or are not satisfied by time-invariant filters, thus requiring the use of filters with parameters that change their values according to a performance metric [29]. These time-varying structures are the so-called adaptive filters.

Adaptive filters are completely specified in terms of three characteristics: the application type, its filter implementation structure, and the selected algorithm for performing the coefficients update. The type of application depends on the problem setup, where some popular examples include system identification, channel equalization, echo cancellation, and signal enhancement. Although many adaptive filtering works discuss these traditional scenarios, alternative approaches suggest new potential applications based on the original adaptive concept, such as the online reconstruction of bandlimited graph signals [15], which is further explored in this dissertation.

The filter structure is related to the realization chosen for implementing the adaptive filter. For example, by restricting ourselves to digital filter realizations we find two major classes: finite-duration impulse response (FIR) and infinite-duration impulse response (IIR) filters. Though adaptive IIR filters seem a reasonable choice for applications handling IIR systems, they are not as popular as their FIR counterpart because they suffer from potential instabilities caused by updating a filter pole to a region outside the unity circle [44]. On the other hand, adaptive FIR filters

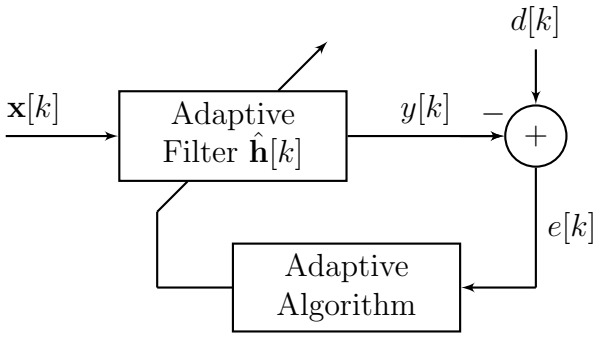


Figure 2.1: General setup of an adaptive-filtering environment.

only involve updates on the system zeros, guaranteeing that the current system is always stable. Therefore, most of the literature studies adaptive FIR filters and the scope of this dissertation is restricted to this structure of digital filters, which gives us the general adaptive-filter configuration displayed in Figure 2.1.

Considering that the adaptive FIR filter has order M, it is characterized by a set of $M+1$ time-varying real coefficients $\{\hat{h}_0[k], \hat{h}_1[k], ..., \hat{h}_{M-1}[k], \hat{h}_M[k]\}$ that are compactly represented by the parameter vector $\hat{\mathbf{h}}[k] \in \mathbb{R}^{M+1}$, where

$$\hat{\mathbf{h}}[k] = \begin{bmatrix} \hat{h}_0[k] & \hat{h}_1[k] & \cdots & \hat{h}_{M-1}[k] & \hat{h}_M[k] \end{bmatrix}^{\mathrm{T}} . \tag{2.1}$$

The filter output $y[k] \in \mathbb{R}$ depends on the filter coefficients $\hat{\mathbf{h}}[k]$ and the input signals at time instant k. If the current system input is given by $x[k]$, for a tapped-delay line the $M+1$ most recent system inputs are represented by the input vector $\mathbf{x}[k] \in \mathbb{R}^{M+1}$ described as

$$\mathbf{x}[k] = \begin{bmatrix} x[k] & x[k-1] & \cdots & x[k-(M-1)] & x[k-M] \end{bmatrix}^{\mathrm{T}} . \tag{2.2}$$

Additionally, in Figure 2.1, the filter output $y[k] = \hat{\mathbf{h}}^{\mathrm{T}}[k]\,\mathbf{x}[k]$ is compared to a reference or desired signal denoted as $d[k] \in \mathbb{R}$ and results in an error signal $e[k] \in \mathbb{R}$ given by

$$e[k] = d[k] - \hat{\mathbf{h}}^{\mathrm{T}}[k]\mathbf{x}[k] \,. \tag{2.3}$$

This error signal is of great importance in adaptive filters since it is the information source of the *Adaptive Algorithm* block in Figure 2.1, indicating how the time-varying coefficients of $\hat{\mathbf{h}}[k]$ must be updated in order to reduce some performance metric considered by the current algorithm.

In particular, the *Adaptive Algorithm* block in Figure 2.1 is the most explored characteristic in adaptive filtering since it provides a wider flexibility for improving the overall performance in some sense. By adjusting the filter coefficients, the adaptive algorithm minimizes a prescribed criterion, represented as an objective function, through a search method whose input is a metric based on the error signal in (2.3).

The algorithm selection is crucial since it determines important aspects of the overall process, like its computational complexity and the existence of biased optimal or suboptimal solutions [29]. Thus, in order to further study this topic, Sections 2.2-2.6 present some traditional adaptive filtering algorithms. However, before delving into this area, let us establish a solid ground for comparing the performance of different adaptive filters.

2.1 Figures of Merit for Adaptive Filters

As there are many different adaptive algorithms, it is useful to first define some performance metrics which will be later used for comparing the algorithms behavior. Based on the instantaneous error in (2.3) and the stochastic nature of adaptive filtering applications, the most commonly used figure of merit (FoM) is the mean-squared error (MSE), described as

$$\mathrm{MSE}[k] = \mathbb{E}[e^2[k]]\,, \tag{2.4}$$

which models how well the adaptive structure is able to produce an output $y[k]$ that resembles the desired signal $d[k]$. Although the MSE in (2.4) is a general metric that provides an overall performance description in any adaptive filtering application, the analysis of some particular aspects can be enhanced by the observation of alternative FoMs. For example, in a system identification application like the one presented in Figure 2.2, where we use an adaptive filter structure with coefficients $\hat{\mathbf{h}}[k]$ for estimating the possibly time-varying real system coefficients $\mathbf{h}[k]$, the metric known as mean-squared deviation (MSD) and given by

$$\mathrm{MSD}[k] = \mathbb{E}\left[\|\hat{\boldsymbol{h}}[k] - \mathbf{h}[k]\|_2^2\right]\,, \tag{2.5}$$

provides a more precise information about how far the current estimate $\hat{\mathbf{h}}[k]$ is from the unknown system parameters. However, the main disadvantage of using the MSD in (2.5) is that it is a theoretical metric with no practical appeal since there is no point in estimating the system vector $\mathbf{h}[k]$ if this vector is known beforehand for evaluating (2.5). On the other hand, the MSE in (2.4) can be estimated in real-time, which is effectively done by some adaptive algorithms, because the system has both the desired signal $d[k]$ and its output signal $y[k]$.

As both the MSE and MSD metrics in (2.4) and (2.5), respectively, consider the expected value of their scalar arguments, for simulation purposes one runs the adaptive algorithm a specific large number of times to form the ensemble, then takes the average of these ensemble runs to provide a better estimate of the expected value.

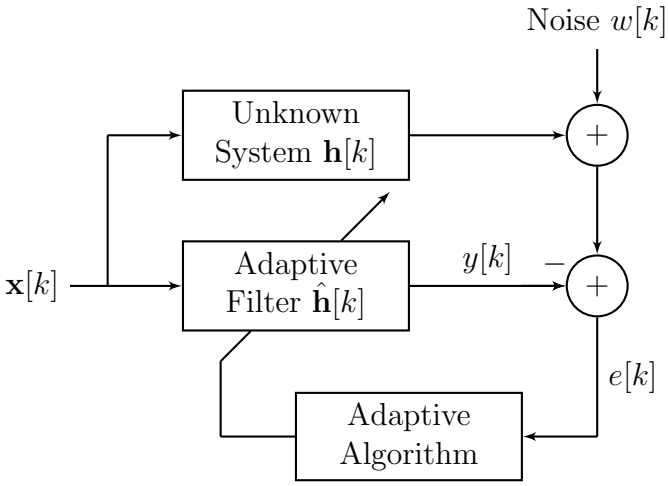


Figure 2.2: Adaptive filter used in a system identification setup.

This procedure relies on the law of large numbers [45], which states that the sample average obtained from a large number of trials becomes closer to the expected value of the underlying random variable as more trials are performed.

The usual behavior of an adaptive algorithm in a stationary environment, in terms of MSE (or MSD), is that it first presents a transient period, then reaches a steady-state behavior. For properly comparing the performance of two or more adaptive strategies, this dissertation considers the approach in which we set the internal parameter of the algorithms so that they present the same steady-state value. When this condition holds, we verify which algorithm first reaches the stationary value and conclude that this method converges more quickly than the others.

Thus, it is expected that one usually searches for the fastest algorithm that produces the same stationary value. However, a typical trade-off for improving the convergence speed of an algorithm is an increase in its complexity, which requires a large number of arithmetic operations performed at each iteration. Then, it is essential to mention that the computational complexity, i.e., the time and memory necessary for performing an algorithm update at each iteration is also a critical concern when designing adaptive filtering algorithms. This trade-off between convergence speed and complexity is at the core of the derivation of adaptive algorithms, as observed in the rest of this work.

2.2 The LMS Algorithm

The most popular adaptive algorithm is the least-mean-square (LMS) [26–29], which basically consists in implementing the stochastic gradient idea for the mean-square error (MSE) defined in (2.4). Since an instantaneous approach is considered, we drop the expected value from expression (2.4) and search for the minimum value of

the squared error in (2.3). For implementing a first-order optimization algorithm we find the gradient $\nabla_{\hat{\mathbf{h}}} e^2[k]$ as

$$\begin{aligned} \nabla_{\hat{\mathbf{h}}} e^2[k] &= -2\,(\,d[k] - \mathbf{x}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k]\,)\,\mathbf{x}[k] \\ &= -2e[k]\,\mathbf{x}[k]\,. \end{aligned} \tag{2.6}$$

Then, following the steepest descent idea, which coincides with the negative gradient direction when considering an objective function based on the Euclidean norm [43], we define a positive convergence factor $\mu_{\mathrm{l}} \in \mathbb{R}_+^*$ and obtain the coefficients update $\hat{\mathbf{h}}[k+1]$ given by

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + 2\mu_{\mathrm{l}}\, e[k]\,\mathbf{x}[k]\,. \tag{2.7}$$

Therefore, the LMS algorithm is implemented by updating the coefficients vector according to equation (2.7), where the only free parameter to be chosen by the system designer is the convergence factor μ_{l}. The role of the factor μ_{l} is modifying the algorithm convergence speed, accelerating it when the convergence factor is increased. However, the price one pays for increasing the convergence speed (reducing convergence time) is that the steady-state error, usually described by the MSE presented in (2.4), is increased. Therefore, the factor μ_{l} indicates a clear trade-off between convergence speed and steady-state error.

Moreover, an important fact is that though μ_{l} must be positive, it cannot assume large values since in these cases the algorithm becomes unstable. In order to obtain a convergent algorithm one must select μ_{l} in the range $]0, \mu_{\mathrm{l}_{\max}}[$, where $\mu_{\mathrm{l}_{\max}} \in \mathbb{R}_+^*$ depends on the eigenvalues of the input signal correlation matrix [29], i.e., the upper limit for the convergence factor is not trivially obtained. Thus, if the designer intends to reduce the LMS algorithm convergence time, it is possible to increase the factor μ_{l} up to the limit $\mu_{\mathrm{l}_{\max}}$, which might be unknown or time-varying.

2.3 The RLS Algorithm

A usual alternative to the simple but slow LMS algorithm is the recursive least-squares (RLS) algorithm [28, 29]. A crucial difference between these two methods is that the LMS searches for an instantaneous solution that minimizes (2.4) while the RLS intends to solve the exponentially weighted least-squares (WLS) problem

$$\underset{\mathbf{h}[k]}{\text{minimize}} \quad C_{\mathrm{r}}(\hat{\mathbf{h}}[k]) = \sum_{i=0}^{k} \beta_{\mathrm{r}}^{k-i} (d[i] - \mathbf{x}^{\mathrm{T}}[i]\hat{\mathbf{h}}[k])^2 \tag{2.8}$$

where $0 \ll \beta_{\mathrm{r}} \leq 1$ is a design parameter called the forgetting factor. Since the RLS cost-function $C_{\mathrm{r}}(\hat{\mathbf{h}}[k])$ is the sum of quadratic scalars weighted by a positive factor, we conclude it is a convex function [43] and the argument $\hat{\mathbf{h}}[k]$ that minimizes it is

obtained by evaluating $\nabla_{\hat{\mathbf{h}}[k]} C_{\mathrm{r}}(\hat{\mathbf{h}}[k]) = \mathbf{0}$. Based on this idea we find the solution

$$\sum_{i=0}^{k} \beta_{\mathrm{r}}^{k-i} d[i]\mathbf{x}[i] = \Big(\sum_{i=0}^{k} \beta_{\mathrm{r}}^{k-i} \mathbf{x}[i]\mathbf{x}^{\mathrm{T}}[i] \Big) \hat{\mathbf{h}}[k] \,. \tag{2.9}$$

For simplification sake, we define $\mathbf{R}_{\mathrm{r}}[k] \in \mathbb{R}^{(M+1)\times(M+1)}$ and $\mathbf{p}_{\mathrm{r}}[k] \in \mathbb{R}^{M+1}$ as

$$\mathbf{R}_{\mathrm{r}}[k] = \sum_{i=0}^{k} \beta^{k-i} \mathbf{x}[i]\mathbf{x}^{\mathrm{T}}[i] \quad \text{and} \quad \mathbf{p}_{\mathrm{r}}[k] = \sum_{i=0}^{k} \beta^{k-i} d[i]\mathbf{x}[i] \,, \tag{2.10}$$

and, from (2.9), if matrix $\mathbf{R}_{\mathrm{r}}$ has full rank we write the solution $\hat{\mathbf{h}}[k]$ of (2.8) as

$$\hat{\mathbf{h}}[k] = \mathbf{R}_{\mathrm{r}}^{-1}[k]\, \mathbf{p}_{\mathrm{r}}[k] \,. \tag{2.11}$$

Although the variables in (2.10) might appear to present a heavy computational burden, they are efficiently evaluated at each time instant by the recursive forms

$$\mathbf{R}_{\mathrm{r}}[k] = \beta_{\mathrm{r}}\, \mathbf{R}_{\mathrm{r}}[k-1] + \mathbf{x}[k]\mathbf{x}^{\mathrm{T}}[k] \quad \text{and} \quad \mathbf{p}_{\mathrm{r}}[k] = \beta_{\mathrm{r}}\, \mathbf{p}_{\mathrm{r}}[k-1] + d[k]\mathbf{x}[k] \,, \tag{2.12}$$

with initial values $\mathbf{R}_{\mathrm{r}}[-1] = \delta_{\mathrm{r}}\, \mathbf{I}$ and $\mathbf{p}_{\mathrm{r}}[-1] = \mathbf{0}$, where $\delta_{\mathrm{r}} \in \mathbb{R}_{+}^{*}$. This initialization of $\mathbf{R}_{\mathrm{r}}$ guarantees that it has full rank and that (2.11) is the solution to the convex problem in (2.8). For additional information about the value of δ_{r} and more efficient implementations of the simple RLS algorithm briefly stated above, as the evaluation of the coefficients vector $\hat{\mathbf{h}}[k]$ in (2.11) based on the recursive expressions from (2.12), refer to [29].

The main advantage of the RLS algorithm is its fast convergence to a steady-state value even when the input signal presents a correlation matrix with a large eigenvalue spread [29], a particular case which considerably slowed down the LMS algorithm convergence speed. However, the most noticeable disadvantage of the RLS algorithm is its large computation complexity, that is easily observed when comparing the RLS update equations in (2.11) and (2.12) to the simple LMS equation in (2.7). Moreover, some implementations of the RLS algorithm suffer from numerical instabilities, such as the ever increasing diagonal elements of matrix $\mathbf{R}_{\mathrm{r}}[k]$ in (2.12). Thus, we verify that the convergence speed improvements brought by the RLS algorithm also brings some undesired drawbacks, like the increased computational complexity and its intrinsic instability. Depending on the application, the designer might require an algorithm faster than the LMS but less complex than the RLS, and this search leads us to analyzing an intermediate class of adaptive algorithms based on the LMS algorithm.

2.4 NLMS Algorithm

Since the LMS algorithm described in Section 2.2 presents a fixed convergence factor μ_l whose main role is controlling the trade-off between convergence time and steady-state error, a reasonable approach for improving the algorithm convergence speed is implementing a time-varying factor $\mu_l[k]$ aimed at minimizing an instantaneous output error [29]. Before explicitly describing the instantaneous metric to be minimized, based on the error signal $e[k]$ in (2.3) we define the *a posteriori* error $\varepsilon[k]$ as

$$\varepsilon[k] = d[k] - \hat{\mathbf{h}}^{\mathrm{T}}[k+1]\mathbf{x}[k]\,. \tag{2.13}$$

From this point on we call $e[k]$ as the *a priori* error whenever necessary for avoiding confusion with the recently defined *a posteriori* error $\varepsilon[k]$. In fact, the only difference between them is that the *a priori* error in (2.3) uses the current coefficient vector $\hat{\mathbf{h}}[k]$ while the *a posteriori* error in (2.13) replaces it with the updated vector $\hat{\mathbf{h}}[k+1]$.

Considering that the update vector $\hat{\mathbf{h}}[k+1]$ is based on the LMS updating equation in (2.7) with possibly time-varying factor $\mu_l[k]$, the *a posteriori* error can be rewritten as

$$\begin{aligned}\varepsilon[k] &= d[k] - \hat{\mathbf{h}}^{\mathrm{T}}[k]\mathbf{x}[k] - 2\mu_l[k]\mathbf{x}^{\mathrm{T}}[k]\mathbf{x}[k]e[k] \\ &= \left(1 - 2\mu_l[k]\mathbf{x}^{\mathrm{T}}[k]\mathbf{x}[k]\right)e[k]\,.\end{aligned} \tag{2.14}$$

As the normalized least-mean-square (NLMS) algorithm defines an instantaneous error metric as $\Delta e[k] = \varepsilon^2[k] - e^2[k]$, from (2.14) we have that

$$\Delta e[k] = 4\mu_l[k]\mathbf{x}^{\mathrm{T}}[k]\mathbf{x}[k]\cdot\left(-1 + \mu_l[k]\mathbf{x}^{\mathrm{T}}[k]\mathbf{x}[k]\right)\cdot e^2[k]\,. \tag{2.15}$$

For minimizing $\Delta e[k]$ we evaluate $\frac{\partial \Delta e[k]}{\partial \mu_l[k]} = 0$ based on (2.15) and obtain

$$\mu_l[k] = \frac{1}{2\,\mathbf{x}^{\mathrm{T}}[k]\mathbf{x}[k]}\,. \tag{2.16}$$

Since this time-varying convergence step depends directly on the input vector inner product $\mathbf{x}^{\mathrm{T}}[k]\mathbf{x}[k] = \|\mathbf{x}[k]\|_2^2$, this relative factor produces a normalizing effect on the update equation. For this reason the current method is known as normalized LMS.

Therefore, according to the initial idea of substituting the original fixed factor by a time-varying convergence step, if we use $\mu_l[k]$ from (2.16) into the LMS update equation (2.7) we obtain an algorithm with faster convergence speed and with update equation given by

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \frac{\mathbf{x}[k]e[k]}{\mathbf{x}^{\mathrm{T}}[k]\mathbf{x}[k]}\,. \tag{2.17}$$

However, as the product $\mathbf{x}^{\mathrm{T}}[k]\mathbf{x}[k]$ in the denominator might assume very small

values, a practical implementation of the NLMS algorithm with equation (2.17) might suffer from numerical instabilities. For overcoming this issue, we add a small term $\delta_{\mathrm{n}} \in \mathbb{R}_{+}^{*}$ that limits the lower value that the denominator of (2.17) can assume. Moreover, a fixed convergence factor $\mu_{\mathrm{n}} \in \mathbb{R}_{+}^{*}$ is usually introduced in (2.17) for controlling the trade-off between convergence speed and steady-state values since the derivations are based on squared error instantaneous values and not on the MSE [29]. Therefore, the practical update equation for the NLMS algorithm is given by

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \frac{\mu_{\mathrm{n}}}{\delta_{\mathrm{n}} + \mathbf{x}^{\mathrm{T}}[k]\mathbf{x}[k]}\mathbf{x}[k]e[k]\,. \tag{2.18}$$

As we observe from its update equation in (2.18), the NLMS algorithm is almost as simple as the original LMS algorithm in (2.7), but considerably less complex than the RLS update expressions in (2.11) and (2.12). In terms of convergence speed, it is usually faster than the LMS algorithm, so it presents an interesting alternative for applications which require reduced convergence time but cannot afford the considerable RLS algorithm complexity. Additionally, an advantage of the NLMS algorithm in comparison to the LMS algorithm is that its convergence factor μ_{n} has the well defined selection range $0 < \mu_{\mathrm{n}} < 2$ (being usually chosen in the range $0 < \mu_{\mathrm{n}} \leq 1$), whereas the LMS factor μ_{l} upper limit depends on the statistics of the input signal [29].

2.5 AP Algorithm

So far we have considered general application environments, in which no specification about the input signals, or the system itself, is previously known. However, as we observe by the dependence between the convergence speed and the eigenvalue spread of the input signal correlation matrix for the LMS algorithm [29], some performance improvements can be obtained by algorithms that take some system characteristics into account. For example, if the system input signal comes from either an auto-regressive (AR) , moving average (MA), or auto-regressive moving average (ARMA) process, its current and past input values present some statistical correlation, then, it makes sense to use the value of previous system variables instead of just discarding them, as the LMS and NLMS algorithms do. Thus, this *data reuse* (DR) concept is very convenient for improving the convergence speed in situations with correlated input signal and forms the base of the so-called affine projection (AP) algorithms.

Since AP algorithms present some slight differences to the traditional adaptive filtering context, it is useful to define convenient system variables that extend the original ideas of input, desired, and error signals. Basically, we just increase the degree of the mathematical structures used for storing data, i.e., vectors and scalars

become matrices and vectors, respectively. For a system that reuses its L previous data values, the input signal $\mathbf{X}_{\mathrm{dr}}[k] \in \mathbb{R}^{(M+1)\times(L+1)}$ is defined as

$$\mathbf{X}_{\mathrm{dr}}[k] = \begin{bmatrix} \mathbf{x}[k] & \mathbf{x}[k-1] & \cdots & \mathbf{x}[k-L] \end{bmatrix}, \tag{2.19}$$

where $\mathbf{x}[k]$ is the current input vector and $\mathbf{x}[k-l]$ is the l^{th} past input entry. By considering the input matrix in (2.19), the vector $\mathbf{y}_{\mathrm{dr}} \in \mathbb{R}^{L+1}$ containing the current and its previous L output values is written as

$$\mathbf{y}_{\mathrm{dr}}[k] = \begin{bmatrix} y[k] & y[k-1] & ... & y[k-L] \end{bmatrix}^{\mathrm{T}} = \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k]\,. \tag{2.20}$$

Likewise, we define the desired signal vector $\mathbf{d}_{\mathrm{dr}}[k] \in \mathbb{R}^{L+1}$ and error signal vector $\mathbf{e}_{\mathrm{dr}}[k] \in \mathbb{R}^{L+1}$ as

$$\begin{aligned} \mathbf{d}_{\mathrm{dr}}[k] &= \begin{bmatrix} d[k] & d[k-1] & ... & d[k-L] \end{bmatrix}^{\mathrm{T}}, \quad \text{and} \\ \mathbf{e}_{\mathrm{dr}}[k] &= \begin{bmatrix} e[k] & e'[k-1] & ... & e'[k-L] \end{bmatrix}^{\mathrm{T}} = \mathbf{d}_{\mathrm{dr}}[k] - \mathbf{y}_{\mathrm{dr}}[k]\,. \end{aligned} \tag{2.21}$$

Then, from (2.20) and (2.21), we rewrite the affine projection *a priori* error vector $\mathbf{e}_{\mathrm{dr}}[k]$ as

$$\mathbf{e}_{\mathrm{dr}}[k] = \mathbf{d}_{\mathrm{dr}}[k] - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k]\,, \tag{2.22}$$

and, based on the *a posteriori* error presented in Section 2.4, we define $\boldsymbol{\varepsilon}_{\mathrm{dr}}[k]$ as

$$\boldsymbol{\varepsilon}_{\mathrm{dr}}[k] = \mathbf{d}_{\mathrm{dr}}[k] - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k+1]\,. \tag{2.23}$$

In general terms, the goal of the AP algorithm is to solve the convex problem [29]

$$\begin{aligned} \underset{\hat{\mathbf{h}}[k+1]}{\text{minimize}} \quad & \frac{1}{2}\|\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\|_2^2 \\ \text{subject to} \quad & \mathbf{d}_{\mathrm{dr}}[k] - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k+1] = \mathbf{0}\,. \end{aligned} \tag{2.24}$$

In other words, the affine projection objective is: by considering the current coefficient vector $\hat{\mathbf{h}}[k]$, the algorithm searches for the closest vector update $\hat{\mathbf{h}}[k+1]$ (minimum disturbance principle) that respects the condition of forcing the *a posteriori* error $\boldsymbol{\varepsilon}_{\mathrm{dr}}[k]$ to be zero.

For solving the original constrained convex problem, we use the method of Lagrange multipliers [43] and rewrite (2.24) as the following function to be minimized

$$\mathcal{L}_{\mathrm{ap}}\big[\hat{\mathbf{h}}[k+1]\big] = \frac{1}{2}\|\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\|_2^2 + \boldsymbol{\lambda}_{\mathrm{ap}}^{\mathrm{T}}[k] \cdot (\,\mathbf{d}_{\mathrm{dr}}[k] - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k+1]\,)\,, \tag{2.25}$$

where $\boldsymbol{\lambda}_{\mathrm{ap}}[k] \in \mathbb{R}^{L+1}$ is the Lagrange multipliers vector for solving the classical AP convex problem in (2.24).

By taking the gradient of (2.25) with respect to $\hat{\mathbf{h}}[k+1]$ and equating this expression to a zero vector we find that

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \mathbf{X}_{\mathrm{dr}}[k]\boldsymbol{\lambda}_{\mathrm{ap}}[k]\,. \tag{2.26}$$

In order to remove $\boldsymbol{\lambda}_{\mathrm{ap}}[k]$ from the above equation, we perform the steps

$$\begin{aligned}
&\mathbf{X}_{\mathrm{dr}}[k]\boldsymbol{\lambda}_{\mathrm{ap}}[k] = \hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\,,\\
&\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{X}_{\mathrm{dr}}[k]\boldsymbol{\lambda}_{\mathrm{ap}}[k] = \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k+1] - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k]\,,\\
&\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{X}_{\mathrm{dr}}[k]\boldsymbol{\lambda}_{\mathrm{ap}}[k] = \mathbf{d}_{\mathrm{dr}}[k] - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k] = \mathbf{e}_{\mathrm{dr}}[k]\,,
\end{aligned} \tag{2.27}$$

and, assuming that $\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{X}_{\mathrm{dr}}[k]$ has full rank, we conclude that

$$\boldsymbol{\lambda}_{\mathrm{ap}}[k] = \big(\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{X}_{\mathrm{dr}}[k]\big)^{-1}\mathbf{e}_{\mathrm{dr}}[k]\,. \tag{2.28}$$

Thus, based on (2.26) and (2.28), the update vector is given by

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \mathbf{X}_{\mathrm{dr}}[k]\big(\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{X}_{\mathrm{dr}}[k]\big)^{-1}\mathbf{e}_{\mathrm{dr}}[k]\,. \tag{2.29}$$

However, similarly to the argument that prevents the use of expression (2.17) in practical implementations of the NLMS algorithm, the update equation (2.29) might be numerically ill-conditioned since the product $(\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{X}_{\mathrm{dr}}[k])$ can be a singular or almost singular matrix. For avoiding this potential problem we slightly modify the original expression (2.29) by adding a positive definite matrix $\delta_{\mathrm{ap}}\,\mathbf{I}$, where $\delta_{\mathrm{ap}} \in \mathbb{R}_+$ is a small number that guarantees a stable computation of the inverse matrix operation. Moreover, a convergence factor $\mu_{\mathrm{ap}} \in \mathbb{R}_+^*$ is included in (2.29) for controlling the trade-off between convergence speed and steady-state MSE. Therefore, the practical update expression for the affine projection algorithm is

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \mu_{\mathrm{ap}}\mathbf{X}_{\mathrm{dr}}[k]\left(\delta_{\mathrm{ap}}\,\mathbf{I} + \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{X}_{\mathrm{dr}}[k]\right)^{-1}\mathbf{e}_{\mathrm{dr}}[k]\,. \tag{2.30}$$

2.5.1 Alternative Derivation of the NLMS Algorithm

Based on the coefficient vector update expressions (2.18) and (2.30), one verifies that the NLMS algorithm is in fact a particular case of the affine projection algorithm when the data reuse factor is $L = 0$. Thus, an interesting outcome from this observation is the existence of an alternative derivation of the NLMS algorithm, which has been originally obtained in Section 2.4 based on the use of a time-varying convergence factor that improves the convergence speed of the LMS algorithm. By following the AP approach suggested in (2.24), it is simple to verify that the partic-

ular case where $L = 0$ leads to the constrained problem

$$\begin{aligned} \underset{\hat{\mathbf{h}}[k+1]}{\text{minimize}} \quad & \frac{1}{2}\|\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\|_2^2 \\ \text{subject to} \quad & d[k] - \mathbf{x}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k+1] = 0\,. \end{aligned} \tag{2.31}$$

Considering the developments involving the Lagrange multipliers method presented in Section 2.4, one finds that the solution of (2.31) is the theoretical NLMS update equation (2.17). As better explained in Section 2.4, for practical applications of the NLMS algorithm we implement expression (2.18) instead of the theoretical equation (2.17).

2.6 Proportionate AP Algorithm

According to the affine projection basic idea, the *a priori* knowledge about the system input statistics can be included into the adaptive algorithm derivation and produce methods with faster convergence speed when the system characteristic agrees with the correlated input assumption. The disadvantage of this concept, besides increasing the computational complexity in comparison with simpler methods like the LMS and the NLMS algorithms, is that we lose some of the generality of the adaptive strategy because AP algorithms are less suitable for handling scenarios with non-colored inputs. However, since the correlated input signal is a known characteristic of many signal processing applications, specific strategies like the AP algorithm are very useful. Following a similar approach, one might require an additional assumption in order to improve even further the system overall converge speed. In particular, the motivation for the proportionate affine projection (PAP) algorithm is that, not only we can deal with correlated input signals, but also that the coefficients vector converges to a vector characterized by a few dominant coefficients.

Originally suggested for reducing the convergence time of the NLMS algorithm in echo cancelers [46, 47], the proportionate method relies on the idea that each coefficient $\hat{h}_m[k]$, $m \in \{0, 1, 2, ..., M\}$, must be updated according to a respective factor $g_m[k] \in \mathbb{R}$ which is heavily dependent on the value of $\hat{h}_m[k]$. Although the name suggests a proportional relation between the coefficient $\hat{h}_m[k]$ and its factor $g_m[k]$, the original proportionate NLMS (PNLMS) work in [46] includes intermediate computations that prevent $g_m[k]$ from being directly proportional to the magnitude of the coefficient $\hat{h}_m[k]$ in some cases. Moreover, due to the undesired PNLMS behavior of being slower than the NLMS in dispersive environments, [47] overcomes this issue by suggesting an improved PNLMS (IPNLMS) algorithm that performs a slightly different set of operations for computing $g_m[k]$. Although this work does not intend to point out the differences between the PNLMS [46] and IPNLMS [47]

rules for selecting the m^{th} proportionate factor $g_m[k]$, or its deviation from a directly proportional relation, its goal is to indicate the clear dependence between this factor and the absolute value of the respective m^{th} adaptive filter coefficient $\hat{h}_m[k]$, where we have $g_m[k] > g_{m'}[k]$ if $|\hat{h}_m[k]| > |\hat{h}_{m'}[k]|$ for $m \neq m'$.

As there is a close relationship between the NLMS and the AP algorithms, as explicitly discussed in Subsection 2.5.1, it is trivial to expect that the PNLMS algorithm can be generalized into a proportionate affine projection (PAP) algorithm [48]. This method intends to improve the overall convergence speed of the AP algorithm when the adaptive filter converges to an approximately sparse vector by including a diagonal proportionate matrix $\mathbf{G}[k] \in \mathbb{R}^{(M+1)\times(M+1)}$ defined as

$$\mathbf{G}[k] = \text{diag}(g_0[k], g_1[k], ..., g_{M-1}[k], g_M[k])\,, \tag{2.32}$$

where $g_m[k] \in \mathbb{R}_+^*$ is the proportionate factor respective to the filter coefficient $h_m[k]$, into the update equation (2.30). Although a practical PAP implementation depends on the rule chosen for obtaining the factors $g_m[k]$ at each iteration, which can be according to the PNLMS approach in [46] or the IPNLMS in [47] and result in the original PAP algorithm [48] or in the improved PAP (IPAP) algorithm [49], respectively, for avoiding confusion this work simply considers the effect of a general diagonal matrix $\mathbf{G}[k]$ on the original AP expression (2.30) and refers to this general scheme as the PAP algorithm (PAPA) . Thus, the practical update equation for the PAP algorithm is given by [48–50]

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \mu_{\text{papa}}\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]\left(\delta_{\text{papa}}\,\mathbf{I} + \mathbf{X}_{\text{dr}}^{\text{T}}[k]\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]\right)^{-1}\mathbf{e}_{\text{dr}}[k]\,, \tag{2.33}$$

where $\mu_{\text{papa}} \in \mathbb{R}_+^*$ is the algorithm convergence factor and δ_{papa} is included for preventing numerical instabilities.

2.6.1 Derivation of the PAP Algorithm

Based on the affine projection constrained convex problem (2.24) whose solution provides the theoretical update equation (2.29), the proportionate affine projection algorithm is defined in a similar way as

$$\begin{aligned} \underset{\hat{\mathbf{h}}[k+1]}{\text{minimize}} \quad & \frac{1}{2}\|\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\|^2_{\mathbf{G}^{-1}[k]} \\ \text{subject to} \quad & \mathbf{d}_{\text{dr}}[k] - \mathbf{X}_{\text{dr}}^{\text{T}}[k]\hat{\mathbf{h}}[k+1] = \mathbf{0}\,, \end{aligned} \tag{2.34}$$

where $\|\cdot\|_{\mathbf{G}^{-1}[k]}$ represents the quadratic norm with respect to the positive definite matrix $\mathbf{G}[k]$ [43]. Then, we have the following equivalence

$$\frac{1}{2}\|\hat{\mathbf{h}}[k+1]-\hat{\mathbf{h}}[k]\|^2_{\mathbf{G}^{-1}[k]}=\frac{1}{2}(\hat{\mathbf{h}}[k+1]-\hat{\mathbf{h}}[k])^{\mathrm{T}}\mathbf{G}^{-1}[k](\hat{\mathbf{h}}[k+1]-\hat{\mathbf{h}}[k])\,. \quad (2.35)$$

By performing similar steps to the ones presented in Section 2.5, we turn (2.34) into an unconstrained problem with objective function

$$\mathcal{L}_{\mathrm{papa}}\big[\hat{\mathbf{h}}[k+1]\big]=\frac{1}{2}\|\hat{\mathbf{h}}[k+1]-\hat{\mathbf{h}}[k]\|^2_{\mathbf{G}^{-1}[k]}+\boldsymbol{\lambda}^{\mathrm{T}}_{\mathrm{papa}}[k]\cdot(\,\mathbf{d}_{\mathrm{dr}}[k]-\mathbf{X}^{\mathrm{T}}_{\mathrm{dr}}[k]\hat{\mathbf{h}}[k+1]\,)\,, \quad (2.36)$$

where $\boldsymbol{\lambda}_{\mathrm{papa}}[k]\in\mathbb{R}^{(L+1)}$ is the Lagrange multipliers vector. After evaluating the gradient of (2.36) with respect to $\hat{\mathbf{h}}[k+1]$ and equating it to $\mathbf{0}$, one finds the expression

$$\mathbf{G}^{-1}[k]\mathbf{h}[k+1]=\mathbf{G}^{-1}[k]\mathbf{h}[k]+\mathbf{X}_{\mathrm{dr}}[k]\boldsymbol{\lambda}_{\mathrm{papa}}[k]\,, \quad (2.37)$$

which, after a pre-multiplication by $\mathbf{G}[k]$, is conveniently represented as

$$\hat{\mathbf{h}}[k+1]=\hat{\mathbf{h}}[k]+\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]\boldsymbol{\lambda}_{\mathrm{papa}}[k]\,. \quad (2.38)$$

Considering similar steps to the mathematical manipulation presented in (2.27), we find that the Lagrangian multipliers vector $\boldsymbol{\lambda}_{\mathrm{papa}}[k]$ is

$$\boldsymbol{\lambda}_{\mathrm{papa}}[k]=(\mathbf{X}^{\mathrm{T}}_{\mathrm{dr}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k])^{-1}\mathbf{e}_{\mathrm{dr}}[k]\,, \quad (2.39)$$

from which we substitute in (2.38) and obtain the PAPA theoretical update equation

$$\hat{\mathbf{h}}[k+1]=\hat{\mathbf{h}}[k]+\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]\big(\mathbf{X}^{\mathrm{T}}_{\mathrm{dr}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]\big)^{-1}\mathbf{e}_{\mathrm{dr}}[k]\,. \quad (2.40)$$

The differences between the theoretical expression (2.40) and the practical update equation (2.33) come from the addition of the $\delta_{\mathrm{papa}}\,\mathbf{I}$ matrix for preventing numerical instabilities and the inclusion of a factor μ_{papa} for controlling the trade-off between convergence speed and steady-state MSE. These practical considerations have also been stated for the NLMS and AP algorithms in Sections 2.4 and 2.5, respectively.

Chapter 3

Data Selection for Adaptive Filtering

As discussed in Chapter 2, an interesting idea for improving the overall performance of a general adaptive filtering algorithm is to take advantage of some prior knowledge of the system or application in which it will be used. For instance, when one employs an adaptive algorithm for a system identification application with correlated input signal, the implementation of the *data reuse* concept in the form of specific algorithms such as the AP and the proportionate AP provides a faster convergence in comparison to the NLMS algorithm. In a similar fashion, one straightforwardly verifies that, after reaching their steady-state in stationary environment, the traditional adaptive algorithms from Chapter 2 keep updating the coefficients vector at each iteration, even though these updates do not produce any improvement over the algorithm estimate. Thus, a reasonable thought is that these unnecessary updates do not need to be evaluated, which results in important power savings for the device that runs the adaptive algorithm [51].

This idea of avoiding the computation of the coefficients vector update at certain iterations is called *data selection* and considers that not all acquired data brings novelty to the current system estimate. For instance, by assuming that the system additive noise is bounded, and that this bound is either known or can be estimated [29], the data-selection idea labels the received information according to its usefulness and updates the algorithm parameters only when the input data is classified as valuable. Therefore, by including simple data tests, the implementation of this concept reduces the algorithm overall complexity since it prevents the complex computation task of evaluating updates of internal parameters at every iteration. The direct implication of this complexity reduction is that the device hosting the adaptive algorithm spends less energy in unnecessary computations, which results in power savings that are extremely useful in battery critic applications, such as some sensor networks or stand-alone IoT devices. Thus, one observes the strong appeal

that the *data selection* concept offers for increasing the lifetime of sensor networks.

This work highlights the implementation of *data selection* along with adaptive filtering based on the data-selective (DS) [36] and set-membership (SM) [34, 35] family of algorithms, which both assess the input data novelty by comparing an absolute error metric to a pre-selected threshold $\bar{\gamma} \in \mathbb{R}_+$. According to this approach, the algorithm parameters are only updated when the absolute error is larger than this threshold, when it considers that the input presents useful data. Then, it follows that the algorithm update rate $P_{\text{up}} \in [0, 1]$ is controlled by properly selecting $\bar{\gamma}$.

This chapter first discusses the difference between the DS and SM *data selection* strategies. The simpler DS method will be recalled in Chapter 5, where we use it for implementing *data selection* strategies along GSP adaptive algorithms, while the more complex SM idea motivates the derivation of the set-membership proportionate affine projection algorithm (SM-PAPA) [52], an algorithm that takes advantage of the *data reuse* and *data selection* ideas in scenarios described by an approximately sparse impulse response and a correlated input signal. The SM-PAPA is further explained in Section 3.2 and draws special attention because this work proposes an optimal procedure for computing an internal parameter of the SM-PAPA based on [53] and suggest a convex problem solver method that improves the evaluation time in comparison to the solver employed in [53]. These contributions of the current dissertation are presented in Section 3.3, and corroborated by numerical simulations in Subsection 3.3.5.

3.1 Data Selection Strategies

3.1.1 DS Adaptive Filtering

Since traditional adaptive algorithms like the ones presented in Chapter 2 perform a coefficients vector update at each iteration, the simplest idea for preventing this procedure is to evaluate an error metric and compute a proper update, according to the usual algorithm update rule, only when the input data is considered to have enough novelty. In opposition, if the error metric does not indicate the presence of useful information, the algorithm simply maintains its current coefficients vector, i.e., $\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k]$.

Although it is based on a trivial procedure, this approach has been adopted recently for the LMS method, with a data-selective LMS algorithm proposed in [54] and later rederived and renamed as the adaptive censoring LMS (AC-LMS) in [37, 38]. The work of [37, 38] also discusses the use of this data-selection strategy along the RLS adaptive estimation, resulting in the so-called adaptive censoring RLS (AC-RLS) algorithm. The implementation of this adaptive censoring strategy

is summarized in Algorithm 1 and consists in updating the algorithm internal variables only when the selected error metric $|e[k]|$ is larger than the threshold value $\bar{\gamma}$, indicating that the current data pair $(\mathbf{x}[k], d[k])$ brings enough novelty to the system.

Algorithm 1 Data-selective strategy original idea

1: $e[k] = d[k] - \hat{\mathbf{h}}^{\mathrm{T}}[k]\mathbf{x}[k]$ % Error metric
2: **if** ($|e[k]| \leq \bar{\gamma}$) **then** % Current pair $(\mathbf{x}[k], d[k])$ brings no novelty
3: Do not update internal variables
4: $\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k]$
5: **else** % Absolute error $|e[k]| > \bar{\gamma}$
6: Update internal variables and $\hat{\mathbf{h}}[k+1]$ according to the original algorithm

As suggested in [36], the censoring scheme described in Algorithm 1 can be improved by including an additional *if* test in order to remove data outliers, which are identified by large values of the instantaneous absolute error $|e[k]|$. Based on this slightly more complex approach, [36] presents advanced versions of the LMS and RLS algorithms incorporating data censorship and proposes a data-selective implementation of the affine projection (AP) algorithm. Although this dissertation considers the simpler censoring method stated in Algorithm 1, disregarding the additional outliers detection step suggested in [36], a clear influence from [36] is the idea of calling the current strategy by the name of data-selective (DS) algorithms instead of employing the adaptive censoring (AC) term as in [37, 38].

Thus, in this work we consider that a data-selective strategy is given by the generic procedure presented in Algorithm 1. Besides its implementation simplicity, another practical advantage of this data selection method over traditional adaptive filtering approaches, such as the set-membership idea discussed in the next section, is that the DS strategy provides a more precise estimation of the algorithm stationary update rate. This improved estimate relies on the $\bar{\gamma}$ parameter selection and provides a useful tool for designing practical systems with energy restrictions.

3.1.2 SM Adaptive Filtering

An alternative to the simplistic data-selective strategy presented in the previous subsection is the set-membership approach [29, 55], which induces a set of solutions at each iteration, searching for an update vector inside an estimated region, while the DS method performs a more straightforward point update. The set-membership filtering (SMF) consists in a general framework for handling linear in parameters adaptive-filtering problems, as the general setup described in Figure 2.1, where the filter output and error signal are given by $\hat{\mathbf{h}}^{\mathrm{T}}[k]\mathbf{x}[k]$ and (2.3), respectively.

The motivation behind the SMF concept is that the requirement of forcing the *a posteriori* error $\varepsilon[k]$ to be zero in algorithms such as the NLMS is often unnecessary since there is usually a certain amount of environmental noise that unjustifies this too-precise approach. Thus, the SMF framework idea is to relax the original *a posteriori* error requirement and, based on the current signals, to search for an updated coefficients vector $\hat{\mathbf{h}}[k+1]$ so that $|\varepsilon[k]| = |d[k]-\hat{\mathbf{h}}^{\mathrm{T}}[k+1]\mathbf{x}[k]|$ is bounded by a threshold parameter $\bar{\gamma} \in \mathbb{R}_{+}^{*}$, which is related to an estimation of the environment noise. In other words, at each iteration we search for an update $\hat{\mathbf{h}}[k+1]$ inside a region described by the inequality

$$|d[k] - \hat{\mathbf{h}}^{\mathrm{T}}[k+1]\mathbf{x}[k]| \leq \bar{\gamma}\,. \tag{3.1}$$

By considering the current input vector and desired signal as the pair $(\mathbf{x}[k], d[k])$, we define the *constraint set* $\mathcal{H}[k]$ as the set of vectors $\hat{\mathbf{h}}[k+1]$ that satisfy (3.1), i.e.,

$$\mathcal{H}[k] = \{\, \hat{\mathbf{h}}[k+1] \in \mathbb{R}^{M+1} \,:\, |d[k] - \hat{\mathbf{h}}^{\mathrm{T}}[k+1]\mathbf{x}[k]| \leq \bar{\gamma} \,\}\,. \tag{3.2}$$

As the constraint set $\mathcal{H}[k]$ in (3.2) is bounded by the parallel hyperplanes $d[k] - \hat{\mathbf{h}}^{\mathrm{T}}[k+1]\mathbf{x}[k] = \bar{\gamma}$ and $d[k] - \hat{\mathbf{h}}^{\mathrm{T}}[k+1]\mathbf{x}[k] = -\bar{\gamma}$, the search region for the update vector $\hat{\mathbf{h}}[k+1]$ is located between these two hyperplanes in the parameter space $\hat{\mathbf{h}}[k+1]$. However, since each new iteration k brings an updated pair of signals $(\mathbf{x}[k], d[k])$, if we consider all past information about the system in order to improve the search region, we notice that as we increment the number of iterations this search region might be modified because at this moment it is not enough to satisfy the previous k data pairs $(\mathbf{x}[i], d[i])$, where $i \in \{0, 1, ..., k-1\}$, but also the most recent information represented by $(\mathbf{x}[k], d[k])$ and the constraint set $\mathcal{H}[k]$. Therefore, the current suitable region for updating $\hat{\mathbf{h}}[k+1]$ is called the *exact membership set* $\boldsymbol{\psi}[k]$ and is formally represented as

$$\boldsymbol{\psi}[k] = \bigcap_{i=0}^{k} \mathcal{H}[k]\,. \tag{3.3}$$

From the polytope defined in (3.3), it is trivial to verify that $\boldsymbol{\psi}[k] = \boldsymbol{\psi}[k-1] \cap \mathcal{H}[k]$ and conclude that the current estimation of the *exact membership set* $\boldsymbol{\psi}[k]$ can be performed recursively by obtaining the region intersecting both the previous set $\boldsymbol{\psi}[k-1]$ and the region between hyperplanes given by $\mathcal{H}[k]$. Then, two possible scenarios can happen: either (i) $\mathcal{H}[k]$ contains $\boldsymbol{\psi}[k-1]$ ($\mathcal{H}[k] \supseteq \boldsymbol{\psi}[k-1]$) or (ii) it does not ($\mathcal{H}[k] \not\supseteq \boldsymbol{\psi}[k-1]$).

Considering a data-selection perspective, when (i) happens there is no need to update the algorithm estimate of $\boldsymbol{\psi}[k]$ because the $\boldsymbol{\psi}[k-1]$ estimate already satisfies

the new information brought by the data pair $(\mathbf{x}[k], d[k])$. In this case we consider that the input information does not bring enough novelty to the current system, which can maintain its previous estimate and the current coefficients vector ($\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k]$). On the other hand, if (ii) is true we need to restrict the *exact membership set* $\boldsymbol{\psi}[k]$ region even further by taking the intersection between $\boldsymbol{\psi}[k-1]$ and $\mathcal{H}[k]$. The practical outcome of this procedure is that the coefficients vector $\hat{\mathbf{h}}[k+1]$ must be properly updated into the more restricted set $\boldsymbol{\psi}[k]$ in (3.3) because the algorithm decides that the current input pair $(\mathbf{x}[k], d[k])$ brings novelty to the overall system estimate.

Thus, when considering all the previous data pairs $(\mathbf{x}[i], d[i])$ with $i \in \{0, 1, ..., k\}$, as the number of iteration increases we expect to reduce the region determined by (3.3) so that it converges to a theoretical set Θ referred to as the *feasibility set*. In simple terms, this *feasibility set* considers all the possible input pairs $(\mathbf{x}, d)$, denoted as the set $\mathcal{I}$, and defines the set of all possible coefficients vectors $\hat{\mathbf{h}}$ that result in an output bounded by $\bar{\gamma}$ when $(\mathbf{x}, d) \in \mathcal{I}$. Then, the formal definition of the *feasibility set* Θ is given by

$$\Theta = \bigcap_{(\mathbf{x},d)\in\mathcal{I}} \{\hat{\mathbf{h}} \in \mathbb{R}^{M+1} \,:\, |d - \hat{\mathbf{h}}^{\mathrm{T}}\mathbf{x}| \leq \bar{\gamma}\}\,. \tag{3.4}$$

Since all practical input pairs $(\mathbf{x}[i], d[i])$ with $i \in \{0, 1, ..., k\}$ up to a time instant k are elements of the more general set $\mathcal{I}$, it is clear that the *feasibility set* Θ in (3.4) is always a subset of the *exact membership set* $\boldsymbol{\psi}[k]$ in (3.3). Then, the *feasibility set* consists in a theoretical limiting set for the estimated *exact membership set* $\boldsymbol{\psi}[k]$.

Therefore, the SMF general goal is to obtain adaptively an estimate that belongs to the theoretical *feasibility set* Θ. This estimate is based on the practical *exact membership set* $\boldsymbol{\psi}[k]$ update idea, which implements a data-selection scheme that wisely prevents unnecessary updates when the previous set $\boldsymbol{\psi}[k]$ already satisfies the new requirement described by $\mathcal{H}[k]$. However, implementation issues for SM adaptive filtering prevent the use of all data pairs $(\mathbf{x}[i], d[i])$ with $i \in \{0, 1, ..., k\}$ for estimating the *feasibility set*, and practical approaches only consider the most recent data pairs in their estimates. For example, a set-membership NLMS (SM-NLMS) algorithm [55] only considers the current input pair $(\mathbf{x}[k], d[k])$, so its estimate of the *feasibility set* Θ is based solely on the constraint set $\mathcal{H}[k]$. On the other hand, data reuse strategies like the AP and proportionate AP algorithms hold information about previous data pairs, which allows them to perform a better estimation of Θ. In the next section we explore the particular case of the set-membership proportionate affine projection algorithm (SM-PAPA) [52], that generalizes both the SM-NLMS [55] and set-membership affine projection (SM-AP) [34] algorithms.

3.2 The SM-PAP Algorithm

In practical terms, the main difference between the derivation of the traditional NLMS, AP and proportionate AP adaptive algorithms from Chapter 2 and their set-membership (SM) counterparts is that the SM versions require the update vector $\hat{\mathbf{h}}[k+1]$ to belong to the intersection of recent constraint sets $\mathcal{H}[i]$, where $i \in \{k-L, k-(L-1), ..., k-1, k\}$ and L is the data reuse factor. On the other hand, the traditional versions of algorithms NLMS, AP, and PAP solve the respective constrained problems (2.31), (2.24) and (2.34) in which the requirement over $\hat{\mathbf{h}}[k+1]$ is that the *a posteriori* error becomes zero. For example, the convex problem related to the derivation of the set-membership proportionate NLMS (SM-PNLMS) is simply given by [52]

$$\begin{aligned} \underset{\hat{\mathbf{h}}[k+1]}{\text{minimize}} \quad & \|\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\|^2_{\mathbf{G}^{-1}[k]} \\ \text{subject to} \quad & |d[k] - \mathbf{x}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k+1]| \leq \bar{\gamma}\,, \end{aligned} \tag{3.5}$$

where $\mathbf{G}[k]$ is the positive-definite proportional matrix defined in (2.32) and the inequality $|\varepsilon[k]| \leq \bar{\gamma}$ represents the set-membership requirement $\hat{\mathbf{h}}[k+1] \in \mathcal{H}[k]$.

Since data-reuse adaptive algorithms, such as the AP and proportionate AP algorithms, do not rely solely on the current data pair $(\mathbf{x}[k], d[k])$, their estimate of the *feasibility set* Θ is improved by considering the current measurements and its L previous data pairs stored. Thus, the set-membership proportionate affine projection (SM-PAP) algorithm is derived based on the *exact membership set*

$$\boldsymbol{\psi}^L[k] = \bigcap_{i=(k-L)}^{k} \mathcal{H}[k]\,, \tag{3.6}$$

which only considers the current and the L-most recent previous data entries, and is summarized by the equivalent l_∞-norm requirement

$$\|\mathbf{d}_{\mathrm{dr}}(k) - \mathbf{X}^{\mathrm{T}}_{\mathrm{dr}}(k)\hat{\mathbf{h}}[k+1]\|_\infty \leq \bar{\gamma}\,, \tag{3.7}$$

i.e., all the components of the current *a posteriori* error $\boldsymbol{\varepsilon}_{\mathrm{dr}}[k]$ must have absolute value equal or smaller than $\bar{\gamma}$. Then, the SM-PAPA intends to solve the inequality constrained convex problem [52]

$$\begin{aligned} \underset{\hat{\mathbf{h}}[k+1]}{\text{minimize}} \quad & \|\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\|^2_{\mathbf{G}^{-1}[k]} \\ \text{subject to} \quad & \|\mathbf{d}_{\mathrm{dr}}[k] - \mathbf{X}^{\mathrm{T}}_{\mathrm{dr}}[k]\hat{\mathbf{h}}[k+1]\|_\infty \leq \bar{\gamma}\,. \end{aligned} \tag{3.8}$$

As the l_∞-norm constraint upon the *a posteriori* error in (3.7) presents an obstacle for finding an analytic solution to this convex problem, a convenient idea for esti-

mating (3.8) is to approach a related problem where we define a vector $\boldsymbol{\gamma}[k] \in \mathbb{R}^{L+1}$, whose components satisfy $|\gamma_i[k]| \leq \bar{\gamma}$, and replace the inequality constraint in (3.8) by the linear constraint $\mathbf{d}_{\mathrm{dr}}(k) - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k+1] = \boldsymbol{\gamma}[k]$. Therefore, it is clear that the solution $\hat{\mathbf{h}}[k+1]$ to the alternative problem

$$\begin{aligned} &\underset{\hat{\mathbf{h}}[k+1]}{\text{minimize}} \quad \|\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\|^2_{\mathbf{G}^{-1}[k]} \\ &\text{subject to} \quad \mathbf{d}_{\mathrm{dr}}[k] - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k+1] = \boldsymbol{\gamma}[k]\,, \end{aligned} \tag{3.9}$$

which, according to Appendix A.1, is given by the equation

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k] \cdot [\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]]^{-1} \cdot [\mathbf{e}_{\mathrm{dr}}[k] - \boldsymbol{\gamma}[k]]\,, \tag{3.10}$$

satisfy the l_∞-norm inequality in (3.7). However, it does not necessarily solve the original problem in (3.8) because there is no guarantee that the obtained vector $\hat{\mathbf{h}}[k+1] \in \boldsymbol{\psi}^L[k]$ provides the smallest disturbance $\|\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\|^2_{\mathbf{G}^{-1}[k]}$.

By considering the SM data selection idea from Subsection 3.1.2 and the possible numerical instabilities of the term $[\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]]^{-1}$ in (3.10), the SM-PAP practical implementation is based on the following update expression [52]

$$\hat{\mathbf{h}}[k+1] = \begin{cases} \hat{\mathbf{h}}[k] + \mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]\,[\delta_{\mathrm{sm}}\mathbf{I} + \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]]^{-1}[\mathbf{e}_{\mathrm{dr}}[k] - \boldsymbol{\gamma}[k]], & \text{if } |e[k]| > \bar{\gamma}\,, \\ \hat{\mathbf{h}}[k] & \text{, otherwise}\,, \end{cases} \tag{3.11}$$

where δ_{sm} is a small positive value.

When observing (3.11), one notices that the parameter $\boldsymbol{\gamma}[k]$ has not been properly defined yet since we have only stated that $\|\boldsymbol{\gamma}[k]\|_\infty \leq \bar{\gamma}$. In fact, this variable selection is important to the algorithm overall behavior because it leads to a better or worse implementation of the theoretical SM-PAPA problem (3.8). This vector $\boldsymbol{\gamma}[k]$ is the so-called constraint vector (CV) and has been usually selected in the literature according to some heuristics, as in [29, 56–59]. Based on the recent work [53], a different approach is suggested in the next section, where the optimal CV idea is generalized in order to cover the SM-PAPA and provide a particular choice of $\boldsymbol{\gamma}[k]$, called an optimal CV, that improves the performance of the SM-PAPA implementation (3.11) in terms of some FoMs.

3.3 Optimal Constraint Vector for the SM-PAPA

The simplest way to define the constraint vector is by implementing the trivial-choice CV (TC-CV) [29] $\boldsymbol{\gamma}_{\mathrm{tc}}$, where we use $\boldsymbol{\gamma}[k] = \boldsymbol{\gamma}_{\mathrm{tc}} = \mathbf{0}$ and the update equation (3.11) becomes identical to the PAPA expression (2.33) when an algorithm update ($|e[k]| >$

$\bar{\gamma}$) is required. Although very similar to the conventional PAPA, the SM-PAPA with TC-CV provides a considerable reduction in computational complexity and, due to the data-selective strategy considered in Subsection 3.1.1 (without special treatment for outliers), one can conclude that the TC-CV SM-PAPA is equivalent to this data-selective strategy. The main problem with the trivial choice method can be observed in Figure 3.1, where one notices that the update vector $\hat{\mathbf{h}}[k+1]$ is too far from the current vector $\hat{\mathbf{h}}[k]$ and right in the center of region $\boldsymbol{\psi}^1[k] = \mathcal{H}[k] \cap \mathcal{H}[k-1]$. Since the SM-PAPA original problem in (3.8) intends to minimize this distance between current and updated vectors, more complex CV definitions must be considered.

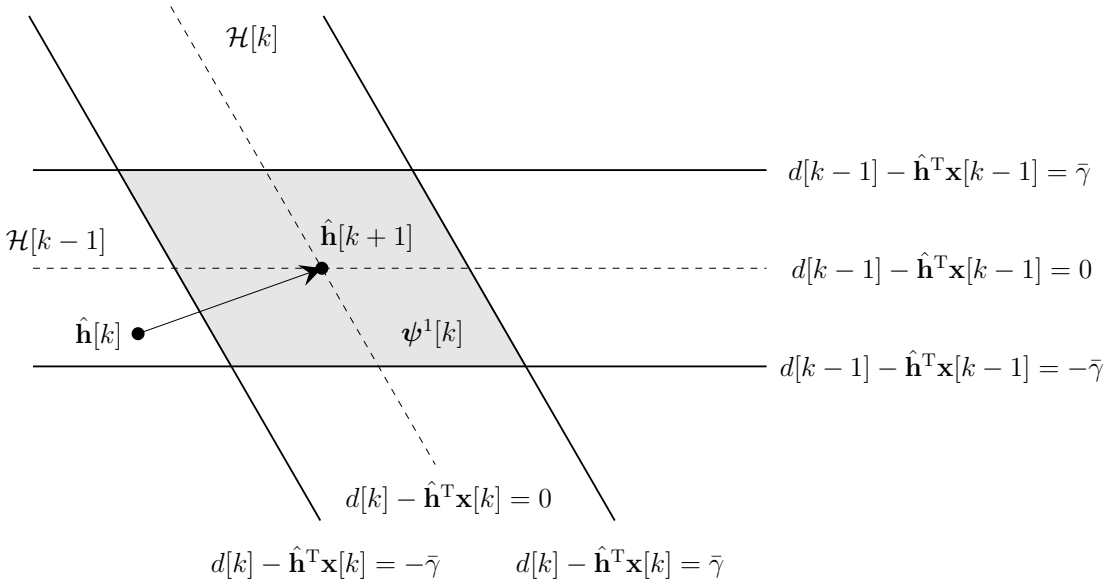


Figure 3.1: TC-CV for the SM-AP (SM-PAPA with $\mathbf{G}[k] = \mathbf{I}$) using $L = 1$.

A reasonably good selection method for defining $\boldsymbol{\gamma}[k]$ is the simple-choice constraint vector (SC-CV) [29], which is easily implemented and usually provides better results than the TC-CV. Based on the theoretical update equation (3.10), the SC-CV method relies on the fact that $\boldsymbol{\varepsilon}_{\mathrm{dr}}[k] = \mathbf{d}_{\mathrm{dr}}[k] - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k+1] = \boldsymbol{\gamma}[k]$ and define the SC-CV $\boldsymbol{\gamma}_{\mathrm{sc}}[k] = [\gamma_{\mathrm{sc}_0}[k] \quad \gamma_{\mathrm{sc}_1}[k] \quad ... \quad \gamma_{\mathrm{sc}_{L-1}}[k] \quad \gamma_{\mathrm{sc}_L}[k]]^{\mathrm{T}}$ as

$$\gamma_{\mathrm{sc}_l}[k] = \begin{cases} \bar{\gamma}\dfrac{e[k]}{|e[k]|} & , \text{ if } l = 0\,, \\ e'[k-l] & , \text{ otherwise}\,, \end{cases} \tag{3.12}$$

where $e[k]$ and $e'[k-l]$ ($l \in \{1, 2, ..., L\}$) represent the components of the current error vector $\mathbf{e}_{\mathrm{dr}}[k]$ in (2.21).

The SC-CV in (3.12) assumes that the current vector $\hat{\mathbf{h}}[k]$ is already inside the set formed by $\cap_{i=k-L}^{k-1}\mathcal{H}[i]$, so it only needs to satisfy the most recent requirement $d[k] - \mathbf{x}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k]$ to be also inside of $\mathcal{H}[k]$. In order to achieve this, it simply places the 0^{th} component of $\boldsymbol{\gamma}_{\mathrm{sc}}[k]$ at the closest border of $\mathcal{H}[k]$, which is represented by

$\gamma_{\mathrm{sc}_0}[k] = \bar{\gamma}\frac{e[k]}{|e[k]|} = \bar{\gamma} \cdot \mathrm{sign}[e[k]]$, as illustrated in Figure 3.2.

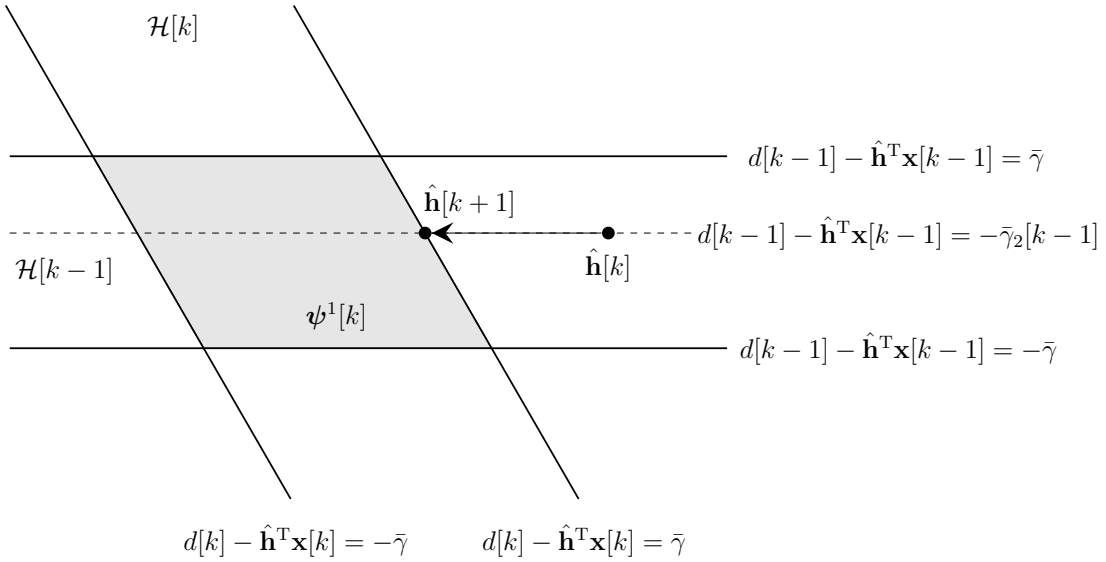


Figure 3.2: SC-CV for the SM-AP (SM-PAPA with $\mathbf{G}[k] = \mathbf{I}$) using $L = 1$.

Like the TC and SC methods, the adaptive filtering literature presents some different heuristic CV choices for the SM-AP and SM-PAPA [56–59]. Among these, the briefly described SC-CV [29, 34] deserves special attention due to its ease of implementation and the fact that it usually induces the best MSE performance. However, it is still an heuristic approach and it considers the theoretical update expression in (3.10) instead of the practical update implementation (with the addition of $\delta_{\mathrm{sm}}\mathbf{I}$ for numerical stability) of (3.11). Thus, motivated by the systematic approach proposed in [53] for obtaining an optimal CV for the SM-AP algorithm, this dissertation proposes a generalization of this method for the SM-PAPA. For understanding this idea, in the next subsection we present the relation between the practical SM-PAPA update equation (3.11) and a convex cost-function, which will be later used as the reference for evaluating the optimal CV.

3.3.1 SM-PAPA Convex Cost-Function

As we have observed, the theoretical solution (3.10) is the vector that solves the alternative constraint problem (3.9), which is an estimate of the original SM-PAPA idea that aims to obtain the solution to the l_∞-norm inequality constraint minimization task in (3.8). However, the addition of $\delta_{\mathrm{sm}}\mathbf{I}$ in the SM-PAPA implementation in (3.11) requires updating results in a deviation from the original problems (3.8) and (3.9). In fact, based on [53], we demonstrate in this work that the practical SM-PAPA update equation in (3.11), when $|e[k]| > \bar{\gamma}$, is equivalent to the optimization

problem

$$\underset{\hat{\mathbf{h}}[k+1]}{\text{minimize}} \quad \|\hat{\mathbf{h}}[k+1]-\hat{\mathbf{h}}[k]\|^2_{\mathbf{G}^{-1}}+\frac{1}{\delta_{\text{sm}}}\|\mathbf{d}_{\text{dr}}[k]-\mathbf{X}^{\text{T}}_{\text{dr}}[k]\hat{\mathbf{h}}[k+1]-\boldsymbol{\gamma}[k]\|^2_2\ . \quad (3.13)$$

As the squared norm $\|\cdot\|^2_2$ function is convex and the sum operation preserves convexity [43], we find that the practical SM-PAPA cost-function in (3.13) is convex, from where we conclude it presents a single minimum point.

In order to prove the relation between (3.11) and (3.13) we first recall that, since $\|\mathbf{b}\|_{\mathbf{Z}} = \|\mathbf{Z}^{1/2}\mathbf{b}\|_2$ for a positive definite matrix $\mathbf{Z}$ and a vector $\mathbf{b}$ with compatible dimension [43], (3.13) can be rewritten considering the l_2-norm as

$$\underset{\hat{\mathbf{h}}[k+1]}{\text{minimize}} \quad \|\mathbf{G}^{-1/2}\big(\hat{\mathbf{h}}[k+1]-\hat{\mathbf{h}}[k]\big)\|^2_2+\frac{1}{\delta_{\text{sm}}}\|\mathbf{d}_{\text{dr}}[k]-\mathbf{X}^{\text{T}}_{\text{dr}}[k]\hat{\mathbf{h}}[k+1]-\boldsymbol{\gamma}[k]\|^2_2\ . \quad (3.14)$$

Thus, the optimization problem (3.13) can be solved analitically by expanding the cost-function expression in (3.14), taking its gradient with respect to $\hat{\mathbf{h}}[k+1]$, and equating the result to zero, which results in

$$\hat{\mathbf{h}}[k+1]=\left[\mathbf{G}^{-1}[k]+\frac{\mathbf{X}_{\text{dr}}[k]\mathbf{X}^{\text{T}}_{\text{dr}}[k]}{\delta_{\text{sm}}}\right]^{-1}\left[\mathbf{G}^{-1}[k]\hat{\mathbf{h}}[k]+\frac{\mathbf{X}_{\text{dr}}[k]}{\delta_{\text{sm}}}(\mathbf{d}_{\text{dr}}[k]-\boldsymbol{\gamma}[k])\right]. \quad (3.15)$$

As the proportionate matrix $\mathbf{G}[k]$ is diagonal and positive definite, one can easily compute $\mathbf{G}^{-1/2}[k]$ such that the inverse matrix in (3.15) is rewritten as

$$\left[\mathbf{G}^{-1}[k]+\frac{\mathbf{X}_{\text{dr}}[k]\mathbf{X}^{\text{T}}_{\text{dr}}[k]}{\delta_{\text{sm}}}\right]^{-1}=\mathbf{G}^{\frac{1}{2}}[k]\left(\mathbf{I}+\frac{1}{\delta_{\text{sm}}}\mathbf{G}^{\frac{1}{2}}[k]\mathbf{X}_{\text{dr}}[k]\mathbf{X}^{\text{T}}_{\text{dr}}[k]\mathbf{G}^{\frac{1}{2}}[k]\right)^{-1}\mathbf{G}^{\frac{1}{2}}[k]\,. \quad (3.16)$$

By using the matrix inversion lemma (Woodbury matrix identity) [29] on the right-hand side of (3.16), we obtain the alternative expression

$$\left[\mathbf{G}^{-1}[k]+\frac{\mathbf{X}_{\text{dr}}[k]\mathbf{X}^{\text{T}}_{\text{dr}}[k]}{\delta_{\text{sm}}}\right]^{-1}=\Big(\mathbf{I}[k]-\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k](\delta_{\text{sm}}\mathbf{I}+\mathbf{X}^{\text{T}}_{\text{dr}}[k]\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k])^{-1}\mathbf{X}^{\text{T}}_{\text{dr}}[k]\Big)\mathbf{G}[k]\,, \quad (3.17)$$

which is used in (3.15), yielding

$$\begin{aligned}\hat{\mathbf{h}}[k+1]=\hat{\mathbf{h}}[k]&-\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]\Big(\delta_{\text{sm}}\mathbf{I}+\mathbf{X}^{\text{T}}_{\text{dr}}[k]\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]\Big)^{-1}\mathbf{X}^{\text{T}}_{\text{dr}}[k]\hat{\mathbf{h}}[k]+\\&\frac{\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]}{\delta_{\text{sm}}}\Big[\mathbf{I}-\Big(\delta_{\text{sm}}\mathbf{I}+\mathbf{X}^{\text{T}}_{\text{dr}}[k]\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]\Big)^{-1}\mathbf{X}^{\text{T}}_{\text{dr}}[k]\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]\Big]\Big(\mathbf{d}_{\text{dr}}[k]-\boldsymbol{\gamma}[k]\Big)\,.\end{aligned} \quad (3.18)$$

However, as the matrix identity $\Big[\mathbf{I}-\big(\delta_{\text{sm}}\mathbf{I}+\mathbf{Z}\big)^{-1}\mathbf{Z}\Big]=\delta_{\text{sm}}\mathbf{I}\big(\delta_{\text{sm}}\mathbf{I}+\mathbf{Z}\big)^{-1}$ holds

and $\mathbf{e}_{\mathrm{dr}}[k] = \mathbf{d}_{\mathrm{dr}}[k] - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k]$, from (3.18) we finally have that

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]\Big(\delta_{\mathrm{sm}}\mathbf{I} + \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]\Big)^{-1}\Big(\mathbf{e}_{\mathrm{dr}}[k] - \boldsymbol{\gamma}[k]\Big)\,. \quad (3.19)$$

Therefore, as (3.19) is equal to the update rule in (3.11), we have proved that the SM-PAPA practical update equation is related to the convex problem of minimizing the cost function (3.13) with respect to $\hat{\mathbf{h}}[k+1]$. By considering this result for the SM-PAPA, we suggest a systematic method for selecting an optimal constraint vector $\boldsymbol{\gamma}_{\mathrm{opt}}[k]$ at each iteration of the algorithm.

3.3.2 Optimal CV Solution

Considering the practical SM-PAPA cost-function minimized in (3.13), we notice the presence of the constraint vector $\boldsymbol{\gamma}[k]$, which has not been taken into account yet. In order to provide a non-heuristic method for defining $\boldsymbol{\gamma}[k]$, [53] suggests incorporating the CV as an additional variable that must be tuned so that it minimizes the convex cost-function. Based on the proposal for SM-AP algorithms [53], we extend this idea for handling the more general SM-PAPA and find that

$$\begin{aligned} &\underset{\hat{\mathbf{h}}[k+1],\boldsymbol{\gamma}_{\mathrm{opt}}[k]}{\text{minimize}} \quad \|\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\|^2_{\mathbf{G}^{-1}[k]} + \frac{1}{\delta_{\mathrm{sm}}}\|\mathbf{d}_{\mathrm{dr}}[k] - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k+1] - \boldsymbol{\gamma}_{\mathrm{opt}}[k]\|_2^2 \\ &\text{subject to} \quad \|\boldsymbol{\gamma}_{\mathrm{opt}}[k]\|_\infty \leq \bar{\gamma}\,. \end{aligned} \quad (3.20)$$

Thus, we define the optimal CV [53] as the solution $\boldsymbol{\gamma}_{\mathrm{opt}}[k]$ that minimizes (3.20).

According to Subsection 3.3.1, by first minimizing (3.20) with respect to $\hat{\mathbf{h}}[k+1]$ one obtains (3.19). Then, if we substitute this expression for $\hat{\mathbf{h}}[k+1]$ into (3.20), the term $\|\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\|^2_{\mathbf{G}^{-1}[k]}$ becomes

$$\begin{aligned} \|\mathbf{G}^{-\frac{1}{2}}[k]\cdot[\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]]\|_2^2 = [\mathbf{e}_{\mathrm{dr}}[k] - \boldsymbol{\gamma}_{\mathrm{opt}}[k]]^{\mathrm{T}}\Big(\delta_{\mathrm{sm}}\mathbf{I} + \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\Big)^{-1}\cdot \\ \cdot\,\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]\Big(\delta_{\mathrm{sm}}\mathbf{I} + \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\Big)^{-1}[\mathbf{e}_{\mathrm{dr}}[k] - \boldsymbol{\gamma}_{\mathrm{opt}}[k]]\,. \end{aligned} \quad (3.21)$$

Similarly, we verify that the second term in (3.20) is rewritten as

$$\begin{aligned} \|\mathbf{d}_{\mathrm{dr}}[k] - \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\hat{\mathbf{h}}[k+1] - \boldsymbol{\gamma}_{\mathrm{opt}}[k]\|_2^2 = \\ = \left\|\delta_{\mathrm{sm}}\Big[\delta_{\mathrm{sm}}\mathbf{I} + \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\Big]^{-1}\Big[\mathbf{e}_{\mathrm{dr}}[k] - \boldsymbol{\gamma}_{\mathrm{opt}}[k]\Big]\right\|_2^2. \end{aligned} \quad (3.22)$$

Therefore, by defining the positive definite matrix $\mathbf{S}_{\mathrm{sm}}[k] \in \mathbb{R}^{(L+1)\times(L+1)}$ as

$$\mathbf{S}_{\mathrm{sm}}[k] = \Big(\delta_{\mathrm{sm}}\mathbf{I} + \mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]\Big)^{-1}, \quad (3.23)$$

and summing expressions (3.21) and $[\delta_{\text{sm}}^{-1} \times (3.22)]$, one concludes that the original minimization task (3.20) becomes the strictly convex quadratic programming (QP) box-constrained problem

$$\begin{aligned} \underset{\boldsymbol{\gamma}_{\text{opt}}[k]}{\text{minimize}} \quad & C_{\text{opt}}(\boldsymbol{\gamma}_{\text{opt}}[k]) = [\boldsymbol{\gamma}_{\text{opt}}[k] - \mathbf{e}_{\text{dr}}[k]]^{\text{T}} \cdot \mathbf{S}_{\text{sm}}[k] \cdot [\boldsymbol{\gamma}_{\text{opt}}[k] - \mathbf{e}_{\text{dr}}[k]] \\ \text{subject to} \quad & \|\boldsymbol{\gamma}_{\text{opt}}[k]\|_{\infty} \leq \bar{\gamma}\,, \end{aligned} \tag{3.24}$$

which must be evaluated at each algorithm iteration k for finding the current optimal constraint-vector $\boldsymbol{\gamma}[k] = \boldsymbol{\gamma}_{\text{opt}}[k]$ to be be used along the SM-PAPA update equation (3.19).

3.3.3 Discussion about Computing the Optimal CV

According to [53], optimization problems such as (3.24) can be numerically solved using either efficient interior-points (IP) methods for precise results [43] or proximal-gradient algorithms for computational efficiency [60]. For instance, a simple way for computing the optimal CV $\boldsymbol{\gamma}_{\text{opt}}[k]$ in (3.24) is by calling a convex problem solver, such as the MATLAB-based toolbox CVX [61], whenever an update of $\hat{\mathbf{h}}[k+1]$ is required.

Although the usefulness of this simple procedure for evaluating the optimal $\boldsymbol{\gamma}_{\text{opt}}[k]$ *offline*, its use is restricted in *real-time* applications since each call of an IP-based solver like CVX might be very time-consuming. Thus, in order to evaluate the optimal CV in *real-time* applications, this work suggests an alternative way for solving quickly and with acceptable precision the bound-constrained QP problem in (3.24).

As QP problems are the simplest, yet the most frequently encountered class of constrained nonlinear optimization problems [62], there are several classes of algorithms for solving them. Considering the case with inequality constraints, the already mentioned IP method [43] is a good choice for large problems, but may underperform when compared to other algorithms that take specific characteristics of the current problem into account.

Particularly, the most effective method for solving a box-constrained QP problem is the gradient projection (GP) [63, 64]. As the GP method can quickly verify the constraint active set and perform subspace minimization, simplifying the search task, a GP implementation is expected to solve bound-constrained QP problems faster than IP methods, which never eliminate inequalities and work on the complete dimensional space. Besides faster convergence, the GP method also presents many good properties such as the simplicity and the warm start possibility [65]. Therefore, differently from the IP solver in [53], this dissertation proposes using a GP approach

for computing the minimization problem (3.24), which is further explained in the next subsection.

3.3.4 GP Method for Computing the Optimal CV

Gradient Projection (GP) methods [63, 64, 66] are simple procedures that might be seen as natural extensions of the *steepest-descent* idea for handling box-constrained problems. Consequently, the GP shares most of the advantages and disadvantages presented by algorithms based on the gradient of a function [66].

Basically, the idea behind using gradient projection methods for evaluating the optimal CV $\boldsymbol{\gamma}_{\text{opt}}$ in (3.24) consists in computing a new algorithm iterate $\boldsymbol{\gamma}^{+} \in \mathbb{R}^{L+1}$ from an initial point $\boldsymbol{\gamma}_0 \in \mathbb{R}^{L+1}$ based on two stages. First, the algorithm searches for the so-called Cauchy point $\boldsymbol{\gamma}^{\text{c}} \in \mathbb{R}^{L+1}$, which locally minimizes the quadratic cost-function $C_{\text{opt}}(\boldsymbol{\gamma}_{\text{opt}}[k])$ along a specific path. This path is formed by a sequence of lines, being initially based on the negative gradient direction $-\nabla_{\boldsymbol{\gamma}_{\text{opt}}} C_{\text{opt}}(\boldsymbol{\gamma}_0)$ (like the steepest-descent algorithm) but bending whenever a bound on $\boldsymbol{\gamma}_{\text{opt}}$ is found in order to keep the CV inside the feasible domain (in this work, $\|\boldsymbol{\gamma}_{\text{opt}}\|_{\infty} \leq \bar{\gamma}$).

The name gradient projection comes from the fact that the Cauchy point search can be summarized by computing

$$\boldsymbol{\gamma}^{\text{c}} = \mathcal{P}(\, \boldsymbol{\gamma}_0 - \alpha \cdot \nabla_{\boldsymbol{\gamma}_{\text{opt}}} C_{\text{opt}}(\boldsymbol{\gamma}_0)\,)\,, \tag{3.25}$$

where $\nabla_{\boldsymbol{\gamma}_{\text{opt}}} C_{\text{opt}}(\boldsymbol{\gamma}_0)$ represents the gradient of a quadratic function $C(\boldsymbol{\gamma}_{\text{opt}})$ with respect to $\boldsymbol{\gamma}_{\text{opt}}$ evaluated at point $\boldsymbol{\gamma}_0$, $\mathcal{P}(\cdot)$ is the projection function and α is the step-length parameter given by a line search scheme, such as the Armijo rule [63, 66]. For illustration purposes, in the particular case of the convex problem (3.24), if we consider a non-projected vector $\boldsymbol{\gamma}' = \boldsymbol{\gamma}_0 - \alpha \nabla_{\boldsymbol{\gamma}_{\text{opt}}} C_{\text{opt}}(\boldsymbol{\gamma}_0)$ with $(L+1)$ components $\boldsymbol{\gamma}' = [\gamma'_0 \quad \gamma'_1 \quad ... \quad \gamma'_L]^{\text{T}}$, the individual effect of the projection function $\mathcal{P}(\cdot)$ on its argument i^{th} component γ'_i is described by

$$\mathcal{P}(\gamma'_i) = \begin{cases} -\bar{\gamma} & , \text{ if } \gamma'_i < -\bar{\gamma}\,, \\ \bar{\gamma} & , \text{ if } \quad \gamma'_i > \bar{\gamma} \\ \gamma'_i & , \text{ otherwise } (|\gamma'_i| \leq \bar{\gamma})\,, \end{cases} \tag{3.26}$$

and its overall effect can be observed in Figure 3.3.

As illustrated in Figure 3.3, we notice that the minimum search for the Cauchy point is restricted to the surface of the box defined by the problem constraints after the first bend. Thus, one expects that, after the Cauchy point $\boldsymbol{\gamma}^{\text{c}}$ has been

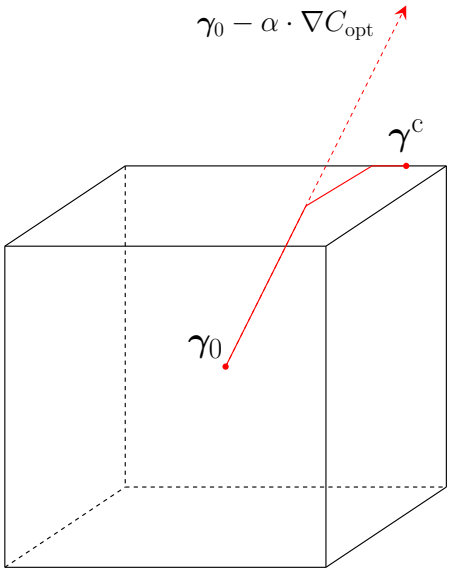


Figure 3.3: Example of a piecewise-linear path in $\mathbb{R}^3$ obtained for the Cauchy point $\boldsymbol{\gamma}^{\text{c}}$ search, the first stage of a gradient projection method.

computed, its components γ_i^{c} define the current active set $\mathcal{A}(\boldsymbol{\gamma}^{\text{c}})$ as

$$\mathcal{A}(\boldsymbol{\gamma}^{\text{c}}) = \{\, i \,|\, \gamma_i^{\text{c}} = \bar{\gamma} \text{ or } \gamma_i^{\text{c}} = -\bar{\gamma} \,\}\,. \tag{3.27}$$

Based on the active set $\mathcal{A}(\boldsymbol{\gamma}^{\text{c}})$, the second stage of the GP method explores the face of the feasible box region on which $\boldsymbol{\gamma}^{\text{c}}$ lies by solving a subproblem where the active components γ_i ($i \in \mathcal{A}(\boldsymbol{\gamma}^{\text{c}})$) are fixed at γ_i^{c} [64], which are either $-\bar{\gamma}$ or $\bar{\gamma}$ in the optimal CV particular case. Thus, the second stage reduces the original subspace dimension in (3.24) from $(L+1)$ to $\big[(L+1) - |\mathcal{A}(\boldsymbol{\gamma}^{\text{c}})|\big]$, resulting in the simpler problem

$$\begin{aligned} \underset{\boldsymbol{\gamma}^{+}}{\text{minimize}} \quad & C_{\text{opt}}(\boldsymbol{\gamma}^{+}) = [\boldsymbol{\gamma}^{+} - \mathbf{e}_{\text{dr}}[k]]^{\text{T}} \cdot \mathbf{S}[k] \cdot [\boldsymbol{\gamma}^{+} - \mathbf{e}_{\text{dr}}[k]]\,, \\ \text{subject to} \quad & \gamma_i^{+} = \gamma_i^{\text{c}}\,,\ i \in \mathcal{A}(\boldsymbol{\gamma}^{\text{c}})\,, \\ & |\gamma_i^{+}| \leq \bar{\gamma}\,,\ i \notin \mathcal{A}(\boldsymbol{\gamma}^{\text{c}})\,, \end{aligned} \tag{3.28}$$

which provides the current iteration update vector $\boldsymbol{\gamma}^{+}$. Although the reduced convex problem (3.28) presents a single solution, it does not have to be solved exactly and this is not even desirable since the subproblem (3.28) may be almost as difficult as the original minimization task in (3.24) [64]. As a matter of fact, the only requirement for the GP procedure global convergence is that the cost-function in approximate solution $\boldsymbol{\gamma}^{+}$ obtained from (3.28) is no worse than that for $\boldsymbol{\gamma}^{\text{c}}$, i.e., $C_{\text{opt}}(\boldsymbol{\gamma}^{+}) \leq C_{\text{opt}}(\boldsymbol{\gamma}^{\text{c}})$. Although one can simply choose $\boldsymbol{\gamma}^{+} = \boldsymbol{\gamma}^{\text{c}}$, some works suggest implementing an intermediate selection between this approach and the proper solution of (3.28), such as conjugate gradient iterations mentioned in [64].

Thus, by computing the vector $\boldsymbol{\gamma}^+$ one reaches the end of the GP algorithm internal iteration. These iterations are performed until the Karush–Kuhn–Tucker (KKT) conditions are satisfied [43]. For further reference about particular implementations of GP methods, a detailed description is presented in [63] and [64].

3.3.5 Numerical Simulations

This subsection compares the performance of three SM-PAPA with different CV selection rules: the SC-CV from (3.12), the optimal CV computed using interior-point (IP) methods, as suggested in [53], and the gradient projection (GP) method proposed in Subsection 3.3.4. For the IP approach we employ the widely known CVX convex optimization toolbox [61], whereas for the GP implementation we use the MATLAB script developed by C. T. Kelley in [67], which is a practical coded version of the algorithm presented in [66].

We use the SM-PAPA for a system identification setup like Figure 2.2, where the system impulse response is described by a sparse vector $\mathbf{h} \in \mathbb{R}^{16}$. Additionally, in order to verify how fast the algorithms can readapt, a proportional increase in the non-zero system coefficients $h_m \in \mathbf{h}$ ($m \in \{0, 1, ..., 15\}$) occurs at $k = 1000$. Thus, the system impulse responses are given by

$$\begin{cases} \mathbf{h}_1 = [1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0]^{\mathrm{T}}, \; 0 \leq k < 1000\,, \text{ and} \\ \mathbf{h}_2 = [2, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 2, 2, 2, 0]^{\mathrm{T}}, \; 1000 \leq k \leq 4000\,. \end{cases}$$

Based on [53], we expect that the optimal CV brings about more noticeable improvements in setups with statiscally colored inputs when compared to the SC-CV. Thus, this simulation focuses only on correlated input signals, as the ones generated by auto-regressive (AR) processes.

Considering that $w_x[k]$ is a white Gaussian real-valued noise with unitary variance $\sigma_{w_x}^2 = 1$, the input signal $x[k]$ is drawn from the following AR processes:

- AR1: $x[k] = 0.95\, x[k-1] + w_x[k]$;
- AR4: $x[k] = 0.95\, x[k-1] + 0.19\, x[k-2] + 0.09\, x[k-3] - 0.5\, x[k-4] + w_x[k]$.

Furthermore, taking a measurement noise with variance $\sigma_w^2 = 10^{-3}$ at the system output, we follow the practical suggestion from [29] and choose an update threshold of $\bar{\gamma} = \sqrt{5}\,\sigma_w \approx 0.071$ for the SM-PAPA. Additionally, we use a data reuse factor $L = 3$, initialize the coefficients vector as $\hat{\mathbf{h}}[0] = \mathbf{0}$ ($\hat{\mathbf{h}}[k] \in \mathbb{R}^{16}$), and adopt the SM-PAPA internal parameters $\delta_{\mathrm{sm}} = 5 \cdot 10^{-7}$ as the regularization factor and $\tau = 0.9$ for updating the proportionate matrix $\mathbf{G}[k]$ in (2.32) according to the rule in [52]

when $e[k] > \bar{\gamma}$, which is conveniently written as

$$\begin{aligned} \nu_{\text{sm}}[k] &= 1 - \frac{\bar{\gamma}}{|e[k]|}\,, \\ g_i[k] &= \frac{1 - \tau\,\nu_{\text{sm}}[k]}{16} + \frac{\tau\,\nu_{\text{sm}}[k]\,|\hat{h}_i[k]|}{\sum_{i=0}^{15}\,|\hat{h}_i[k]|}\,, \quad i \in \{0, 1, ..., 15\}\,, \end{aligned} \tag{3.29}$$

in the current application. If $\hat{\mathbf{h}}[k] = \mathbf{0}$, as in the first algorithm iteration, it is considered that the second term in the right-hand side of $g_i[k]$ in (3.29) assumes a null value.

For comparing the performances induced by the three CVs we choose the following FoMs: cost-function value $C(\boldsymbol{\gamma}[k])$ in (3.24), MSE in (2.4), percentage of updates, and the average iteration time interval in steady-state (last 500 iterations). This last FoM is based on MATLAB-functions *tic* and *toc* and aims to measure roughly the time interval taken for calculating the new CV $\boldsymbol{\gamma}(k)$ for each rule. Thus, although the absolute value of this FoM depends on the simulation environment used, its relative result gives a good idea about the speed of each implementation.

For simulation purposes, each algorithm has been performed for 4000 iterations in every run and, for evaluating the estimated FoMs, the average of 250 runs has been taken. The numerical results verified for the simulations involving the three CV selection rules are presented in Figures 3.4 and 3.5, where the cost-function and MSE time-varying behaviors are illustrated, and Table 3.1, that summarizes the FoMs average iteration time interval and percentage of updates. In particular, the optimal CV based on an IP implementation is not explicitly illustrated in Figures 3.4 and 3.5 because it is virtually identical to the signals obtained using the GP optimal CV.

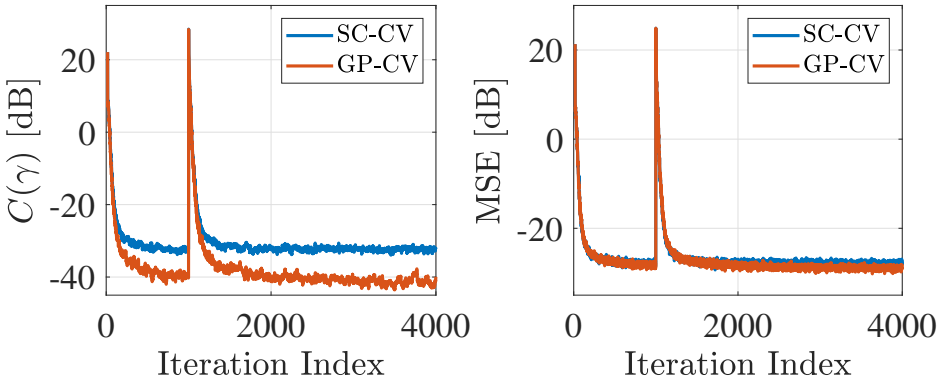


Figure 3.4: Cost-function $C(\boldsymbol{\gamma}[k])$ in (3.24) and MSE for the AR1 input scenario.

Based on the information presented, one can notice that using the optimal CV approach in systems with correlated input signals results in lower steady-state values for the cost-function $C(\boldsymbol{\gamma}[k])$ and MSE, besides the reduced number of coefficient updates, in comparison to the SC-CV. However, due to its simplicity, the SC-CV

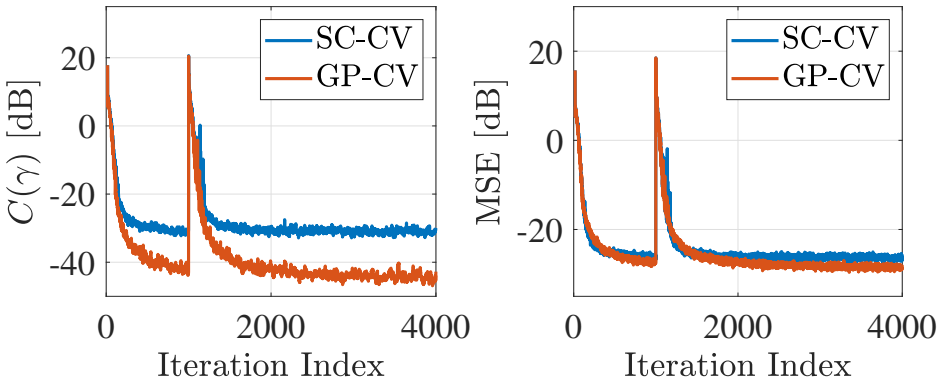


Figure 3.5: Cost-function $C(\boldsymbol{\gamma}[k])$ in (3.24) and MSE for the AR4 input scenario.

Table 3.1: Average iteration time interval and update percentage of different CV selection rules for the SM-PAPA

	AR1 input		AR4 input	
CV Rule	Average Iteration Time Interval [s]	Updates [%]	Average Iteration Time Interval [s]	Updates [%]
Simple-Choice	$2.8 \cdot 10^{-7}$	12.9%	$5.6 \cdot 10^{-7}$	20.7%
IP Optimal	$1.8 \cdot 10^{-2}$	11.3%	$2.4 \cdot 10^{-2}$	17.0%
GP Optimal	$2.0 \cdot 10^{-4}$	11.3%	$5.3 \cdot 10^{-4}$	17.0%

is much faster than the optimal CV implementations when it comes to the average time interval taken at every iteration for calculating $\boldsymbol{\gamma}[k]$.

Moreover, an important conclusion drawn from the comparison between optimal CV selection rules in Table 3.1, and one of the contributions of this work, is that the use of a GP method for obtaining the optimal CV results in faster implementations than those based on IP methods. Although the average time interval results presented in Table 3.1 focus on the steady-state performance, by plotting the computing time interval in both transient and steady states, one can verify a similar behavior in which the GP-based implementation is much faster than the IP-based one, being more than 45 times faster in the steady-state region for both input signals tested.

3.3.6 Summary of Contributions

The optimal CV for SM-PAPA presented in this section extended an exact formulation from [53] to the family of set-membership proportionate affine projection algorithms. The theoretical generalization is corroborated by a numerical performance comparison with the simple-choice CV, which indicates that the optimal CV

approach presents advantages for colored input signals in sparse environments.

In addition, the gradient projection method for obtaining the optimal CV produces similar overall results when compared to a standard interior-point implementation. The great advantage of the suggested GP approach is that it is considerably faster than the IP-based method, enabling its use in real-time applications.

Chapter 4

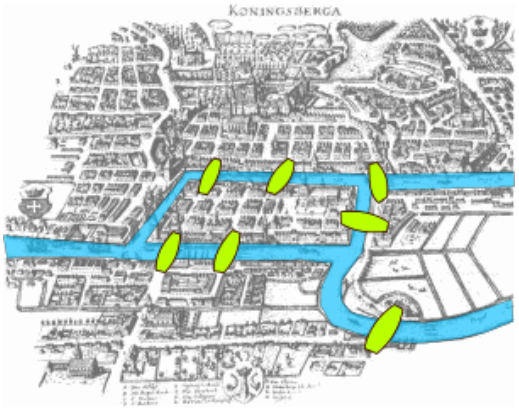
Graph Signal Processing

In this chapter we first present the basic ideas concerning the graph signal processing (GSP) framework, where we start by describing the mathematical graph structure and defining the graph signal concept, which allows us to discuss the two alternative approaches for implementing GSP: based on the algebraic signal processing (ASP) and on the graph spectral theory [6]. Then, we analyze the extension of classical signal processing ideas, such as Fourier transform and compact representation of information, for defining bandlimited graph signals and obtaining proper methods for sampling and reconstructing these signals. Finally, the last part of this chapter deals with the adaptive approach for online estimating bandlimited graph signals, presenting the recently suggested connection between the adaptive filtering and the GSP research areas, where LMS and RLS-based algorithms are employed for solving an original GSP problem.

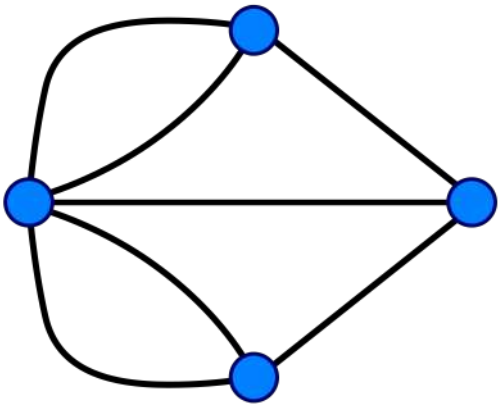
4.1 Basic Concepts about Graphs

Intended for modeling relations among objects, a graph is a generic mathematical structure made up of two simple elements: vertices (or nodes) and edges (or links). The use of these elements arise naturally in some practical applications, as the ones involving spatial distances between different regions, and allow useful abstractions for solving specific problems. For example, let us consider the classical problem of the seven bridges of Konigsberg (now Kaliningrad, Russia), whose solution by Leonard Euler led to the development of the mathematical branch known as graph theory [5]. Since the city of Konigsberg was formed by four landmasses connected by seven bridges in Euler's time, as can be observed in Figure 4.1a, a practical question arises when one asks if it is possible for a citizen to walk through the town so that each bridge would be crossed exactly once.

For solving this problem, Euler devised a convenient representation where the landmasses become vertices and the bridges are seen as edges connecting these nodes,



(a) Map showing the city bridges.



(b) Respective graph representation.

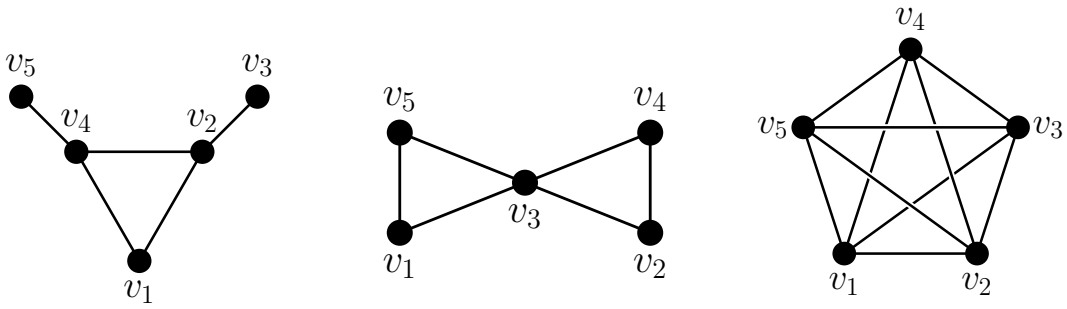
Figure 4.1: City of Konigsberg in Euler's time. Taken from [5].

as shown in Figure 4.1b. Based on this graph representation, Euler proved that it was impossible to cross each bridge of Konigsberg once and only once because the graph in Figure 4.1b presented nodes of odd degree [5], i.e., with an odd number of edges connected to them. The importance of this result is that its generalization can be applied to similar problems, where the problem essence is extracted from its particularities, in the same way as basic arithmetic operations do not require a precise knowledge about the objects they operate on. The main results and properties of graph structures gave rise to the research field of graph theory [5].

4.1.1 Graph Structure

In general terms, a graph structure $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N nodes or vertices is formally defined by the vertices set $\mathcal{V} = \{v_1, v_2, ..., v_N\}$ and edges set $\mathcal{E}$. Although a general graph might present any number of edges connecting two vertices v_n, $v_m \in \mathcal{V}$, as happens for the Konigsberg graph in Figure 4.1b, in this work we assume that the connection between two nodes has at most one edge (multiple edge graphs are not considered). Then, the N-node structure $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ has at most $N(N-1)/2$ edges, which are uniquely characterized by the notation $\widehat{v_n v_m}$ if the relation between vertices v_n and v_m exists. This observation allows us to uniquely represent the edges set as $\mathcal{E} = \{\widehat{v_n v_m}\}$. For didactic purpose, we present some simple graph structures with 5 vertices and different edge set cardinality ($|\mathcal{E}|$) in Figure 4.2. These three graphs $\mathcal{G}_a = (\mathcal{V}_a, \mathcal{E}_a)$, $\mathcal{G}_b = (\mathcal{V}_b, \mathcal{E}_b)$ and $\mathcal{G}_c = (\mathcal{V}_c, \mathcal{E}_c)$ have the same vertex set $\mathcal{V}_a = \mathcal{V}_b = \mathcal{V}_c = \{v_1, v_2, v_3, v_4, v_5\}$, but different edge sets respectively given by

$$\begin{aligned}
\mathcal{E}_a &= \{\widehat{v_1v_2}, \widehat{v_1v_4}, \widehat{v_2v_3}, \widehat{v_2v_4}, \widehat{v_4v_5}\}\,, \\
\mathcal{E}_b &= \{\widehat{v_1v_3}, \widehat{v_1v_5}, \widehat{v_2v_3}, \widehat{v_2v_4}, \widehat{v_3v_4}, \widehat{v_3v_5}\}\,, \\
\mathcal{E}_c &= \{\widehat{v_1v_2}, \widehat{v_1v_3}, \widehat{v_1v_4}, \widehat{v_1v_5}, \widehat{v_2v_3}, \widehat{v_2v_4}, \widehat{v_2v_5}, \widehat{v_3v_4}, \widehat{v_3v_5}, \widehat{v_4v_5}\}\,.
\end{aligned} \tag{4.1}$$



(a) $\mathcal{G}_a(\mathcal{V}_a, \mathcal{E}_a)$ with $|\mathcal{E}_a| = 5$. (b) $\mathcal{G}_b(\mathcal{V}_b, \mathcal{E}_b)$ with $|\mathcal{E}_b| = 6$. (c) $\mathcal{G}_c(\mathcal{V}_c, \mathcal{E}_c)$ with $|\mathcal{E}_c| = 10$.

Figure 4.2: Simple graph structures with 5 nodes and different numbers of edges.

Since a graph structure intends to properly represent the relation between objects, and the intensity of these connections are not necessarily the same throughout the whole graph, it is useful to include an auxiliary parameter for indicating stronger or weaker node links in a graph. For describing the connection strength between nodes v_n and $v_m \in \mathcal{V}$, we consider that each edge $\widehat{v_n v_m} \in \mathcal{E}$ is associated with a non-null weight $a_{nm} \in \mathbb{R}_*$, which can be understood as a proximity or similarity metric. Based on these values, by taking the weight a_{nm} as an element of the n^{th} row and m^{th} column, one can obtain an $N \times N$ matrix called the adjacency matrix $\mathbf{A}$, which is useful since it stores information about the graph connections in a compact representation, being widely used in some GSP approaches [4, 7, 8]. Additionally, an alternative graph representation is given by the Laplacian matrix $\mathbf{L} \in \mathbb{R}^{N \times N}$ defined as $\mathbf{L} = \mathbf{K} - \mathbf{A}$, where $\mathbf{K}$ is a diagonal matrix with diagonal entries $k_n = \sum_{m=1}^{N} a_{nm}$ [4]. By considering the bull graph $\mathcal{G}_a(\mathcal{V}_a, \mathcal{E}_a)$ in Figure 4.2a, the simplest way of assigning edge weights is to consider that $a_{nm} = 1$ if $\widehat{v_n v_m} \in \mathcal{E}_a$ and $a_{nm} = 0$ if $\widehat{v_n v_m} \notin \mathcal{E}_a$, which results in the respective adjacency and Laplacian matrices given by

$$\mathbf{A}'_a = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad \mathbf{L}'_a = \begin{bmatrix} 2 & -1 & 0 & -1 & 0 \\ -1 & 3 & -1 & -1 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ -1 & -1 & 0 & 3 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}. \tag{4.2}$$

On the other hand, if we wish to describe the connections among some nodes as stronger than others, we employ the edge weights as $\{a_{12}, a_{14}, a_{23}, a_{24}, a_{45}\} =$

$\{2, 2, 1, 2, 1\}$ and obtain the adjacency and Laplacian matrices

$$\mathbf{A}''_a = \begin{bmatrix} 0 & 2 & 0 & 2 & 0 \\ 2 & 0 & 1 & 2 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \; \mathbf{L}''_a = \begin{bmatrix} 4 & -2 & 0 & -2 & 0 \\ -2 & 5 & -1 & -2 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ -2 & -2 & 0 & 5 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}. \tag{4.3}$$

In particular, from (4.2) and (4.3) we observe that both the adjacency and Laplacian matrices are symmetric, i.e., they are equal to their transpose. Although this matrix property is not true for directed graphs (*digraphs*), in which $a_{nm} \neq a_{mn}$ for some $n, m \in \mathcal{N} = \{1, 2, \ldots, N\}$, as this work only considers undirected graph structures, where $a_{nm} = a_{mn}$, it follows that both $\mathbf{A}$ and $\mathbf{L}$ are always symmetric in our scope.

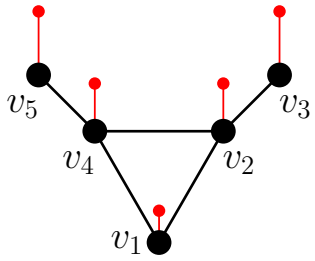
4.1.2 Graph Signal

So far we have discussed about the graph structure, a topic which is extensively studied in the research field of graph theory [5] since many useful graph properties can be obtained by a proper structural analysis. However, in certain practical problems the simple abstraction of objects and their relations as vertices and weighted edges, respectively, is not enough for a complete description of the problem. For example, let us consider Figure 4.3a, where we have data about the location of 5 cities, designated as the nodes $\{v_1, v_2, v_3, v_4, v_5\}$, and their respective temperature values at a specific time instant. If we consider an explicit connection between cities when their Euclidean distance $d_{\mathrm{E}}(v_i, v_j)$ (equal to $\sqrt{(\Delta x)^2 + (\Delta y)^2}$) is smaller than 3, it is trivial to conclude that the bull graph in Figure 4.2a can be used for the abstraction of the current scenario. Even though the city distances might be more accurately reported by assigning different values of edge weights for different distances, the temperature values are still not properly represented by the use of the graph structure only.

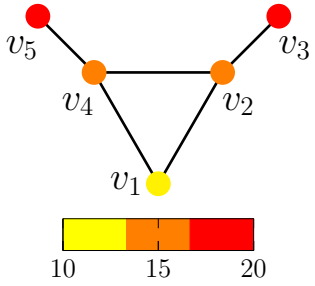
Thus, based on our simple example of temperature values across neighbor cities, it is natural to come up with the idea of a graph signal, in which scalar values are assigned to the vertices of a graph structure. In formal terms, by considering a generic graph structure $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N vertices, one defines on the nodes of $\mathcal{G}$ the graph signal (GS) $x : \mathcal{V} \to \mathbb{R}$, that can be compactly represented as a vector $\mathbf{x}_{\mathrm{G}} \in \mathbb{R}^N$, whose n^{th} entry $x_{\mathrm{G_n}}$ contains the function value at vertex $v_n \in \mathcal{V}$ [6]. For instance, the problem data presented in Figure 4.3a corresponds to the graph signal

City	Coordinates (x, y)	Temp. (° C)
v_1	$(0, 0)$	10
v_2	$(1, \sqrt{3})$	15
v_3	$(2, 1+\sqrt{3})$	20
v_4	$(-1, \sqrt{3})$	15
v_5	$(-2, 1+\sqrt{3})$	20

(a) Summary of the problem data.



(b) GS representation 1.



(c) GS representation 2.

Figure 4.3: Graph signal concept illustrated on an application scenario with the spatial location of 5 nearby cities and their current temperature measurements.

$\mathbf{x}'_{\mathrm{G}} \in \mathbb{R}^5$ given by

$$\mathbf{x}'_{\mathrm{G}} = \begin{bmatrix} 10 & 15 & 20 & 15 & 20 \end{bmatrix}^{\mathrm{T}}, \tag{4.4}$$

which is also graphically represented as in Figure 4.3b or in Figure 4.3c. In particular, the graphic representation in Figure 4.3c is more compact than the one in Figure 4.3b, thus, it will be used in this work as the default description for illustrating the GS over the graph structure.

4.1.3 Graph Structure Inference

Although some practical applications exhibit an explicit graph structure representation, such as the Konigsberg bridge problem summarized in Figure 4.1, there are cases in which one has a signal dataset obtained from individual objects but does not know how these objects relate to each other, i.e., the graph structure remains implicit. A clear example of this scenario has been given in the previous subsection, where we did not have the explicit graph structure for precisely describing the relations among cities but, since we assumed that nearby regions behave in a more similar way than far-away regions, we follow a set of rules and try to estimate the actual structure. Basically, in this case we evaluated the Euclidean distance between cities and assumed the existence of a graph edge when this distance was smaller than a threshold $T \in \mathbb{R}$.

Besides this trivial procedure for estimating the graph structure, where we just indicate the existence of a simple edge between nodes, the inference process can be improved if we assign different weights to each possible edge, in which these weights are somewhat proportional to a proximity or similarity metric between vertices. For example, by considering an N-node sensor network application that each sensor is taken as a graph vertex v_n ($n \in \mathcal{N}$), as we expect that closer sensors provide a more similar measurement, we may adopt the Euclidean distance $d_{\mathrm{E}}(v_i, v_j)$ among

vertices v_i and v_j to evaluate their relatedness. However, as one wishes to obtain higher weights for closer nodes, a suitable metric mapping for evaluating the edge weight a_{ij} is to use the Gaussian kernel weighting function [6]

$$a_{ij} = \exp\left(\frac{-d_{\mathrm{E}}^2(v_i, v_j)}{2\theta^2}\right), \tag{4.5}$$

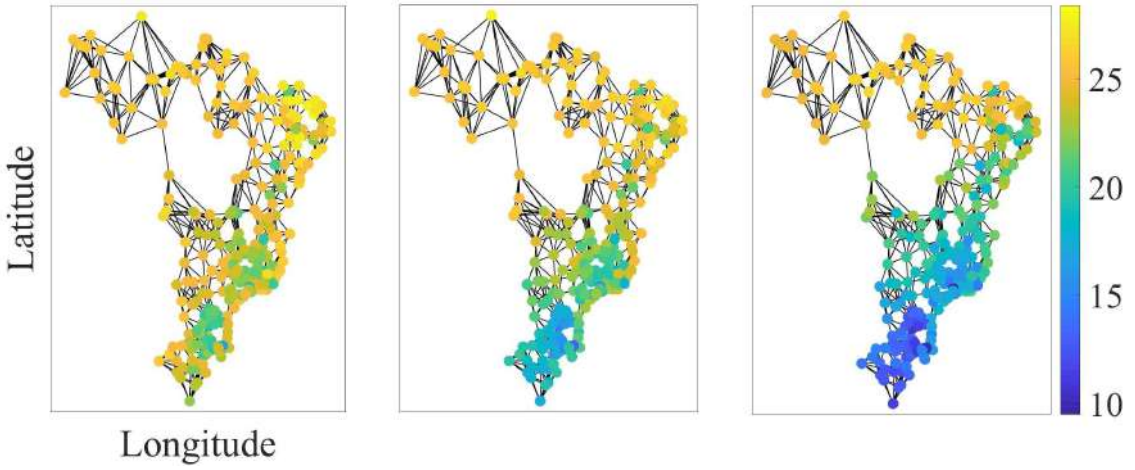
where θ is an internal variance parameter that depends on the application. Moreover, in a similar way to the simpler inference method, we only evaluate (4.5) if the Euclidean distance $d_{\mathrm{E}}(v_i, v_j)$ is smaller than the threshold T.

Although this inference method based on a threshold metric seems reasonable for general applications, it presents some disadvantages in more irregular scenarios, where one has a graph with both dense and sparsely populated regions. In these cases, the threshold selection leads to a trade-off between keeping the graph connected and finding a sparse adjacency matrix since, for covering a large distance between nodes, we choose a large threshold T that implies in increasing the number of edges [68]. In order to overcome this issue and obtain a graph structure that maintains a regular number of edges among vertices, an alternative method is to connect each node to its K closest neighbors and assign the edge weights according to a given similarity metric [6], such as the mapping expression in (4.5). A practical example of this scenario, where implementing the K-closest-neighbor strategy presents some advantages, occurs when one deals with certain datasets collected from sensors spread across the Brazilian territory, such as the temperature measurements from weather stations [41] later presented in Section 6.1. As we observe from the average temperature measurements taken in three distinct months of the year in Figure 4.4, the spatial distribution of weather stations is irregular, which justifies the use of the 8 closest neighbors for estimating the graph structure.

4.2 GSP Frameworks

Since we start modeling the information represented on a graph structure as a signal defined over its nodes, as suggested in Subsection 4.1.2, we expect to extend classical signal processing ideas in order to further explore this graph signal representation. However, at this point it is necessary to distinguish between two distinct GSP frameworks that grew independently throughout the last years and have been established as default mindsets when dealing with graph signals [68].

The first approach [7, 8] is based on the algebraic signal processing (ASP) theory [69, 70], which uses the graph adjacency matrix $\mathbf{A}$ as elementary block by associating it to the *graph shift* operation, from where most of the further developments, covering



(a) January (b) April (c) July

Figure 4.4: Inferred graph structure and graph signals representation of 1961-1990 monthly average temperatures (in °C) from Brazilian weather stations [41]. This graph structure inference is based on the K closest neighbors strategy.

aspects such as graph filtering, arise. On the other hand, the second framework [6] relies on graph spectral theory, a research branch of graph theory concerned with the eigendecomposition of graph characteristic matrices, and uses the graph Laplacian matrix $\mathbf{L}$ for defining the basis of its signal space [68]. Due to their relation with the adjacency and Laplacian matrices described in Subsection 4.1.1, these alternative approaches are conveniently called as GSP_A and GSP_L, respectively, in [68], where further particularities between these two frameworks are discussed. Additionally, another work that also pinpoints these GSP approach differences is [4]. In terms of practical applications, the GSP_L imposes the graph structure to be an undirected graph with non-negative real edge weights, while the GSP_A framework does not require any restrictions on the graph structure or edge weight values [3].

4.3 GSP Toolset

Although the GSP frameworks developed so far extend a wide range of traditional signal processing ideas [1], such as graph filtering, wavelet decomposition, denoising and filter banks on graphs [4], in this dissertation we focus on a particular branch of the GSP analysis, based on the useful properties that some graph signals present in their frequency domain. Thus, in this section we introduce the idea of performing a Fourier transform on a graph signal, which results in a sparse representation for smooth graph signals, that can be sampled and perfectly reconstructed if some conditions are satisfied. The recovery of sampled bandlimited graph signals from noisy measurements is the inspiration for a first attempt on merging the well-establish adaptive filtering research with the emerging area of GSP, culminating in the im-

plementation of LMS and RLS-based algorithms for graph signal estimation [19] further explained in Section 4.4.

4.3.1 Graph Fourier Transform

Motivated by the classical signal processing tool known as the Fourier transform [1], which expands an original time-domain signal into a Fourier basis of signals that are linear-filtering invariant, one can define the *graph Fourier transform* (GFT) of a GS $\mathbf{x}_{\mathrm{G}} \in \mathbb{R}^N$ as its projection onto a set of orthonormal vectors $\{\mathbf{u}_n\} \subset \mathbb{R}^N$, where $n \in \mathcal{N}$. Those basis vectors are usually chosen as the orthonormal eigenvectors of either the adjacency matrix $\mathbf{A}$ [7, 8] or the Laplacian matrix $\mathbf{L}$ [6], depending on the GSP approach considered ($\mathrm{GSP_A}$ or $\mathrm{GSP_L}$, respectively), so that the information inherent to the graph structure is naturally embedded in the resulting frequency-domain representation. Although the choice for the more general approach $\mathrm{GSP_A}$ may lead to a particular case where matrix $\mathbf{A}$ is not diagonalizable and the GSP definition requires the use of a Jordan form [7], as we only consider undirected graphs in this work, both matrices $\mathbf{L}$ and $\mathbf{A}$ are real symmetric and their spectral decompositions assume the form $\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{T}}$, where $\mathbf{\Lambda} \in \mathbb{R}^{N\times N}$ is the diagonal eigenvalues matrix and $\mathbf{U} \in \mathbb{R}^{N\times N}$ is the orthonormal eigenvectors matrix formed by the eigenvectors $\{\mathbf{u}_n\}$ as columns of $\mathbf{U}$. Thus, the GFT of a graph signal $\mathbf{x}_{\mathrm{G}}$ is given by [6, 7]

$$\mathbf{s} = \mathbf{U}^{\mathrm{T}}\mathbf{x}_{\mathrm{G}}\,, \tag{4.6}$$

and, for recovering $\mathbf{x}_{\mathrm{G}}$ from its frequency-domain representation $\mathbf{s}$, one defines the *inverse graph Fourier transform* (IGFT) as

$$\mathbf{x}_{\mathrm{G}} = \mathbf{U}\mathbf{s}\,. \tag{4.7}$$

For illustration purpose, let us consider the simple 5-node graph structure displayed in Figure 4.3, which we choose to be represented either by the adjacency matrix $\mathbf{A}''_a$ or Laplacian matrix $\mathbf{L}''_a$ in (4.3), and the temperature graph signal $\mathbf{x}'_{\mathrm{G}}$ in (4.4). Based on the adjacency matrix $\mathbf{A}''_a$, we compute the orthonormal eigenvectors matrix $\mathbf{U}''_{\mathrm{A}}$ and obtain the respective frequency-domain representation $\mathbf{s}''_{\mathrm{A}}$ according

to the GFT expression (4.6) as

$$\mathbf{s}''_{\mathrm{A}} = \begin{bmatrix} 0.0000 & 0.7648 & 0.3333 & 0.0000 & 0.5513 \\ 0.6533 & -0.4135 & -0.0000 & 0.2706 & 0.5736 \\ -0.2706 & 0.1912 & -0.6667 & 0.6533 & 0.1378 \\ -0.6533 & -0.4135 & 0.0000 & -0.2706 & 0.5736 \\ 0.2706 & 0.1912 & -0.6667 & -0.6533 & 0.1378 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} 10 \\ 15 \\ 20 \\ 15 \\ 20 \end{bmatrix} = \begin{bmatrix} 0.0000 \\ 2.8934 \\ -23.3333 \\ 0.0000 \\ 28.2344 \end{bmatrix}. \tag{4.8}$$

In a similar way, if we use the Laplacian $\mathbf{L}''_a$ in (4.3), we first evaluate $\mathbf{U}''_{\mathrm{L}}$ and, using the GFT in (4.6), find its respective frequency-domain signal $\mathbf{s}''_{\mathrm{L}}$ as

$$\mathbf{s}''_{\mathrm{L}} = \begin{bmatrix} -0.4472 & 0.0000 & -0.4865 & 0.7505 & 0.0000 \\ -0.4472 & 0.1133 & -0.2979 & -0.4596 & 0.6980 \\ -0.4472 & 0.6980 & 0.5412 & 0.0843 & -0.1133 \\ -0.4472 & -0.1133 & -0.2979 & -0.4596 & -0.6980 \\ -0.4472 & -0.6980 & 0.5412 & 0.0843 & 0.1133 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} 10 \\ 15 \\ 20 \\ 15 \\ 20 \end{bmatrix} = \begin{bmatrix} -35.7771 \\ 0.0000 \\ 7.8445 \\ -2.9093 \\ 0.0000 \end{bmatrix}. \tag{4.9}$$

From both results in (4.8) and (4.9) we observe that the graph signal $\mathbf{x}'_{\mathrm{G}}$ in (4.4) presents an interesting property in its frequency domain since it can be represented by a reduced number of non-null values. This useful sparsity property in an alternative domain can be exploited in order to provide a more compact representation of the original signal.

4.3.2 Sampling and Reconstruction of Graph Signals

Similarly to the definition used for time signals, we can extend the idea of bandlimited signals to graph structures and say that a graph signal $\mathbf{x}_{\mathrm{o}} \in \mathbb{R}^N$ is *bandlimited* or *spectrally sparse (ssparse)* when its frequency representation $\mathbf{s}$ given by the GFT in (4.6) is sparse [12], as in (4.8) and (4.9). Taking $\mathcal{F}$ as an index subset of $\mathcal{N}$ ($\mathcal{F} \subseteq \mathcal{N}$), a graph signal $\mathbf{x}_{\mathrm{o}}$ is defined as $\mathcal{F}$-ssparse if $\mathbf{s}$ is such that $\mathbf{s}_{\mathcal{N}\setminus\mathcal{F}}$ is a zero vector [12], i.e., the components of $\mathbf{s}$ with indices in $\mathcal{N}\setminus\mathcal{F}$ are equal to zero, where $\mathcal{N}\setminus\mathcal{F}$ denotes the difference between sets $\mathcal{N}$ and $\mathcal{F}$. In other words, this *support* or *frequency set* of $\mathcal{F}$ can be described as $\mathcal{F} = \{f \in \mathcal{N} \,|\, s_f \neq 0\}$ [15]. For example, based on the frequency-domain signals $\mathbf{s}''_{\mathrm{A}}$ and $\mathbf{s}''_{\mathrm{L}}$ from (4.8) and (4.9) we have the support sets $\mathcal{F}_{\mathrm{A}} = \{2, 3, 5\}$ and $\mathcal{F}_{\mathrm{L}} = \{1, 3, 4\}$, respectively. Then, we conclude that the graph signal $\mathbf{x}'$ is bandlimited with respect to both $\mathrm{GSP}_{\mathrm{A}}$ or $\mathrm{GSP}_{\mathrm{L}}$ approaches.

Moreover, if we consider that $\mathbf{U}_{\mathcal{F}} \in \mathbb{R}^{N\times|\mathcal{F}|}$ and $\mathbf{s}_{\mathcal{F}} \in \mathbb{R}^{|\mathcal{F}|}$ are, respectively,

the matrix formed by eigenvectors and the frequency representation indexed by the elements in $\mathcal{F}$, from (4.7) we can write that

$$\mathbf{x}_{\mathrm{o}} = \mathbf{U}_{\mathcal{F}}\mathbf{s}_{\mathcal{F}}\,. \tag{4.10}$$

So, by only using the columns of $\mathbf{U}''_{\mathrm{A}}$ and positions of $\mathbf{s}''_{\mathrm{A}}$ that are indexed by the support set $\mathcal{F}_{\mathrm{A}} = \{2, 3, 5\}$ we rewrite the graph signal $\mathbf{x}'_{\mathrm{G}}$ as

$$\underbrace{\begin{bmatrix} 10 \\ 15 \\ 20 \\ 15 \\ 20 \end{bmatrix}}_{\mathbf{x}'_{\mathrm{G}}} = \underbrace{\begin{bmatrix} 0.7648 & 0.3333 & 0.5513 \\ -0.4135 & -0.0000 & 0.5736 \\ 0.1912 & -0.6667 & 0.1378 \\ -0.4135 & 0.0000 & 0.5736 \\ 0.1912 & -0.6667 & 0.1378 \end{bmatrix}}_{\mathbf{U}''_{\mathrm{A}_{\mathcal{F}_{\mathrm{A}}}}} \underbrace{\begin{bmatrix} 2.8934 \\ -23.3333 \\ 28.2344 \end{bmatrix}}_{\mathbf{s}''_{\mathrm{A}_{\mathcal{F}_{\mathrm{A}}}}}. \tag{4.11}$$

As bandlimited time signals can be sampled and reconstructed with no loss of information as long as the Nyquist criterion is satisfied, we describe now a similar result for graph signals. In terms of sampling and reconstruction of bandlimited graph signals, some works [11, 20] describe these operations over a GS $\mathbf{x}_{\mathrm{o}}$ as the result of pre-multiplying it by a sampling matrix $\mathbf{D}_{\mathcal{S}} \in \mathbb{R}^{N\times N}$ and an interpolation matrix $\mathbf{\Phi} \in \mathbb{R}^{N\times N}$. Sampling is the operation of collecting only a limited number of values from the GS, whose reading positions are determined by the *sampling set* $\mathcal{S} \subseteq \mathcal{V}$. In this context, let $\mathbf{D}_{\mathcal{S}} \in \mathbb{R}^{N\times N}$ denote a diagonal matrix with entries $d_{\mathcal{S}_n}$, where $d_{\mathcal{S}_n} = 1$ if $v_n \in \mathcal{S}$ and $d_{\mathcal{S}_n} = 0$ otherwise. Thus, one can write the sampled vector $\mathbf{x}_{\mathcal{S}} \in \mathbb{R}^N$ as

$$\mathbf{x}_{\mathcal{S}} = \mathbf{D}_{\mathcal{S}}\mathbf{x}_{\mathrm{G}}\,. \tag{4.12}$$

In order to recover a original bandlimited signal $\mathbf{x}_{\mathrm{o}}$ from its sampled version $\mathbf{x}_{\mathcal{S}}$, that is $\mathbf{x}_{\mathrm{o}} = \mathbf{\Phi}\mathbf{D}_{\mathcal{S}}\mathbf{x}_{\mathrm{o}}$, we remember the IGFT expression of an $\mathcal{F}$-ssparse signal in (4.10) and verify that, when $\left(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\right)$ has full rank, the interpolation matrix $\mathbf{\Phi}$ can be chosen as [10, 11]

$$\mathbf{\Phi} = \mathbf{U}_{\mathcal{F}}\left(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\right)^{-1}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\,. \tag{4.13}$$

Thus, by considering expressions (4.12) and (4.13), it is clear that the sampling and interpolation procedure described by $\mathbf{\Phi}\mathbf{D}_{\mathcal{S}}\mathbf{x}_{\mathrm{o}}$ results in the original graph signal

$\mathbf{x}_\mathrm{o} = \mathbf{U}_\mathcal{F}\mathbf{s}_\mathcal{F}$ [11, 71]. This conclusion follows from observing that

$$\begin{aligned}\mathbf{\Phi}\mathbf{D}_\mathcal{S}\mathbf{x}_\mathrm{o} &= \left[\mathbf{U}_\mathcal{F}\left(\mathbf{U}_\mathcal{F}^\mathrm{T}\mathbf{D}_\mathcal{S}\mathbf{U}_\mathcal{F}\right)^{-1}\mathbf{U}_\mathcal{F}^\mathrm{T}\right]\left[\mathbf{D}_\mathcal{S}\right]\left[\mathbf{U}_\mathcal{F}\mathbf{s}_\mathcal{F}\right] \\ &= \mathbf{U}_\mathcal{F}\Big[\left(\mathbf{U}_\mathcal{F}^\mathrm{T}\mathbf{D}_\mathcal{S}\mathbf{U}_\mathcal{F}\right)^{-1}\left(\mathbf{U}_\mathcal{F}^\mathrm{T}\mathbf{D}_\mathcal{S}\mathbf{U}_\mathcal{F}\right)\Big]\mathbf{s}_\mathcal{F} \\ &= \mathbf{U}_\mathcal{F}\mathbf{s}_\mathcal{F} = \mathbf{x}_\mathrm{o}\,.\end{aligned} \tag{4.14}$$

Therefore, we conclude that perfect reconstruction of an $\mathcal{F}$-ssparse graph signal from its sampled version $\mathbf{x}_\mathcal{S}$ is possible as long as the chosen sampling set $\mathcal{S}$ guarantees that [10, 11]

$$\mathrm{rank}(\mathbf{D}_\mathcal{S}\mathbf{U}_\mathcal{F}) = |\mathcal{F}|\,. \tag{4.15}$$

As $\mathrm{rank}(\mathbf{D}_\mathcal{S}) = |\mathcal{S}|$, from (4.15) we conclude that a necessary condition for perfect recovery of a sampled graph signal is that $|\mathcal{S}| \geq |\mathcal{F}|$ [11, 71], i.e., the number of samples retained must be at least the amount of non-zero frequency components of $\mathbf{s}$. This condition is not sufficient, though, and for guaranteeing perfect reconstruction the sampling set $\mathcal{S}$ must be chosen in such a way that (4.15) is satisfied. This fact clarifies the importance of an adequate choice for the sampling set $\mathcal{S}$ and its connection to the graph structure $\mathbf{U}_\mathcal{F}$, a topic discussed in the next subsection.

4.3.3 Sampling Set Selection

As so far we have considered recovering a bandlimited graph signal $\mathbf{x}_\mathrm{o}[k]$ from sampled values $\mathbf{x}_\mathcal{S}[k]$, any choice of *sampling set* $\mathcal{S}$ respecting condition (4.15) results in a perfect recovery of the original graph signal when applying the interpolation matrix in (4.13). However, in a practical situation where one acquires data from distributed sensors we expect the obtained measurements to be corrupted by noise, which influences the performance of the traditional reconstruction method.

Based on this assumption, a more adequate modeling for a practical graph signal obtained from distributed measurements across a sensor network is represented by

$$\boldsymbol{x}_w[k] = \mathbf{x}_\mathrm{o}[k] + \boldsymbol{w}[k]\,, \tag{4.16}$$

where $\boldsymbol{x}_w[k]$ is the noisy random signal[1] available at the vertices of the graph, $\mathbf{x}_\mathrm{o}[k] \in \mathbb{R}^N$ is the original bandlimited graph signal, and $\boldsymbol{w}[k]$ is a zero-mean noise vector with covariance matrix $\mathbf{C}_w[k] \in \mathbb{R}^{N\times N}$.

Considering the noisy model in (4.16), if one evaluates some common figures of merit (FoM) for an estimated graph signal $\hat{\mathbf{x}}_\mathrm{o}[k]$, such as the *mean-squared deviation* (MSD_G) defined as

$$\mathrm{MSD}_\mathrm{G} = \mathbb{E}\left\{\|\hat{\boldsymbol{x}}_\mathrm{o}[k] - \mathbf{x}_\mathrm{o}[k]\|_2^2\right\}\,, \tag{4.17}$$

[1] In this work, $\boldsymbol{x}$ denotes a random vector with realizations denoted as $\mathbf{x}$.

or the *squared deviation* ($\mathrm{SD_G}$) given by

$$\mathrm{SD_G} = \|\hat{\mathbf{x}}_\mathrm{o}[k] - \mathbf{x}_\mathrm{o}[k]\|_2^2 \,, \tag{4.18}$$

it is simple to verify that an estimate $\hat{\mathbf{x}}_\mathrm{o}[k]$ based on the linear interpolation procedure $\mathbf{\Phi}\mathbf{D}_\mathcal{S}$, where $\mathbf{\Phi}$ is given by (4.13), results in

$$\begin{aligned} \mathrm{MSD_G}(\mathcal{S}) &= \mathbb{E}\left\{\|\left(\mathbf{U}_\mathcal{F}^\mathrm{T}\mathbf{D}_\mathcal{S}\mathbf{U}_\mathcal{F}\right)^{-1}\mathbf{U}_\mathcal{F}^\mathrm{T}\mathbf{D}_\mathcal{S}\boldsymbol{w}[k]\|_2^2\right\} \,, \\ \mathrm{SD_G}(\mathcal{S}) &= \|(\mathbf{U}_\mathcal{F}^\mathrm{T}\mathbf{D}_\mathcal{S}\mathbf{U}_\mathcal{F})^{-1}\mathbf{U}_\mathcal{F}^\mathrm{T}\mathbf{D}_\mathcal{S}\mathbf{w}[k]\|_2^2 \,, \end{aligned} \tag{4.19}$$

where we recalled that $\mathbf{U}_\mathcal{F}^\mathrm{T}\mathbf{U}_\mathcal{F} = \mathbf{I}$. Both expressions in (4.19) present an explicit dependency on $\mathbf{D}_\mathcal{S}$, which indicates that the sampling set $\mathcal{S}$ must be properly chosen to reduce a desired FoM. Nonetheless, the resulting optimization problem of choosing N' indices from the N available in $\mathcal{N} = \{1, 2, ..., N\}$ to form the set $\mathcal{S}$ is combinatorial in nature, requiring an intensive computational procedure for large N since the number of possible sets is given by

$$\binom{N}{N'} = \frac{N!}{(N-N')!N'!} \,. \tag{4.20}$$

Although the number of possible sets is just $\binom{5}{3} = 10$ for the didactic bandlimited graph signal with frequency representation described by (4.11), this number assumes huge orders for certain real-world problems where many nodes are considered. For instance, for the temperature measurements illustrated in Figure 4.4 and further analyzed in Section 6.1, the parameters $N = 299$ and $N' = 210$ provide a total of $\frac{299!}{89!210!}$ sets to be tested for finding the one that results in the smallest chosen metric.

Hence, in order to obtain an interesting trade-off between reasonable reconstruction performance and time required for calculating the sampling set $\mathcal{S}$, a common approach is to employ a greedy algorithm for minimizing a specific FoM. A greedy algorithm basically reduces the overall computational complexity by searching for an optimal selection at each stage and expecting to find a near optimal final value. Detailed information regarding the reconstruction performance of greedy strategies in GSP is presented in [12].

Particularly, in this work we consider the sampling set method displayed in Algorithm 2, which employs at each iteration a greedy search for the index $n \in \mathcal{N}$ to be added to the current set $\mathcal{S}$ in order to maximize the minimum non-negative eigenvalue of $(\mathbf{U}_\mathcal{F}^\mathrm{T}\mathbf{D}_{\mathcal{S}\cup\{n\}}\mathbf{U}_\mathcal{F})$. This method has been initially suggested in [11] and takes the same form as one of the sampling strategies described in [15]. In fact, following an idea similar to [11], it can be shown that Algorithm 2 uses a greedy scheme for minimizing the $\mathrm{SD_G}$ in (4.19).

Algorithm 2 Greedy algorithm for selection of $\mathcal{S}$

$\mathcal{S} \leftarrow \emptyset$
while $|\mathcal{S}| < N'$ **do**
  $n' = \underset{n \in \mathcal{N}}{\operatorname{argmax}} \ \lambda^{+}_{\min}(\mathbf{U}^{\mathrm{T}}_{\mathcal{F}} \mathbf{D}_{\mathcal{S} \cup \{n\}} \mathbf{U}_{\mathcal{F}})$
  $\mathcal{S} \leftarrow \mathcal{S} + \{n'\}$
end
return $\mathcal{S}$

Although more general selection approaches allow adaptive graph sampling [15, 17, 19, 20], in which $\mathbf{D}_{\mathcal{S}}[k]$ might change at each instant k, this dissertation assumes a static sampling matrix $\mathbf{D}_{\mathcal{S}}$ for simplifying some of its future derivations. This premise relies on a practical appeal since, in many sensor network scenarios, the sampled nodes are fixed and previously defined due to external particularities of the problem, such as the costs of sensor placement. In mathematical terms, this assumption implies there is prior knowledge about the signal representation in the frequency domain, which defines $\mathbf{U}_{\mathcal{F}}$, while the adaptive sampling idea is more suitable for the cases where the support $\mathcal{F}$ is unknown. In particular, for the recovery of bandlimited graph signals scenarios analyzed ahead, we consider the existence of two previously known static matrices $\mathbf{D}_{\mathcal{S}}$ and $\mathbf{U}_{\mathcal{F}}$. Although this assumption might not be realistic for some practical applications, such as when the graph structure is time-varying, it is still useful because it allows us to focus exclusively on the adaptive algorithms for online reconstruction of bandlimited graph signals presented in the remaining parts of this work.

4.4 Adaptive Estimation of Graph Signals

A first attempt to merge the traditional area of adaptive filtering [29], discussed in Chapter 2, with the brand-new field of GSP is done in [15], where the authors suggest LMS-based strategies for handling the problem of graph signal reconstruction. Soon after, an RLS-based algorithm is also proposed in [18] for an identical estimation task. These adaptive approaches for graph reconstruction are inspired by the possibility of robust online estimation and tracking of time-varying graph signals. The robustness of the adaptive approach is handy to work in noisy scenarios such as (4.16), when we expect an adaptive strategy to provide a smaller $\mathrm{MSD}_{\mathrm{G}}$ in comparison to an instantaneous signal interpolation.

Basically, the adaptive estimation of graph signals deals with a scenario where one intends to recover a bandlimited, or approximately bandlimited, reference GS $\mathbf{x}_{\mathrm{o}}[k] \in \mathbb{R}^N$ from a reduced set $\mathcal{S} \subseteq \mathcal{V}$ of noisy measurements represented as $\boldsymbol{x}_w[k]$ in (4.16). Since only the node signals indexed by $\mathcal{S}$ are acquired, the reference

measurements at time instant k are in fact $\mathbf{D}_{\mathcal{S}}\mathbf{x}_w[k]$. Then, we define the estimation error vector $\mathbf{e}[k] \in \mathbb{R}^N$ as the difference between the measured noisy signal $\mathbf{D}_{\mathcal{S}}\mathbf{x}_w[k]$ and the respective positions of the current estimate $\hat{\mathbf{x}}_{\mathrm{o}}$, i.e.,

$$\mathbf{e}[k] = \mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[k] - \mathbf{U}_{\mathcal{F}}\hat{\mathbf{s}}_{\mathcal{F}}[k])\,, \tag{4.21}$$

where $\hat{\mathbf{s}}_{\mathcal{F}}[k] \in \mathbb{R}^{|\mathcal{F}|}$ is the frequency-domain estimate of $\hat{\mathbf{x}}_{\mathrm{o}}[k]$, as given in (4.10).

Although both standard and GSP adaptive scenarios are motivated by similar ideas, the corresponding instantaneous errors for the adaptive filtering framework in (2.3) and its recast version for the GSP context in (4.21) have distinct dimensions. Besides that, (4.21) replaces the influence of the input signal $\mathbf{x}[k]$ in (2.3) with the graph structure represented by $\mathbf{U}_{\mathcal{F}}$, taken here as time invariant. Thus, due to these slight differences, we expect the GS estimation algorithms to resemble their classical counterparts, yet presenting some peculiarities. To investigate these differences, we present the GSP LMS [15] and RLS [18] algorithms and establish the equivalent FoMs for evaluating the performance of adaptive methods for GS estimation.

4.4.1 The GSP LMS Algorithm

As argued in Section 2.2, the LMS algorithm [26, 29] stands out as one of the most popular techniques due to its simple implementation and reduced computational complexity. Based on the theoretical Wiener filter formulation, the LMS method takes a practical approach by replacing the minimization of the MSE in (2.4) with the minimization of the instantaneous squared error in (2.3) to define its expression for computing the coefficients vector update $\hat{\mathbf{h}}[k+1]$ as in (2.7).

In an attempt to produce an LMS-based equivalent algorithm for the GS reconstruction context, [15] considers the error signal available as $\mathbf{e}[k]$ from (4.21) and, with a clear inspiration from the Wiener filter idea, defines a reference convex problem as

$$\min_{\mathbf{s}_{\mathcal{F}}} \mathbb{E}\left[\, \|\mathbf{D}_{\mathcal{S}}(\boldsymbol{x}_w[k] - \mathbf{U}_{\mathcal{F}}\mathbf{s}_{\mathcal{F}})\|_2^2 \,\right]\,. \tag{4.22}$$

Similarly to the original LMS, the GSP LMS algorithm in [15, 19] employs a stochastic gradient approach to solve (4.22) and finds an update expression for $\hat{\mathbf{s}}_{\mathcal{F}}[k+1]$. Then, from the IGFT in (4.7) one easily obtains a vertex-domain estimate $\hat{\mathbf{x}}_{\mathrm{o}}[k]$ for the bandlimited GS $\mathbf{x}_{\mathrm{o}}[k]$, which corresponds to the GSP LMS update equation [15, 19]

$$\hat{\mathbf{x}}_{\mathrm{o}}[k+1] = \hat{\mathbf{x}}_{\mathrm{o}}[k] + \mu_{\mathrm{L}}\mathbf{U}_{\mathcal{F}}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{e}[k]\,, \tag{4.23}$$

where $\mu_{\mathrm{L}} \in \mathbb{R}_+$ is a design parameter whose purpose is to control the trade-off between increasing the convergence speed and reducing the steady-state error. Due to its resemblance to the classical LMS parameter μ_{l} in Section 2.2, μ_{L} is also called

convergence factor. An analysis about the range of μ_{L} that guarantees algorithm stability is presented in [15, 19], while the simple LMS-based procedure for the online estimation of a graph signal using is displayed in Algorithm 3.

As discussed in Chapter 2, alternative LMS-based algorithms have been proposed in order to take advantage of unexplored features of the original method, mainly for enhancing its convergence speed. A particular LMS-based algorithm that is worth mentioning is the normalized least-mean-squares (NLMS) algorithm [28, 29] analyzed in Section 2.4, which usually improves the convergence speed by using a time-varying convergence factor. Considering these issues, we shall extend the idea of finding LMS-based strategies for GS estimation [15, 21] and propose the GSP NLMS algorithm in Section 5.1.

Algorithm 3 LMS estimation of graph signals

1: $k \leftarrow 0$
2: **while** (true) **do**
3: $\quad \mathbf{e}[k] = \mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[k] - \hat{\mathbf{x}}_{\mathrm{o}}[k])$
4: $\quad \hat{\mathbf{x}}_{\mathrm{o}}[k+1] = \hat{\mathbf{x}}_{\mathrm{o}}[k] + \mu_{\mathrm{L}}\, \mathbf{U}_{\mathcal{F}}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{e}[k]$
5: $\quad k \leftarrow k+1$
6: **end**

4.4.2 The GSP RLS Algorithm

An alternative approach to enhance convergence speed with respect to the LMS algorithm is to consider a different cost function, like the weighted least-squares (WLS) from which the original RLS algorithm in Section 2.3 is derived [28, 29]. By following a similar idea, the authors in [19] propose the GSP RLS via a centralized version of the RLS method for online reconstruction of graph signals, which results in an algorithm with faster convergence but higher computational burden than the GSP LMS, as verified in [18, 31].

Instead of finding an instantaneous solution to the convex problem (4.22), the GSP RLS [19] evaluates its frequency-domain update estimate $\hat{\mathbf{s}}_{\mathcal{F}}[k+1]$ by minimizing the objective function

$$\min_{\mathbf{s}_{\mathcal{F}}} \sum_{l=1}^{k} \beta_{\mathrm{R}}^{k-l} \|\mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[l] - \mathbf{U}_{\mathcal{F}}\mathbf{s}_{\mathcal{F}})\|^2_{\mathbf{C}_w^{-1}[k]} + \beta_{\mathrm{R}}^{k} \|\mathbf{s}_{\mathcal{F}}\|^2_{\mathbf{\Pi}}, \tag{4.24}$$

where the forgetting factor β_{R} is in the range $0 \ll \beta_{\mathrm{R}} \leq 1$, and the regularization matrix $\mathbf{\Pi}$, which is usually taken as $\mathbf{\Pi} = \delta_{\mathrm{R}}\mathbf{I}$ with a small $\delta_{\mathrm{R}} > 0$, is included to account for ill-conditioning in the first iterations.

An online method for evaluating the WLS solution of (4.24) employs the ancillary

variables $\mathbf{R}_{\mathrm{R}}[k] \in \mathbb{R}^{|\mathcal{F}|\times|\mathcal{F}|}$ and $\mathbf{p}_{\mathrm{R}}[k] \in \mathbb{R}^{|\mathcal{F}|}$ defined recursively as

$$\begin{aligned}\mathbf{R}_{\mathrm{R}}[k] &= \beta_{\mathrm{R}}\mathbf{R}_{\mathrm{R}}[k-1] + \mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_{w}^{-1}[k]\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\,,\\ \mathbf{p}_{\mathrm{R}}[k] &= \beta_{\mathrm{R}}\mathbf{p}_{\mathrm{R}}[k-1] + \mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_{w}^{-1}[k]\mathbf{D}_{\mathcal{S}}\mathbf{x}_{w}[k]\,.\end{aligned} \tag{4.25}$$

From these variables, the solution $\hat{\mathbf{s}}_{\mathcal{F}}[k+1]$ of (4.24) satisfies $\mathbf{R}_{\mathrm{R}}[k]\,\hat{\mathbf{s}}_{\mathcal{F}}[k+1] = \mathbf{p}_{\mathrm{R}}[k]$. Since $\mathbf{R}_{\mathrm{R}}[k]$ has full rank, one finds from (4.7) that the estimate $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$ for the RLS algorithm is

$$\hat{\mathbf{x}}_{\mathrm{o}}[k+1] = \mathbf{U}_{\mathcal{F}}\mathbf{R}_{\mathrm{R}}^{-1}[k]\mathbf{p}_{\mathrm{R}}[k]. \tag{4.26}$$

In particular, it is worth mentioning that equations in (4.25) are more general than the ones in [18, 19] because they allow the symmetric matrix $\mathbf{C}_w[k]$ to assume non-diagonal structures, a case not covered in the original work. Moreover, though the GSP RLS algorithm in [18, 19] suggests the use of (4.25) and (4.26) for computing the estimate $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$, Appendix A.2 shows that these expressions are equivalent to

$$\begin{aligned}\mathbf{R}_{\mathrm{R}}[k] \quad &= \beta_{\mathrm{R}}\mathbf{R}_{\mathrm{R}}[k-1] + \mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_{w}^{-1}[k]\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\,,\\ \hat{\mathbf{x}}_{\mathrm{o}}[k+1] &= \hat{\mathbf{x}}_{\mathrm{o}}[k] + \mathbf{U}_{\mathcal{F}}\mathbf{R}_{\mathrm{R}}^{-1}[k]\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_{w}^{-1}[k]\mathbf{e}[k]\,.\end{aligned} \tag{4.27}$$

Then, in contrast to its original expanded form in [18, 31], we explicitly present the GSP RLS procedure in Algorithm 4 using its compact representation (4.27) and the initialization $\mathbf{R}_{\mathrm{R}}[-1] = \mathbf{\Pi}$ suggested in [18, 19].

Additionally, by considering $\mathbf{R}_{\mathrm{R}}[-1] = \mathbf{\Pi}$ and $\mathbf{C}_w^{-1}[k] = \mathbf{C}_w^{-1}$, we rewrite $\mathbf{R}_{\mathrm{R}}[k]$ in (4.27) as

$$\mathbf{R}_{\mathrm{R}}[k] = \beta_{\mathrm{R}}^{k+1}\mathbf{\Pi} + (\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_{w}^{-1}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})\frac{(1-\beta_{\mathrm{R}}^{k})}{(1-\beta_{\mathrm{R}})}\,. \tag{4.28}$$

Particularly, as $k \to \infty$, one has that $\beta_{\mathrm{R}}^{k} \to 0$. Thus, by defining

$$\mathbf{M}_{\mathrm{R}} \;=\; (\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_{w}^{-1}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})^{-1}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\,, \tag{4.29}$$

we verify that, when k increases, the GSP RLS update tends to the simple expression

$$\hat{\mathbf{x}}_{\mathrm{o}}[k+1] = \hat{\mathbf{x}}_{\mathrm{o}}[k] + (1-\beta_{\mathrm{R}})\mathbf{U}_{\mathcal{F}}\mathbf{M}_{\mathrm{R}}\mathbf{C}_{w}^{-1}\mathbf{e}[k]\,, \tag{4.30}$$

which will be further explored for providing steady-state error metrics at the algorithm complementary analysis in Section 5.2.

4.4.3 Figures of Merit

Relying on the same motivation mentioned in Section 2.1, where we discussed the use of metrics for comparing the performance of traditional adaptive filtering algorithms,

Algorithm 4 RLS estimation of graph signals

$k \leftarrow 0$
$\mathbf{R}_{\mathrm{R}}[-1] = \mathbf{\Pi}$
$\mathbf{B}_{\mathrm{R}} = \mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}$
while (true) **do**
  $\mathbf{R}_{\mathrm{R}}[k] = \beta_{\mathrm{R}}\mathbf{R}_{\mathrm{R}}[k-1] + \mathbf{B}_{\mathrm{R}}\mathbf{C}_{w}^{-1}[k]\mathbf{B}_{\mathrm{R}}^{\mathrm{T}}$
  $\hat{\mathbf{x}}_{\mathrm{o}}[k+1] = \hat{\mathbf{x}}_{\mathrm{o}}[k] + \mathbf{U}_{\mathcal{F}}\mathbf{R}_{\mathrm{R}}^{-1}[k]\mathbf{B}_{\mathrm{R}}\mathbf{C}_{w}^{-1}[k]\mathbf{e}[k]$
  $k \leftarrow k+1$
end

we naturally wish to adopt similar figures of merit for the brand new idea of using adaptive algorithms in GSP. Considering expressions (2.4) and (2.5), the extension of the traditional MSE and MSD to the GSP estimation context is straightforward, being given by

$$\mathrm{MSE}_{\mathrm{G}}[k] = \mathbb{E}\{\|\boldsymbol{e}[k]\|_2^2\} \text{ and } \mathrm{MSD}_{\mathrm{G}}[k] = \mathbb{E}\{\|\Delta\hat{\boldsymbol{x}}_{\mathrm{o}}[k]\|_2^2\}, \tag{4.31}$$

where $\Delta\hat{\boldsymbol{x}}_{\mathrm{o}}[k] = \hat{\boldsymbol{x}}_{\mathrm{o}}[k] - \mathbf{x}_{\mathrm{o}}[k]$ is the difference between the current estimator $\hat{\boldsymbol{x}}_{\mathrm{o}}[k]$ and the original GS $\mathbf{x}_{\mathrm{o}}[k]$, and we use the subscript G for avoiding confusion with (2.4) and (2.5). In particular, for bandlimited graph signals, if we define

$$\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k] = \hat{\boldsymbol{s}}_{\mathcal{F}}[k] - \mathbf{s}_{\mathcal{F}}[k]\,, \tag{4.32}$$

and use the compact representation in (4.10) and the property $\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{U}_{\mathcal{F}} = \mathbf{I}$, from (4.31) we find that the $\mathrm{MSD}_{\mathrm{G}}$ is equivalent to

$$\mathrm{MSD}_{\mathrm{G}}[k] = \mathbb{E}\{\|\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k]\|_2^2\}\,. \tag{4.33}$$

A disadvantage of using the scalar metric $\mathrm{MSE}_{\mathrm{G}}[k]$ in (4.31) is that it potentially hides the occurrence of large error entries in (4.21). Then, we define an alternative FoM for estimating each component of the error vector $\mathbf{e}[k]$. This more general FoM relies on statistics of $e_n[k]$, the n^{th} component of $\mathbf{e}[k]$ in (4.21), and provides a more accurate insight of the algorithm overall performance. Note that, from (4.16), (4.21), and (4.32), one has

$$e_n[k] = d_n(w_n[k] - \mathbf{u}_{n_{\mathcal{F}}}^{\mathrm{T}}\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k])\,, \tag{4.34}$$

where $d_n \in \{0,1\}$, $w_n[k]$ is the n^{th} entry of $\boldsymbol{w}[k]$, and $\mathbf{u}_{n_{\mathcal{F}}}^{\mathrm{T}}$ is the n^{th} row of $\mathbf{U}_{\mathcal{F}}$.

If one works with an asymptotically unbiased estimator $\hat{\boldsymbol{s}}_{\mathcal{F}}[k]$ such that $\mathbb{E}[\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k]]$ converges to the null vector in steady state, which holds true for both GSP LMS and RLS algorithms [19], and by recalling that the noise vector $\boldsymbol{w}[k]$ has zero mean, then one has $\mathbb{E}[e_n[k]] \to 0$ as k grows to infinity.

By assuming that $\boldsymbol{w}[k]$ is uncorrelated with $\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k]$, then taking the expected

value of the squared expression in (4.34) allows one to compute the steady-state error variance

$$\sigma_{e_n}^2 = \lim_{k\to\infty} d_n(\,\mathbb{E}[w_n^2[k]] + \mathbf{u}_{n_\mathcal{F}}^{\mathrm{T}}\mathbb{E}[\Delta\hat{\boldsymbol{s}}_\mathcal{F}[k]\Delta\hat{\boldsymbol{s}}_\mathcal{F}^{\mathrm{T}}[k]]\mathbf{u}_{n_\mathcal{F}}), \tag{4.35}$$

thus yielding, from (4.31), the following steady-state $\mathrm{MSE_G}$:

$$\mathrm{MSE_G^*} = \lim_{k\to\infty} \mathrm{MSE_G}[k] = \sum_{n=1}^{N} \sigma_{e_n}^2\,. \tag{4.36}$$

Similarly, based on the $\mathrm{MSD_G}$ in (4.33), one can also define

$$\mathrm{MSD_G^*} = \lim_{k\to\infty} \mathrm{MSD}[k] = \lim_{k\to\infty} \mathrm{tr}\{\,\mathbb{E}[\Delta\hat{\boldsymbol{s}}_\mathcal{F}[k]\Delta\hat{\boldsymbol{s}}_\mathcal{F}^{\mathrm{T}}[k]]\,\}, \tag{4.37}$$

in which the matrix trace $\mathrm{tr}\{\cdot\}$ operator is employed to show the dependency of this FoM on the steady-state matrix $\mathbb{E}[\Delta\hat{\boldsymbol{s}}_\mathcal{F}[\infty]\Delta\hat{\boldsymbol{s}}_\mathcal{F}^{\mathrm{T}}[\infty]]$. This matrix plays a central role in this work, being the first one to be computed for each algorithm in order to evaluate expressions (4.35), (4.36), and (4.37).

Although the stationary value $\mathrm{MSD_G^*}$ is provided for the GSP RLS algorithm in [19] and approximated for the GSP LMS algorithm in small convergence factor scenarios in [15, 19], Section 5.2 generalizes these results and obtain closed formulas for evaluating the $\mathrm{MSD_G^*}$ and $\mathrm{MSE_G^*}$ for the GSP LMS and RLS algorithms in any situation. Moreover, considering the GSP NLMS algorithm proposed in Chapter 5, we also provide the same stationary figures of merit for the new adaptive algorithm for graph signal estimation. Additionally, the solid results about the individual error estimates in (4.35) for the LMS, RLS, and NLMS algorithms are used as the basis for the data-selective adaptive strategies suggested in Section 5.3.

Chapter 5

NLMS Algorithm and DS Strategies for GSP

Inspired by the LMS and RLS methods described in Section 4.4 for solving the GSP problem of online reconstructing bandlimited graph signals, one certainly longs for further exploring the adaptive algorithms from Chapter 2 in order to provide alternative methods to the basic algorithms proposed so far. Considering the well-known advantages of the normalized LMS (NLMS) over both the LMS and the RLS algorithms in the traditional adaptive filtering context, being usually faster than the LMS and much less complex than the RLS, the appeal for deriving an NLMS-based procedure for the GSP framework is clear. Thus, based on the traditional NLMS derivation from Section 2.4 we propose the GSP NLMS algorithm in Section 5.1, which intends to provide a faster convergence algorithm but with the same computational complexity of the GSP LMS method from Subsection 4.4.1.

Besides its derivation, Section 5.1 also presents a complete analysis on the proposed GSP NLMS algorithm, which includes: the internal parameter selection range that guarantees the algorithm stability and convergence to an unbiased solution; a computational complexity comparison in terms of flops with the GSP LMS and RLS methods; remarks highlighting the beneficial effects of using the NLMS algorithm instead of the similar LMS procedure; and closed formulas for the stationary figures of merit (FoMs) suggested in Subsection 4.4.3. In particular, these steady-state FoMs are also obtained for the LMS and RLS algorithms in Section 5.2, complementing the previous analysis presented in [19]. Then, this work provides the expressions for evaluating the error metrics suggested in Subsection 4.4.3 for all adaptive GSP algorithms proposed until this moment.

At last, considering the appeal of using the *data-selection* idea, presented in Chapter 3, for reducing the overall computational complexity in adaptive filtering algorithms, which results in useful power savings depending on the application, we propose implementing the simple data-selective strategy described in Subsec-

tion 3.1.1 along adaptive GSP algorithms. Due to the multidimensional error signal used in the GSP context, in contrast to the scalar error in traditional adaptive filtering, we suggest two different approaches for assessing the novelty brought by the input data and implement the data-selective strategy with the adaptive GSP algorithms in both approaches. Moreover, as precise estimates are obtained for the stationary individual error variances $\sigma_{e_n}^2$ in Sections 5.1 and 5.2, Subsections 5.3.1 and 5.3.2 provide constraint vector choices and expressions that directly relate an algorithm internal parameter, known as the update factor κ, to the expected update rate when using the GSP LMS, RLS and NLMS algorithms.

Therefore, this chapter contains the main original contributions of this dissertation, which intends to expand the current adaptive GSP framework by proposing an NLMS-based algorithm and adopting data selection ideas for graph signal applications. Both of these contributions are supported by a detailed analysis in the present chapter and corroborated by numerical simulations in Chapter 6.

5.1 NLMS Graph Signal Estimation Algorithm

5.1.1 Algorithm Derivation

Inspired by the traditional NLMS algorithm derivation in Section 2.4, we search for a possibly time-varying factor $\mu_{\mathrm{L}} \in \mathbb{R}_+$ that improves the overall convergence rate of the GSP LMS algorithm described in Subsection 4.4.1. However, as in the GSP context one has to deal with an error vector $\mathbf{e}[k]$ instead of a scalar error $e[k]$, we generalize this idea of a convergence factor μ_{L} and adopt a possibly time-varying symmetric convergence matrix $\mathbf{M}_{\mathrm{L}}[k] \in \mathbb{R}^{|\mathcal{F}|\times|\mathcal{F}|}$. Then, from (4.23) the frequency-domain update equation becomes

$$\hat{\mathbf{s}}_{\mathcal{F}}[k+1] = \hat{\mathbf{s}}_{\mathcal{F}}[k] + \mathbf{M}_{\mathrm{L}}[k]\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[k] - \mathbf{U}_{\mathcal{F}}\hat{\mathbf{s}}_{\mathcal{F}}[k])\,. \tag{5.1}$$

Following the same reasoning for defining the GSP error vector $\mathbf{e}[k]$ in (4.21), we can also define the *a posteriori* reconstruction error vector $\boldsymbol{\varepsilon}[k]$ as

$$\boldsymbol{\varepsilon}[k] = \mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[k] - \mathbf{U}_{\mathcal{F}}\hat{\mathbf{s}}_{\mathcal{F}}[k+1])\,. \tag{5.2}$$

In order to avoid confusion with the recently defined *a posteriori* error $\boldsymbol{\varepsilon}[k]$, we shall call $\mathbf{e}[k]$ in (4.21) as the *a priori* error.

According to equations (5.1) and (5.2), the *a posteriori* error $\boldsymbol{\varepsilon}[k]$ can be rewritten as function of the *a priori* error, i.e.

$$\boldsymbol{\varepsilon}[k] = \mathbf{D}_{\mathcal{S}}(\mathbf{I} - \mathbf{U}_{\mathcal{F}}\mathbf{M}_{\mathrm{L}}[k]\mathbf{U}_{\mathcal{F}}^{\mathrm{T}})\mathbf{e}[k]\,. \tag{5.3}$$

Since an instantaneous estimate of how close the error vectors are to each other is given by $\Delta\tilde{e}^2[k] = \|\boldsymbol{\varepsilon}[k]\|_2^2 - \|\mathbf{e}[k]\|_2^2$, when using equation (5.3) we find that

$$\Delta\tilde{e}^2[k] = \mathbf{e}^{\mathrm{T}}[k]\mathbf{U}_{\mathcal{F}}(-2\mathbf{M}_{\mathrm{L}}[k] + \mathbf{M}_{\mathrm{L}}[k]\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\mathbf{M}_{\mathrm{L}}[k])\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{e}[k]. \tag{5.4}$$

In order to select the symmetric matrix $\mathbf{M}_{\mathrm{L}}[k]$ that minimizes $\Delta\tilde{e}^2[k]$, we take the derivative of (5.4) with respect to $\mathbf{M}_{\mathrm{L}}[k]$ [72], yielding

$$2\left[(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})\mathbf{M}_{\mathrm{L}}[k] - \mathbf{I}\right]\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{e}[k]\mathbf{e}^{\mathrm{T}}[k]\mathbf{U}_{\mathcal{F}} = \mathbf{0}\,, \tag{5.5}$$

so that the constant symmetric matrix $\mathbf{M}_{\mathrm{L}} = (\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})^{-1}$ minimizes the squared error difference $\Delta\tilde{e}^2[k]$. Based on this result, the frequency-domain update expression given by

$$\hat{\mathbf{s}}_{\mathcal{F}}[k+1] = \hat{\mathbf{s}}_{\mathcal{F}}[k] + (\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})^{-1}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{e}[k] \tag{5.6}$$

should be able to yield faster convergence to its steady-state value than the GSP LMS in Subsection 4.4.1.

As in the original NLMS algorithm from Section 2.4, we also include an additional fixed convergence factor $\mu_{\mathrm{N}} \in \mathbb{R}_+$ for controlling the trade-off between convergence speed and steady state performance, resulting in the vertex-domain update equation

$$\hat{\mathbf{x}}_{\mathrm{o}}[k+1] = \hat{\mathbf{x}}_{\mathrm{o}}[k] + \mu_{\mathrm{N}}\mathbf{U}_{\mathcal{F}}(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})^{-1}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{e}[k]\,. \tag{5.7}$$

For a practical implementation of the GSP NLMS algorithm, one notices that $(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})^{-1}$ can be written as $\mathbf{L}_{\mathrm{N}}\mathbf{L}_{\mathrm{N}}^{\mathrm{T}}$, where $\mathbf{L}_{\mathrm{N}} \in \mathbb{R}^{|\mathcal{F}|\times|\mathcal{F}|}$ is a lower triangular matrix obtained using the Cholesky decomposition. Then, after defining the ancillary matrix $\mathbf{B}_{\mathrm{N}} \in \mathbb{R}^{N\times|\mathcal{F}|}$ as $\mathbf{B}_{\mathrm{N}} = \mathbf{U}_{\mathcal{F}}\mathbf{L}_{\mathrm{N}}$, the suggested implementation of the GSP NLMS is presented in Algorithm 5.

Algorithm 5 NLMS estimation of graph signals

1: $k \leftarrow 0$
2: Find $\mathbf{L}_{\mathrm{N}}$ such that $(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})^{-1} = \mathbf{L}_{\mathrm{N}}\,\mathbf{L}_{\mathrm{N}}^{\mathrm{T}}$
3: $\mathbf{B}_{\mathrm{N}} = \mathbf{U}_{\mathcal{F}}\mathbf{L}_{\mathrm{N}}$
4: **while** (true) **do**
5: $\quad\mathbf{e}[k] = \mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[k] - \hat{\mathbf{x}}_{\mathrm{o}}[k])$
6: $\quad\hat{\mathbf{x}}_{\mathrm{o}}[k+1] = \hat{\mathbf{x}}_{\mathrm{o}}[k] + \mu_{\mathrm{N}}\mathbf{B}_{\mathrm{N}}\mathbf{B}_{\mathrm{N}}^{\mathrm{T}}\mathbf{e}[k]$
7: $\quad k \leftarrow k+1$
8: **end**

Alternatively, by considering the bond between the NLMS and the AP algorithms illustrated in Subsection 2.5.1, a different derivation of the GSP NLMS algorithm can be obtained by solving the following constrained convex problem in the frequency domain: to minimize the distance between the current $\hat{\mathbf{s}}_{\mathcal{F}}[k]$ and the

updated estimate $\hat{\mathbf{s}}_{\mathcal{F}}[k+1]$ (minimum disturbance principle), such that the Fourier transform of the *a posteriori* error $\boldsymbol{\varepsilon}[k]$ is equal to zero on the frequency support $\mathcal{F}$. Mathematically, one has

$$\begin{aligned} &\underset{\hat{\mathbf{s}}_{\mathcal{F}}[k+1]}{\text{minimize}} \quad \|\hat{\mathbf{s}}_{\mathcal{F}}[k+1] - \hat{\mathbf{s}}_{\mathcal{F}}[k]\|_2^2 \\ &\text{subject to} \quad \mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[k] - \mathbf{U}_{\mathcal{F}}\hat{\mathbf{s}}_{\mathcal{F}}[k+1]) = \mathbf{0}\,, \end{aligned} \tag{5.8}$$

whose solution $\hat{\mathbf{s}}_{\mathcal{F}}[k+1]$ is given by (5.6). This result can be simply verified by writing the Lagrangian of (5.8) and following the intermediate steps presented in Appendix A.3. Likewise, by defining a constraint-vector $\boldsymbol{\gamma}[k]$, if we consider $\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\boldsymbol{\varepsilon}[k] = \boldsymbol{\gamma}[k]$ instead of the null *a posteriori* error constraint in (5.8), it is trivial to show that a similar demonstration can lead to the derivation of a SM-NLMS algorithm, an adaptive method with data selection inspired by the SM-PAPA in Section 3.2.

5.1.2 Stability and Convergence to Unbiased Solution

As the GSP NLMS procedure in Algorithm 5 allows one to select different normalized convergence factors for controlling the trade-off between convergence speed and steady-state FoM values, it is essential to determine for which range of μ_{N} values $\hat{\boldsymbol{s}}_{\mathcal{F}}[k]$ is guaranteed to be asymptotically unbiased. As we shall focus on steady-state values, let us assume a time-invariant reference graph signal such that $\mathbf{s}_{\mathcal{F}}[k] = \mathbf{s}_{\mathcal{F}}$ and a time-invariant noise covariance matrix so that $\mathbf{C}_w[k] = \mathbf{C}_w$.

Based on the definition of $\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k]$ in (4.32) and the corresponding frequency-domain representation $\hat{\mathbf{s}}_{\mathcal{F}}$ of (5.7), if we define the ancillary constant matrix $\mathbf{M}_{\mathrm{N}} \in \mathbb{R}^{|\mathcal{F}|\times N}$ as

$$\mathbf{M}_{\mathrm{N}} = (\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})^{-1}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}} \tag{5.9}$$

it follows that

$$\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k+1] = (1-\mu_{\mathrm{N}})\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k] + \mu_{\mathrm{N}}\mathbf{M}_{\mathrm{N}}\boldsymbol{w}[k]\,. \tag{5.10}$$

By taking the expected value on both sides of (5.10) we find the recursive expression

$$\mathbb{E}\{\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k+1]\} = (1-\mu_{\mathrm{N}})\,\mathbb{E}\{\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k]\}\,, \tag{5.11}$$

which allows one to write

$$\mathbb{E}\{\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k]\} = (1-\mu_{\mathrm{N}})^k\,\mathbb{E}\{\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[0]\}\,. \tag{5.12}$$

Considering that $\mathbb{E}\{\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[0]\}$ can be any vector, to guarantee that $\mathbb{E}\{\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k]\}$ in (5.12) converges to a null vector as k increases, we must choose a parameter μ_{N} such that $|1-\mu_{\mathrm{N}}| < 1$. As a result, the interval that guarantees convergence to an

unbiased solution is

$$0 < \mu_{\mathrm{N}} < 2\,. \tag{5.13}$$

In addition, by defining $\mathbf{S}_{\mathrm{N}}[k] = \mathbb{E}[\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k]\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}^{\mathrm{T}}[k]]$, one has from (5.10) that

$$\mathbf{S}_{\mathrm{N}}[k+1] = (1-\mu_{\mathrm{N}})^2\mathbf{S}_{\mathrm{N}}[k] + \mu_{\mathrm{N}}^2\mathbf{M}_{\mathrm{N}}\mathbf{C}_w\mathbf{M}_{\mathrm{N}}^{\mathrm{T}}, \tag{5.14}$$

which is a difference equation that converges to a solution as long as $|1-\mu_{\mathrm{N}}| < 1$, i.e., the condition in (5.13) holds true. In this case, stability is guaranteed and $\mathbb{E}[\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k]\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}^{\mathrm{T}}[k]]$ approaches $\mathbf{S}_{\mathrm{N}}^{*} \in \mathbb{R}^{|\mathcal{F}|\times|\mathcal{F}|}$ as $k \to \infty$, given as

$$\mathbf{S}_{\mathrm{N}}^{*} = \mathbf{S}_{\mathrm{N}}[\infty] = \frac{\mu_{\mathrm{N}}}{2-\mu_{\mathrm{N}}}\mathbf{M}_{\mathrm{N}}\mathbf{C}_w\mathbf{M}_{\mathrm{N}}^{\mathrm{T}}\,. \tag{5.15}$$

Therefore, the μ_{N} parameter range that assures a stable behavior for the proposed NLMS algorithm is more straightforward than the range predicted in [15] for the GSP LMS algorithm, which depends on the eigenvalues of $\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}$. In fact, this well defined convergence range is also an advantage of the traditional NLMS algorithm in comparison to its LMS counterpart [28, 29]. Furthermore, the convergence range discussed in Section 2.4 for the traditional NLMS algorithm agrees with the predicted interval (5.13) obtained for the GSP NLMS algorithm.

5.1.3 Computational Complexity Analysis

In order to provide a fair comparison of the computational complexity among the GSP LMS, RLS, and NLMS algorithms, we estimate the amount of floating-point operations (FLOPs) required to evaluate the estimate $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$ at each iteration. As all algorithms present some common steps, we focus on the differences among them, which basically consist in how the update of $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$ is performed.

According to the LMS and the NLMS practical procedures in Algorithms 3 and 5, which simply differ in the update expression implemented, we assume the $N \times |\mathcal{F}|$ constant matrices $\mathbf{U}_{\mathcal{F}}$ and $\mathbf{B}_{\mathrm{N}}$ to be pre-evaluated structures stored for efficient algorithm implementation. For evaluating $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$ in both cases, we propose using the expressions

$$\begin{aligned}
\hat{\mathbf{x}}_{\mathrm{o}}[k+1] &= \hat{\mathbf{x}}_{\mathrm{o}}[k] + \Big[\,\mathbf{U}_{\mathcal{F}}\Big[\,\mu_{\mathrm{L}}\,(\,\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{e}[k]\,)\,\Big]\Big] \;\text{ and} \\
\hat{\mathbf{x}}_{\mathrm{o}}[k+1] &= \hat{\mathbf{x}}_{\mathrm{o}}[k] + \Big[\,\mathbf{B}_{\mathrm{N}}\Big[\,\mu_{\mathrm{N}}\,(\,\mathbf{B}_{\mathrm{N}}^{\mathrm{T}}\mathbf{e}[k]\,)\,\Big]\Big]\,,
\end{aligned} \tag{5.16}$$

for the GSP LMS and the GSP NLMS algorithms, respectively. Then, for computing $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$ we follow the operations order indicated in (5.16) and first perform a matrix-vector product between a $|\mathcal{F}| \times N$ matrix and the N-dimensional error signal $\mathbf{e}[k]$,

Table 5.1: GSP adaptive filtering algorithms' complexity for computing $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$

Algorithm	$\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$	FLOPs/iteration
LMS	(4.23)	$4\lvert\mathcal{F}\rvert N$
RLS	(4.27)	$\frac{1}{3}\lvert\mathcal{F}\rvert^3 + 4\lvert\mathcal{F}\rvert N + 4\lvert\mathcal{F}\rvert^2 - 2\lvert\mathcal{F}\rvert$
NLMS	(5.7)	$4\lvert\mathcal{F}\rvert N$

which requires $|\mathcal{F}|N$ multiplications and $|\mathcal{F}|(N-1)$ sums for producing a vector with $|\mathcal{F}|$ components. The resulting vector is scaled by the algorithm convergence factor μ_{L} or μ_{N}, increasing by $|\mathcal{F}|$ the amount of scalar multiplications, and yields a new vector with the same dimensionality. The product of an $N \times |\mathcal{F}|$ matrix with this $|\mathcal{F}|$-dimensional vector takes $|\mathcal{F}|N$ multiplications and $N(|\mathcal{F}|-1)$ sums, and the desired update vector is obtained after the summation of this vector with $\hat{\mathbf{x}}_{\mathrm{o}}[k]$, which accounts for an increase of N sums. Thus, the LMS and the NLMS strategies demand $2|\mathcal{F}|N$ multiplication and $2|\mathcal{F}|N$ sum operations, resulting in $4|\mathcal{F}|N$ FLOPs per iteration.

For the RLS algorithm we first assume that $\mathbf{C}_w[k] = \mathbf{C}_w$, which allows us to consider that matrix $(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_w^{-1}) \in \mathbb{R}^{|\mathcal{F}|\times N}$ and $(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_w^{-1}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}) \in \mathbb{R}^{|\mathcal{F}|\times|\mathcal{F}|}$ are fixed and known. In this case, the steps for updating the current system estimate $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$ can be computed as

$$\begin{aligned} &\mathbf{R}_{\mathrm{R}}[k] = (\,\beta_{\mathrm{R}}\mathbf{R}_{\mathrm{R}}[k-1]\,) + (\,\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_w^{-1}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\,), \\ &\hat{\mathbf{x}}_{\mathrm{o}}[k+1] = \hat{\mathbf{x}}_{\mathrm{o}}[k] + \left\{\,\mathbf{U}_{\mathcal{F}}\left[\,\mathbf{R}_{\mathrm{R}}^{-1}[k]\,[\,(\,\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_w^{-1}\,)\,\mathbf{e}[k]\,]\,\right]\right\}. \end{aligned} \tag{5.17}$$

Then, $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$ requires $|\mathcal{F}|(2N+|\mathcal{F}|)$ multiplications and $|\mathcal{F}|(2N+|\mathcal{F}|-2)$ sums for its computation. Although calculating $\mathbf{R}_{\mathrm{R}}[k]$ in (5.17) involves $|\mathcal{F}|^2$ sums and $|\mathcal{F}|^2$ multiplications, the algorithm still requires the inversion of this $|\mathcal{F}| \times |\mathcal{F}|$ matrix, adding $\frac{1}{3}|\mathcal{F}|^3$ FLOPs per iteration via Cholesky factorization [73]. Thus, its overall number of FLOPs per iteration is $\frac{1}{3}|\mathcal{F}|^3 + 4|\mathcal{F}|N + 4|\mathcal{F}|^2 - 2|\mathcal{F}|$.

Table 5.1 contains a summary of these results. This analysis only considered the update expressions for each GSP adaptive algorithm because the remaining steps of the adaptive techniques are the same. Although an efficient implementation of these adaptive algorithms can take advantage from the null values of $\mathbf{e}[k]$ and reduce the amount of arithmetic operations, from TABLE 5.1 it is straightforward to conclude that the RLS algorithm presents a heavier computational burden than the LMS and NLMS methods.

5.1.4 Steady-State FoM Analysis

In this subsection we derive the steady-state values of the FoMs $\sigma_{e_n}^2$, MSE_G^*, and MSD_G^* discussed in Subsection 4.4.3, when using the GSP NLMS.

From (4.35) and (5.15), and by defining $\sigma_{w_n}^2 = \mathbb{E}\{w_n^2[k]\}$, the steady-state value for $\sigma_{e_n}^2$ is given by

$$\sigma_{e_n}^2 = d_n \left[\sigma_{w_n}^2 + \frac{\mu_\text{N}}{2-\mu_\text{N}} \mathbf{u}_{n_\mathcal{F}}^\text{T} \mathbf{M}_\text{N} \mathbf{C}_w \mathbf{M}_\text{N}^\text{T} \mathbf{u}_{n_\mathcal{F}}\right] . \tag{5.18}$$

Moreover, according to (4.36), we find the MSE_G^* for the GSP NLMS by simply summing $\sigma_{e_n}^2$ for all $n \in \mathcal{N}$, which results in

$$\text{MSE}_\text{G}^* = \sum_{n=1}^{N} d_n \left[\sigma_{w_n}^2 + \frac{\mu_\text{N}}{2-\mu_\text{N}} \mathbf{u}_{n_\mathcal{F}}^\text{T} \mathbf{M}_\text{N} \mathbf{C}_w \mathbf{M}_\text{N}^\text{T} \mathbf{u}_{n_\mathcal{F}}\right] . \tag{5.19}$$

Finally, based on (4.37) and (5.15), the MSD_G^* is

$$\text{MSD}_\text{G}^* = \frac{\mu_\text{N}}{2-\mu_\text{N}} \text{tr}\left\{\mathbf{M}_\text{N} \mathbf{C}_w \mathbf{M}_\text{N}^\text{T}\right\} . \tag{5.20}$$

5.1.5 Remarks

Let us get a better feeling regarding the effect of matrix $(\mathbf{U}_\mathcal{F}^\text{T} \mathbf{D}_\mathcal{S} \mathbf{U}_\mathcal{F})^{-1}$ by comparing the GSP LMS and NLMS update equations. As $\{\mathbf{u}_n\}$ is a basis of $\mathbb{R}^N$, then there exists $\boldsymbol{\alpha}[k] \in \mathbb{R}^N$ such that $\mathbf{e}[k] = \mathbf{D}_\mathcal{S} \mathbf{U} \boldsymbol{\alpha}[k]$. Without loss of generality, we can write $\mathbf{U} = [\mathbf{U}_\mathcal{F} \;\; \mathbf{U}_{\overline{\mathcal{F}}}]$ and $\boldsymbol{\alpha}[k] = [\boldsymbol{\alpha}_\mathcal{F}^\text{T}[k] \;\; \boldsymbol{\alpha}_{\overline{\mathcal{F}}}^\text{T}[k]]^\text{T}$, where $\overline{\mathcal{F}} = \mathcal{N} \setminus \mathcal{F}$. Note that $\boldsymbol{\alpha}[k]$ is the frequency-domain representation of $\mathbf{w}[k] + (\mathbf{x}_\text{o}[k] - \hat{\mathbf{x}}_\text{o}[k])$. In this case, vector $(\mathbf{x}_\text{o}[k] - \hat{\mathbf{x}}_\text{o}[k])$ is $\mathcal{F}$-ssparse, which means that $\boldsymbol{\alpha}_{\overline{\mathcal{F}}}[k]$ has only contributions from the measurement noise.

In the GSP LMS algorithm, the error signal $\mathbf{e}[k]$ is multiplied by matrix $\mu_\text{L} \mathbf{B}_\text{L} = \mu_\text{L} \mathbf{U}_\mathcal{F} \mathbf{U}_\mathcal{F}^\text{T}$, thus yielding, in the frequency domain, the correction term

$$\hat{\mathbf{s}}_\mathcal{F}[k+1] - \hat{\mathbf{s}}_\mathcal{F}[k] = \mu_\text{L} (\mathbf{U}_\mathcal{F}^\text{T} \mathbf{D}_\mathcal{S} \mathbf{U}_\mathcal{F}) \boldsymbol{\alpha}_\mathcal{F}[k] + \mathbf{w}_\text{L}[k] , \tag{5.21}$$

where $\mathbf{w}_\text{L}[k] = \mu_\text{L} \left(\mathbf{U}_\mathcal{F}^\text{T} \mathbf{D}_\mathcal{S} \mathbf{U}_{\overline{\mathcal{F}}}\right) \boldsymbol{\alpha}_{\overline{\mathcal{F}}}[k]$ is essentially noise after processing.[1] As for the GSP NLMS algorithm, the error signal is multiplied by matrix $\mu_\text{N} \mathbf{B}_\text{N} = \mu_\text{N} \mathbf{U}_\mathcal{F} (\mathbf{U}_\mathcal{F}^\text{T} \mathbf{D}_\mathcal{S} \mathbf{U}_\mathcal{F})^{-1} \mathbf{U}_\mathcal{F}^\text{T}$, thus yielding, again in the frequency domain, the correction term

$$\hat{\mathbf{s}}_\mathcal{F}[k+1] - \hat{\mathbf{s}}_\mathcal{F}[k] = \mu_\text{N} \boldsymbol{\alpha}_\mathcal{F}[k] + \mathbf{w}_\text{N}[k] , \tag{5.22}$$

where $\mathbf{w}_\text{N}[k] = \mu_\text{N} (\mathbf{U}_\mathcal{F}^\text{T} \mathbf{D}_\mathcal{S} \mathbf{U}_\mathcal{F})^{-1} \left(\mathbf{U}_\mathcal{F}^\text{T} \mathbf{D}_\mathcal{S} \mathbf{U}_{\overline{\mathcal{F}}}\right) \boldsymbol{\alpha}_{\overline{\mathcal{F}}}[k]$.

[1] Roughly speaking, $\mathbf{U}_\mathcal{F}^\text{T} \mathbf{D}_\mathcal{S} \mathbf{U}_{\overline{\mathcal{F}}}$ tends to be close to $\mathbf{0}$, since $\mathbf{U}_\mathcal{F}^\text{T} \mathbf{U}_{\overline{\mathcal{F}}} = \mathbf{0}$.

By comparing expressions (5.21) and (5.22) one can see that the estimation error within the frequency support $\mathcal{F}$ in the NLMS has a clean and direct impact on the correction of the previous estimate $\hat{\mathbf{s}}_{\mathcal{F}}[k]$, without distortions as in the LMS case, imposed by the factor $\left(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\right)$. The so-called normalization of the NLMS is responsible for this effect, which turns out to be a key aspect for enhancing the algorithm performance, as will be demonstrated with numerical simulations in Section 6.2.

In addition, it is noticeable that the proposed NLMS update equation in (5.7) resembles the RLS long-term expression in (4.30), thus indicating that the inclusion of $\left(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\right)^{-1}$ brings about some RLS-like features to the resulting algorithm. Particularly, when the covariance matrix is given by $\mathbf{C}_w = \sigma_w^2\mathbf{I}$, with $\sigma_w^2 > 0$, both algorithms present an equivalent performance for large k if the NLMS convergence factor μ_{N} and the RLS forgetting factor β_{R} follow the relation

$$\mu_{\mathrm{N}} = 1 - \beta_{\mathrm{R}}\,. \tag{5.23}$$

Differently from its traditional counterpart, the GSP NLMS algorithm relies on a fixed normalization term $\left(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\right)^{-1}$ for a constant graph structure, yielding an update expression (5.7) that requires the same computational complexity as the GSP LMS algorithm. Besides, the GSP NLMS algorithm has a strong practical appeal since it presents a well-defined range of values for its convergence factor μ_{N} that guarantees the method stability, in opposition to the equivalent factor choice for the LMS algorithm [15].

5.2 GSP LMS and RLS Complementary Analysis

The analyses of the GSP LMS and RLS algorithms in [19] cover the possibility of signal reconstruction via sparse sampling and the MSD analysis, along with the proposition of optimal sampling strategies. Here, we extend those analyses by generalizing the stationary $\mathrm{MSD}_{\mathrm{G}}$ formula for the GSP LMS algorithm, whose previous expressions assume a small convergence factor μ_{L} approximation, and incorporating the steady-state FoMs $\sigma_{e_n}^2$ and $\mathrm{MSE}_{\mathrm{G}}$ from Subsection 4.4.3 for both LMS and RLS-based methods. Based on the accurate estimates for the error variances evaluated in this section, the GSP LMS and the GSP RLS algorithms can also be used with the data-selection strategies, as we shall see in Section 5.3.

5.2.1 LMS Algorithm Error Analysis

From (4.23) and (4.32), one can write

$$\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k+1] = (\mathbf{I} - \mu_{\mathrm{L}}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k] + \mu_{\mathrm{L}}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\boldsymbol{w}[k]\,, \tag{5.24}$$

thus implying that

$$\begin{aligned}\mathbb{E}[\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k+1]\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}^{\mathrm{T}}[k+1]] = \mu_{\mathrm{L}}^2\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_w\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}} + \\ +(\mathbf{I}-\mu_{\mathrm{L}}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})\mathbb{E}[\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k]\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}^{\mathrm{T}}[k]](\mathbf{I}-\mu_{\mathrm{L}}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})\,.\end{aligned} \tag{5.25}$$

If μ_{L} is in the range that guarantees the algorithm stability [15], then matrix $\mathbb{E}[\Delta\hat{\mathbf{s}}_{\mathcal{F}}[k]\Delta\hat{\mathbf{s}}_{\mathcal{F}}^{\mathrm{T}}[k]]$ converges to $\mathbf{S}_{\mathrm{L}}^* \in \mathbb{R}^{|\mathcal{F}|\times|\mathcal{F}|}$ when $k \to \infty$. Thus, by defining $\mathbf{P}, \mathbf{Q} \in \mathbb{R}^{|\mathcal{F}|\times|\mathcal{F}|}$ such that

$$\mathbf{P} = \mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}} \quad \text{and} \quad \mathbf{Q} = \mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_w\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\,, \tag{5.26}$$

we verify that the stationary expression (5.25) can be written as

$$\begin{aligned}\mathbf{S}_{\mathrm{L}}^* &= \mu_{\mathrm{L}}^2\mathbf{Q} + (\mathbf{I} - \mu_{\mathrm{L}}\mathbf{P})\mathbf{S}_{\mathrm{L}}^*(\mathbf{I} - \mu_{\mathrm{L}}\mathbf{P}) \\ \mathbf{S}_{\mathrm{L}}^* &= \mu_{\mathrm{L}}^2\mathbf{Q} + \mathbf{S}_{\mathrm{L}}^* - \mu_{\mathrm{L}}\mathbf{P}\mathbf{S}_{\mathrm{L}}^* - \mu_{\mathrm{L}}\mathbf{S}_{\mathrm{L}}^*\mathbf{P} + \mu_{\mathrm{L}}^2\mathbf{P}\mathbf{S}_{\mathrm{L}}^*\mathbf{P}\end{aligned} \tag{5.27}$$

where it follows that

$$\mathbf{P}\mathbf{S}_{\mathrm{L}}^* + \mathbf{S}_{\mathrm{L}}^*\mathbf{P} - \mu_{\mathrm{L}}\mathbf{P}\mathbf{S}_{\mathrm{L}}^*\mathbf{P} = \mu_{\mathrm{L}}\mathbf{Q}\,. \tag{5.28}$$

However, for solving the implicit equation (5.28) we notice that it is, in fact, a generalized Lyapunov matrix equation [74], being equivalent to

$$[(\mathbf{I}\otimes\mathbf{P}) + (\mathbf{P}\otimes\mathbf{I}) - \mu_{\mathrm{L}}(\mathbf{P}\otimes\mathbf{P})]\,\mathrm{vec}(\mathbf{S}_{\mathrm{L}}^*) = \mu_{\mathrm{L}}\,\mathrm{vec}(\mathbf{Q}), \tag{5.29}$$

where $\otimes$ indicates the Kronecker product [75] and $\mathrm{vec}(\mathbf{S}_{\mathrm{L}}^*)$ represents the vectorization of $\mathbf{S}_{\mathrm{L}}^*$, performed by stacking its columns into a single column vector. When the left-hand side matrix of (5.29) has full rank, $\mathrm{vec}(\mathbf{S}_{\mathrm{L}}^*)$ is obtained by solving

$$\mathrm{vec}(\mathbf{S}_{\mathrm{L}}^*) = \mu_{\mathrm{L}}\,[(\mathbf{I}\otimes\mathbf{P}) + (\mathbf{P}\otimes\mathbf{I}) - \mu_{\mathrm{L}}(\mathbf{P}\otimes\mathbf{P})]^{-1}\mathrm{vec}(\mathbf{Q}). \tag{5.30}$$

After recovering matrix $\mathbf{S}_{\mathrm{L}}^*$ from its vectorized version, the n^{th} variance $\sigma_{e_n}^2$ for the GSP LMS algorithm is computed by replacing $\mathbf{S}_{\mathrm{L}}^*$ in (4.35), yielding

$$\sigma_{e_n}^2 = d_n\,(\,\sigma_{w_n}^2 + \mathbf{u}_{n_{\mathcal{F}}}^{\mathrm{T}}\mathbf{S}_{\mathrm{L}}^*\mathbf{u}_{n_{\mathcal{F}}}\,)\,, \tag{5.31}$$

and from (4.36)

$$\mathrm{MSE}_{\mathrm{G}}^{*} = \sum_{n=1}^{N} d_n \left(\sigma_{w_n}^2 + \mathbf{u}_{n_{\mathcal{F}}}^{\mathrm{T}} \mathbf{S}_{\mathrm{L}}^{*} \mathbf{u}_{n_{\mathcal{F}}}\right) . \tag{5.32}$$

Moreover, an additional result that comes straightforwardly from the knowledge of $\mathbf{S}_{\mathrm{L}}^{*}$ is the $\mathrm{MSD}_{\mathrm{G}}^{*}$ in (4.37), so that

$$\mathrm{MSD}_{\mathrm{G}}^{*} = \mathrm{tr}\{\mathbf{S}_{\mathrm{L}}^{*}\} . \tag{5.33}$$

Although a steady-state $\mathrm{MSD}_{\mathrm{G}}$ analysis for the LMS algorithm is presented in [15, 19], it is based on an approximation assuming small values of the convergence factor μ_{L}. Thus, it requires a relatively small μ_{L} for providing better estimates of $\mathrm{MSD}_{\mathrm{G}}^{*}$. On the other hand, as the derivation of (5.33) presented in this subsection does not rely on this assumption, it is more general than the previous analysis in [15, 19], being valid for any μ_{L} that guarantees the algorithm stability.

5.2.2 RLS Algorithm Error Analysis

Similarly, the GSP RLS analysis starts by rewriting (4.30) as

$$\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k+1] = \beta_{\mathrm{R}} \Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k] + (1-\beta_{\mathrm{R}}) \mathbf{M}_{\mathrm{R}} \mathbf{C}_w^{-1} \boldsymbol{w}[k] . \tag{5.34}$$

If we evaluate the outer product of (5.34) and take its expected value we find that

$$\begin{aligned} \mathbb{E}[\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k+1] \Delta\hat{\boldsymbol{s}}_{\mathcal{F}}^{\mathrm{T}}[k+1]] = \beta_{\mathrm{R}}^2 \mathbb{E}[\Delta\hat{\boldsymbol{s}}_{\mathcal{F}}[k] \Delta\hat{\boldsymbol{s}}_{\mathcal{F}}^{\mathrm{T}}[k]] + \\ + (1-\beta_{\mathrm{R}})^2 \mathbf{M}_{\mathrm{R}} \mathbf{C}_w^{-1} \mathbf{D}_{\mathcal{S}} \mathbf{C}_w \mathbf{D}_{\mathcal{S}} \mathbf{C}_w^{-1} \mathbf{M}_{\mathrm{R}}^{\mathrm{T}} . \end{aligned} \tag{5.35}$$

Then, by considering the convergence of $\mathbb{E}[\Delta\hat{\mathbf{s}}_{\mathcal{F}}[k] \Delta\hat{\mathbf{s}}_{\mathcal{F}}^{\mathrm{T}}[k]]$ to $\mathbf{S}_{\mathrm{R}}^{*} \in \mathbb{R}^{|\mathcal{F}| \times |\mathcal{F}|}$ as k grows to infinity, one gets

$$\mathbf{S}_{\mathrm{R}}^{*} = \frac{1-\beta_{\mathrm{R}}}{1+\beta_{\mathrm{R}}} \cdot \mathbf{M}_{\mathrm{R}} \mathbf{C}_w^{-1} \mathbf{D}_{\mathcal{S}} \mathbf{C}_w \mathbf{D}_{\mathcal{S}} \mathbf{C}_w^{-1} \mathbf{M}_{\mathrm{R}}^{\mathrm{T}} , \tag{5.36}$$

allowing us to write the corresponding stationary FoMs for the GSP RLS algorithm

$$\sigma_{e_n}^2 = d_n \left(\sigma_{w_n}^2 + \mathbf{u}_{n_{\mathcal{F}}}^{\mathrm{T}} \mathbf{S}_{\mathrm{R}}^{*} \mathbf{u}_{n_{\mathcal{F}}}\right) , \tag{5.37}$$

$$\mathrm{MSE}_{\mathrm{G}}^{*} = \sum_{n=1}^{N} d_n \left(\sigma_{w_n}^2 + \mathbf{u}_{n_{\mathcal{F}}}^{\mathrm{T}} \mathbf{S}_{\mathrm{R}}^{*} \mathbf{u}_{n_{\mathcal{F}}}\right) , \tag{5.38}$$

$$\mathrm{MSD}_{\mathrm{G}}^{*} = \mathrm{tr}\left\{\mathbf{S}_{\mathrm{R}}^{*}\right\} . \tag{5.39}$$

5.3 GSP Data-selective Estimation Algorithms

As a direct application of the recently developed field of graph signal processing is its use in distributed networks, which can be battery-powered, topics related to energy consumption in GSP procedures draw special attention because the adoption of power savings strategies might considerably increase the lifespan of an autonomous network. In particular, a promising idea is that the freshly proposed adaptive algorithms for graph signal estimation can be more efficiently implemented if we use them along a data selection approach similar to the ones presented in Section 3.1, where the adaptive algorithm only updates its signal estimation when it verifies that the current input data brings enough novelty to the current system estimate. Then, the overall computation complexity is reduced since the algorithm internal parameters are not updated at each iteration, which provides useful power savings for devices with strict energy consumption requirements.

Although the adoption of the data selection concepts from Section 3.1 along the GSP LMS, RLS and NLMS algorithms seems trivial, the main challenge one faces is related to the dimensionality of the error signal in the current graph signal application, which requires an alternative definition for assessing data novelty in the GSP scenario. This occurs because the update decision on traditional algorithms [34, 36] depends on the scalar error (2.3) and the GS estimation problem deals with the error vector $\mathbf{e}[k]$ in (4.21). Thus, in order to check data innovation we consider two approaches: the first one performs component-wise comparisons between the error vector $\mathbf{e}[k]$ and a threshold vector $\bar{\boldsymbol{\gamma}}_{\mathrm{DS}} \in \mathbb{R}_+^N$, while the second one compares a squared ℓ_2-norm error-based metric with a scalar $\bar{\gamma}_{\mathrm{DS}} \in \mathbb{R}_+$. Due to their characteristics, these strategies are called component-wise error constraint (CW-EC) and ℓ_2-norm error constraint (ℓ_2N-EC), respectively. These two different testing approaches are presented and further discussed in Subsections 5.3.1 and 5.3.2, respectively.

Finally, after testing the update condition, the proposed data-selection strategies either maintain their previous internal values, if the current error condition test returns true, or perform the usual algorithm update otherwise. This update approach for GS estimation is based on the data-selective adaptive scheme from Subsection 3.1.1, which is a straightforward method for obtaining DS versions of the adaptive GSP algorithms LMS, RLS, and NLMS by simply using the specific update expressions (4.23), (4.26), and (5.7), respectively. In the next subsections we detail the two proposed DS strategies for GS estimation and suggest practical choices for the thresholds that allow to control the mean update rate of the DS adaptive algorithms.

5.3.1 Component-Wise Error Constraint Strategy

The first update idea consists in defining a threshold vector $\bar{\boldsymbol{\gamma}}_{\mathrm{DS}} = [\bar{\gamma}_{\mathrm{DS}_1} \quad \bar{\gamma}_{\mathrm{DS}_2} \quad \dots \quad \bar{\gamma}_{\mathrm{DS}_N}]^{\mathrm{T}}$ such that the n^{th} error component $e_n[k]$ from (4.21) is compared to its respective bound $\bar{\gamma}_{\mathrm{DS}_n} \in \mathbb{R}_+$. If all absolute components $|e_n[k]|$ are smaller than their respective $\bar{\gamma}_{\mathrm{DS}_n}$, the strategy assumes the input data does not bring enough innovation to the current system. In other words, considering that function abs$(\cdot)$ performs a component-wise modulus operation on its argument vector, when

$$\mathrm{abs}(\mathbf{e}[k]) \preceq \bar{\boldsymbol{\gamma}}_{\mathrm{DS}} \tag{5.40}$$

is true, there is no algorithm update. Otherwise, if any $e_n[k]$ has a larger absolute value than its respective $\bar{\gamma}_{\mathrm{DS}_n}$, we evaluate the new estimate according to (4.23), (4.26), or (5.7), depending on the choice of the LMS, RLS, or NLMS algorithms.

Although the CW-EC data-selective constraint-vector $\bar{\boldsymbol{\gamma}}_{\mathrm{DS}}$ can be defined in many different ways, its choice influences the algorithm behavior in terms of the update probability/rate. In order to provide a fair estimate of the update rate $P_{\mathrm{up}} \in [0, 1]$ for the component-wise strategy, we consider that each $e_n[k]$ is modeled as a zero-mean Gaussian random variable (RV) with variance $\sigma^2_{e_n}$. Based on expressions (5.18), (5.31), and (5.37) for the NLMS, LMS, and RLS algorithms, respectively, we find the variances $\sigma^2_{e_n}$ for the particular algorithm and define $\bar{\boldsymbol{\gamma}}_{\mathrm{DS}}$ as

$$\bar{\boldsymbol{\gamma}}_{\mathrm{DS}} = \kappa \begin{bmatrix} \sigma_{e_1} & \sigma_{e_2} & \dots & \sigma_{e_N} \end{bmatrix}^{\mathrm{T}}, \tag{5.41}$$

in which the so-called update factor $\kappa \in \mathbb{R}_+$ is a design parameter included to fine tune the update rate.

The probability that all error entries will be in their respect intervals $[-\kappa\,\sigma_{e_n}, \kappa\,\sigma_{e_n}]$ is $\left[\mathrm{erf}(\kappa/\sqrt{2})\right]^{|\mathcal{S}|}$, since only $|\mathcal{S}|$ components of $\mathbf{e}[k]$ are non-zero, and the error function $\mathrm{erf}\big(\bar{\gamma}_{\mathrm{DS}_n}/(\sqrt{2}\sigma_{e_n})\big)$ describes the probability of $e_n[k]$ to fall in the interval $[-\bar{\gamma}_{\mathrm{DS}_n}, \bar{\gamma}_{\mathrm{DS}_n}]$ [76]. However, as the update rate is the complement of this value, the update probability P_{up} for the CW-EC DS estimation algorithm is

$$P_{\mathrm{up}} = 1 - \left[\mathrm{erf}\left(\frac{\kappa}{\sqrt{2}}\right)\right]^{|\mathcal{S}|}, \tag{5.42}$$

Alternatively, if the designer expects a P_{up} update rate, we find that

$$\kappa = \sqrt{2} \cdot \mathrm{erf}^{-1}\left(\sqrt[|\mathcal{S}|]{1 - P_{\mathrm{up}}}\right). \tag{5.43}$$

The proposed CW-EC strategy in Algorithm 6 adds $2|\mathcal{S}|$ FLOPs per iteration ($|\mathcal{S}|$ modulus operations and $|\mathcal{S}|$ comparisons) to the algorithm complexity due to

the test condition (5.40). However, it provides a considerable reduction of the overall complexity by avoiding unnecessary updates when condition (5.40) holds.

Algorithm 6 CW-EC data-selective strategy

1: Define update factor $\kappa \in \mathbb{R}_+^*$ via (5.43)
2: Evaluate $\bar{\boldsymbol{\gamma}}_{\mathrm{DS}} = \kappa \begin{bmatrix} \sigma_{e_1} & \sigma_{e_2} & \dots & \sigma_{e_N} \end{bmatrix}^{\mathrm{T}}$, where σ_{e_n} is given by (5.18), (5.31), or (5.37)
3: $k \leftarrow 0$
4: **while** (true) **do**
5: $\quad \mathbf{e}[k] = \mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[k] - \hat{\mathbf{x}}_{\mathrm{o}}[k])$
6: $\quad$ **if** ($\mathrm{abs}(\mathbf{e}[k]) \preceq \bar{\boldsymbol{\gamma}}_{\mathrm{DS}}$) **then**
7: $\quad\quad \hat{\mathbf{x}}_{\mathrm{o}}[k+1] = \hat{\mathbf{x}}_{\mathrm{o}}[k]$
8: $\quad$ **else**
9: $\quad\quad$ Find $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$ using (5.7), (4.23), or (4.26)
10: $\quad k \leftarrow k+1$
11: **end**

5.3.2 ℓ_2-Norm Error Constraint Strategy

An alternative to the CW-EC strategy consists in representing the instantaneous error vector $\mathbf{e}[k]$ by a single scalar value, which is directly compared to a scalar threshold $\bar{\gamma}_{\mathrm{DS}} \in \mathbb{R}_+$. In order to map $\mathbf{e}[k]$ into a scalar value we first define the normalized error vector $\bar{\mathbf{e}}[k] \in \mathbb{R}^N$ according to its individual components $\bar{e}_n$ described by

$$\bar{e}_n = \begin{cases} \dfrac{e_n[k]}{\sigma_{e_n}} & , \text{if } \sigma_{e_n} \neq 0\,, \\ 0 & , \text{otherwise}\,, \end{cases} \tag{5.44}$$

where σ_{e_n} comes from (5.18), (5.31), or (5.37). Then, we select the squared ℓ_2-norm $\|\bar{\mathbf{e}}[k]\|_2^2$ for performing the scalar mapping. The ℓ_2N-EC strategy consists in verifying if the condition

$$\|\bar{\mathbf{e}}[k]\|_2^2 \leq \bar{\gamma}_{\mathrm{DS}} \tag{5.45}$$

holds true, in which case there is no algorithm update.

By choosing an update expression based on either the NLMS, the LMS, or the RLS algorithms, we once again consider that each $e_n[k]$ is modeled as a zero-mean Gaussian RV with variance $\sigma_{e_n}^2$. As the square of a normal RV results in a chi-squared RV χ_1^2 with one-degree of freedom [77], then $\|\bar{\mathbf{e}}[k]\|_2^2$ is described by a $\chi_{|\mathcal{S}|}^2$ distribution, i.e., a chi-squared distribution with $|\mathcal{S}|$ degrees of freedom.

For an update factor $\kappa \in \mathbb{R}_+$, if we consider the threshold value

$$\bar{\gamma}_{\mathrm{DS}} = \kappa\,|\mathcal{S}|\,, \tag{5.46}$$

and remember the cumulative distribution function (CDF) of a chi-squared distribution with $|\mathcal{S}|$ degrees of freedom [77], the probability P_{up} for the ℓ_2N-EC strategy is estimated as

$$P_{\text{up}} = \frac{\Gamma_{\text{i}}(0.5\kappa|\mathcal{S}|)}{\Gamma(0.5|\mathcal{S}|)}\,, \tag{5.47}$$

where $\Gamma(\cdot)$ denotes the standard gamma function, and $\Gamma_{\text{i}}(0.5\kappa|\mathcal{S}|) = \int_{0.5\kappa|\mathcal{S}|}^{\infty} t^{0.5|\mathcal{S}|-1}\text{e}^{-t}\text{d}t$ is an upper incomplete gamma function. Alternatively, if the designer expects a P_{up} update rate, we find that

$$\kappa = \frac{2}{|\mathcal{S}|}\Gamma_{\text{i}}^{-1}\left(P_{\text{up}} \cdot \Gamma(0.5|\mathcal{S}|)\right)\,. \tag{5.48}$$

Finally, the proposed ℓ_2N-EC DS strategy is summarized in Algorithm 7.

Algorithm 7 ℓ_2N-EC data-selective strategy

Define update factor $\kappa \in \mathbb{R}_+^*$ via (5.48)
Evaluate $\bar{\gamma}_{\text{DS}} = \kappa\,|\mathcal{S}|$
$k \leftarrow 0$
while (true) **do**
  $\mathbf{e}[k] = \mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[k] - \hat{\mathbf{x}}_{\text{o}}[k])$
  Obtain $\bar{\mathbf{e}}[k]$, whose entries $\bar{e}_n[k]$ are given by (5.44)
  if ($\|\bar{\mathbf{e}}[k]\|_2^2 \leq \bar{\gamma}_{\text{DS}}$) **then**
    $\hat{\mathbf{x}}_{\text{o}}[k+1] = \hat{\mathbf{x}}_{\text{o}}[k]$
  else
    Find $\hat{\mathbf{x}}_{\text{o}}[k+1]$ using (5.7), (4.23), or (4.26)
  $k \leftarrow k+1$
end

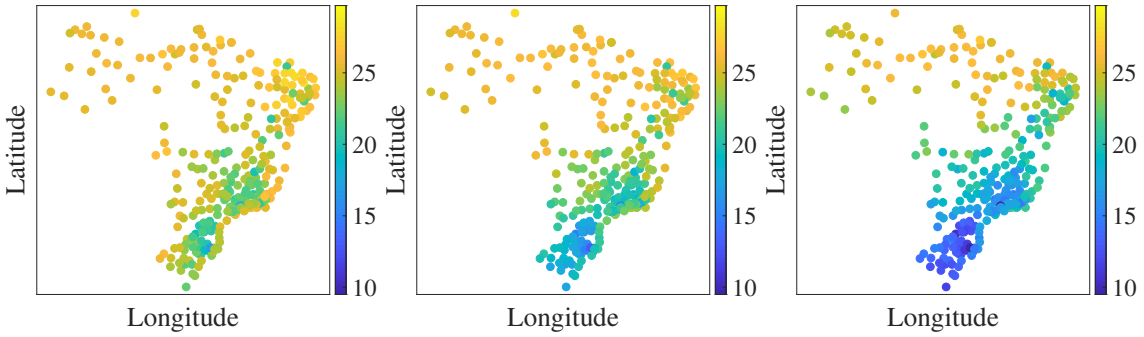
Chapter 6

GSP Database and Simulation Results

This chapter suggests a practical application of GSP and, based on this scenario, corroborates the contributions proposed in Chapter 5 by assessing their performance through a large number of numerical simulations. The simulations discussed in this chapter have been performed in a MATLAB environment and are available at the GitHub repository [42].

Initially, Section 6.1 introduces a GSP database obtained from a set of monthly average temperature measurements acquired from Brazilian weather stations [41]. The relevance of this section comes from the fact that it applies some GSP concepts and ideas discussed in Chapter 4, suggesting a practical procedure for GSP modeling that can be extended to different datasets with spatial correlation and providing the bandlimited GS scenario adopted in Section 6.2.

Relying on this real-world inspired scenario, Section 6.2 delves into the recent research branch that uses adaptive filtering-based algorithm for solving the problem of online estimating a bandlimited graph signal from sampled noisy measurements. The GSP LMS, RLS, and NLMS algorithms are compared in terms of convergence speed and computational complexity in Subsection 6.2.2, where one clearly notices the advantages offered by the NLMS-based algorithm proposed in Section 5.1. Furthermore, the accuracy of the stationary FoMs and update rate (when using the data-selective strategies from Section 5.3) predictions for the GSP LMS, RLS, and NLMS algorithms is demonstrated in Subsections 6.2.3 and 6.2.4.



(a) January (b) April (c) July

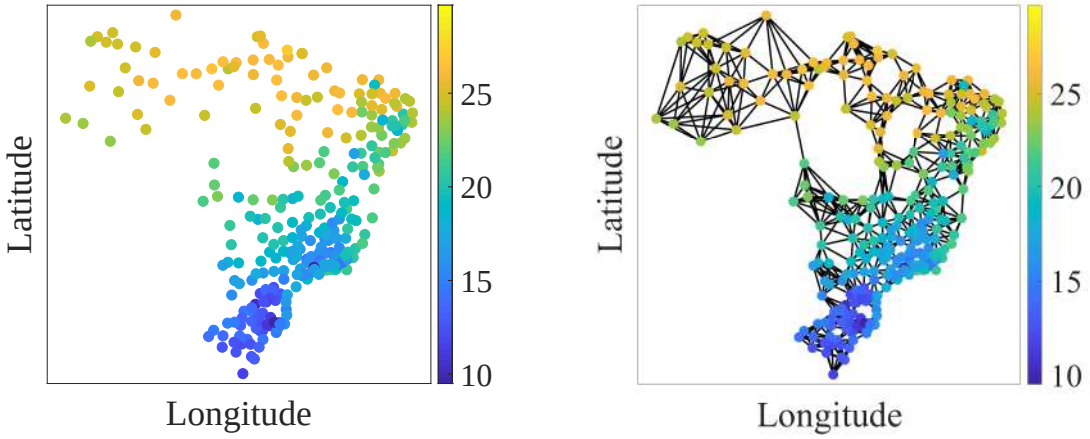
Figure 6.1: Simple graph signal representation of 1961-1990 monthly average temperatures (in °C) from Brazilian weather stations.

6.1 Approximating Temperature Measurements as a Bandlimited Graph Signal

Based on the promising idea of turning smooth data compiled across a geographic region into a graph signal and graph structure in order to exploit the recent GSP framework, this section shows that the temperature graph signal extracted from Brazilian weather stations is approximately bandlimited. In particular, the current section is useful for demonstrating some GSP concepts and ideas introduced in Chapter 4 and providing the bandlimited GS scenario considered in Section 6.2.

For data acquisition, two datasets are collected from the Instituto Nacional de Meteorologia (INMET) website [41]: the first one contains the latitude and longitude coordinates of active weather stations, while the second dataset presents a monthly average temperature recorded in some of these stations, during the 1961-1990 period. From these data we obtain a total of 299 nodes for the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, in which each of these vertices represents a weather station. Thus, a node v_n of the graph $\mathcal{G}$ and its signal value $x_n[k]$ are given, respectively, by the geographical coordinates and the average temperature of the associated weather station in a given month, as illustrated for the months of January, April, and July in Figure 6.1.

As we have no explicit connection between weather stations, we are free to design the set $\mathcal{E}$ and choose the weights $\{a_{ij}\}$, inferring the underlying graph structure. As discussed in Subsection 4.1.3, since the Brazilian weather stations in Figure 4.4 are irregularly distributed, for preventing a graph with both dense and sparsely populated regions we adopt an approach that constructs the graph edges by connecting a vertex v_n to, at least, its 8 closest neighbor nodes. However, the closest nodes identification depends on the definition of a distance metric among nodes. As each vertex is represented by its geographical coordinates, we consider that the distance between two nodes is given by the Haversine formula [78], which evaluates the great-



(a) GS with no edges.

(b) GS along connected graph structure.

Figure 6.2: July's 1961-1990 average temperatures (in °C) from Brazilian weather stations [41] represented as: (a) graph signal with no connections and (b) graph signal with edges determined by the closest-neighbor procedure.

circle distance between points using their latitude and longitude coordinates. For example, if one assumes two vertices v_i and v_j with latitudes φ_i^{lat} and φ_j^{lat}, and longitudes λ_i^{lon} and λ_j^{lon}, respectively, the Haversine distance $d_{\text{H}}(i,j)$ between these two nodes is given by

$$d_{\text{H}}(i,j) = 2\, r_{\text{E}}\, \arcsin\left(\sqrt{\sin^2\left(\frac{\varphi_i^{\text{lat}} - \varphi_j^{\text{lat}}}{2}\right) + \cos(\varphi_i^{\text{lat}})\cos(\varphi_j^{\text{lat}})\sin^2\left(\frac{\lambda_i^{\text{lon}} - \lambda_j^{\text{lon}}}{2}\right)}\right), \tag{6.1}$$

where r_{E} represents the approximate Earth radius (since the Earth is not a perfect sphere), taken here as $r_{\text{E}} = 6360$ km. Thus, based on this procedure we obtain the graph structure previously displayed in Figure 4.4. In particular, the original GS with no connections and its version considering the edges evaluated are illustrated in Figure 6.2b for the month of July.

As GSP techniques require the use of either the Laplacian $\mathbf{L}$ or adjacency matrix $\mathbf{A}$ introduced in Subsection 4.1.1, we need to define the edge weights $\{a_{ij}\}$ of $\mathbf{A}$, which can be seen as a similarity measure between neighboring vertices. Following the suggestion in Subsection 4.1.3, if there is an the edge connecting nodes v_i and v_j ($\widehat{v_i v_j} \in \mathcal{E}$), for evaluating the respective non-null weight a_{ij} we adopt the Gaussian kernel weighting function in (4.5), where in this case we replace the Euclidean distance d_{E} with the Haversine distance $d_{\text{H}}(v_i, v_j)$ between vertices v_i and v_j, as in (6.1). The kernel parameter θ is chosen as $2 \cdot 10^3$ after testing different values through numerical simulations and verifying that, for the graph signal considered,

this selection provides edge weights that result in a frequency representation with desired bandlimited characteristics.

A common assumption in GSP literature is that smooth signals on graphs present a bandlimited or approximately bandlimited frequency representation, in this case given by their low-frequency components or eigenvectors. As a graph signal $\mathbf{x}_w[k]$ obtained from temperature measurements across the country presents this smooth behavior (despite minor outlier points) at every instant k, we expect $\mathbf{x}_w[k]$ to be approximately bandlimited. However, before doing so we need to decide how many frequency components are necessary for representing the graph signal with an acceptable deviation error.

As this task consists in a signal compression problem, we take a similar procedure to [7] and evaluate the average reconstruction error (ARE) $\|\mathbf{x}_\mathrm{o} - \bar{\mathbf{x}}_\mathrm{o}^P\|_2/\|\mathbf{x}_\mathrm{o}\|_2$ for different estimates $\bar{\mathbf{x}}_\mathrm{o}^P$ using only the P-largest frequency components of the original bandlimited graph signal $\mathbf{x}_\mathrm{o}$, taken as the signal from the July dataset depicted in Figure 6.2b. These frequency components are respective to the eigendecomposition of the adjacency matrix $\mathbf{A}$, i.e., we follow the GSP_A approach from Section 4.2. Then, the signal $\bar{\mathbf{x}}_\mathrm{o}^P$ is obtained by sorting the absolute values of the frequency-domain signal $\mathbf{s}$ obtained from (4.6) and selecting the indices p of the P-largest components $|s_n|$ to form the auxiliar set $\mathcal{F}_P$. Based on these indices $p \in \mathcal{F}_\mathrm{P} \subseteq \mathcal{N}$, we pick the p^{th} eigenvector of $\mathbf{U}$ and the p^{th} frequency component of $\mathbf{s}$ to define $\mathbf{U}_{\mathcal{F}_P}$ and $\mathbf{s}_{\mathcal{F}_P}$. Then, the estimate $\bar{\mathbf{x}}_\mathrm{o}^P$ using P components is given by $\mathbf{U}_{\mathcal{F}_P}\mathbf{s}_{\mathcal{F}_P}$.

Following this compression procedure, we compute the ARE percentage for different values of used components P and display the error results when using from 50 up to 250 frequency components in Figure 6.3. Assuming that a deviation error of 2.5% is acceptable in the current application, we approximate the original graph signal by its $P = 200$ largest frequency components. From this assumption we can define the bandlimited set $\mathcal{F} = \mathcal{F}_\mathcal{P}$, where $|\mathcal{F}| = P$.

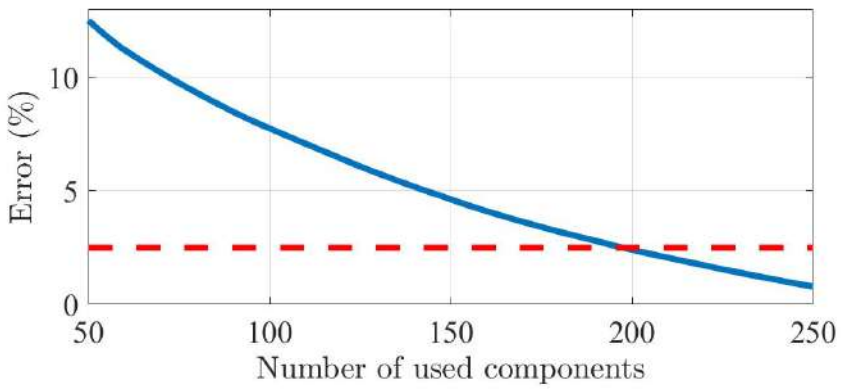


Figure 6.3: Percentage of reconstruction error when the original signal is compressed using P frequency components.

Based on this approximately $\mathcal{F}$-ssparse signal $\mathbf{x}_{\mathrm{o}}$, we need to take a practical project decision and select both the amount $|\mathcal{S}|$ and which vertices $v_n \in \mathcal{V}$ of the graph signal should be sampled. As stated in [12], increasing the number of samples in $\mathcal{S}$ always decreases the $\mathrm{MSD}_{\mathrm{G}}$ in (4.17). However, as we also want to reduce the amount of nodes to be measured, we consider that $|\mathcal{S}| = 210$ provides a reasonable trade-off and then find the *sampling set* $\mathcal{S}$ by using Algorithm 2, with $M = |\mathcal{S}| = 210$. Then, at this point we obtain the reference bandlimited GS $\mathbf{x}_{\mathrm{o}}$, the frequency set $\mathcal{F}$, the sampling set $\mathcal{S}$, and their respective matrices $\mathbf{U}_{\mathcal{F}}$ and $\mathbf{D}_{\mathcal{S}}$ to be used in the next subsections simulations. For practical purposes, $\mathbf{x}_{\mathrm{o}}$, $\mathbf{U}_{\mathcal{F}}$ and $\mathbf{D}_{\mathcal{S}}$ are explicitly presented in [42].

6.2 Adaptive Algorithms for GS Estimation

For assessing the theoretical predictions from Sections 5.1, 5.2 and 5.3, this part of the work delves into the adaptive GS estimation ideas and evaluates the performance of the GSP NLMS, LMS, and RLS adaptive algorithms when using the approximate bandlimited graph signal $\mathbf{x}_{\mathrm{o}}$ obtained from Section 6.1. Then, at this point one has the constant matrices $\mathbf{U}_{\mathcal{F}}$ and $\mathbf{D}_{\mathcal{S}}$, and the reference bandlimited graph signal $\mathbf{x}_{\mathrm{o}} = \mathbf{U}_{\mathcal{F}}\mathbf{s}_{\mathcal{F}}$. To complete the description of the simulation environment used in this adaptive setup, the last point one needs to introduce is the noise scenarios in which the sampled graph signals are exposed.

6.2.1 Noise Scenarios

In order to evaluate the adaptive GSP algorithms in different noise scenarios, one adopts a generic covariance matrix $\mathbf{C}_w$ represented as

$$\mathbf{C}_w = \mathrm{diag}\big(\sigma^2_{w_a}\,\mathbf{1} + \sigma^2_{w_b}\,\mathbf{r}_w\big)\,, \tag{6.2}$$

where $\mathbf{1} \in \mathbb{R}^N$ is a vector with all components equal to 1, $\mathbf{r}_w \in \mathbb{R}^N$ is a realization of a random vector whose entries follow a uniform distribution between $[0, 1]$, and $\sigma^2_{w_a}, \sigma^2_{w_b} \in \mathbb{R}_+$ are variances that scale the elements of $\mathbf{1}$ and $\mathbf{r}_w$, respectively.

Although the theoretical predictions provided are valid for any symmetric $\mathbf{C}_w$, the presented simulation environment focuses on diagonal matrices because this work considers that the sensor positions are distant from each other, then, the influence of correlated noise is not expected on more than one graph vertex measurement. For $\mathbf{C}_w$ in (6.2), $\sigma^2_{w_a}$ represents the noise variance commonly estimated in all nodes, which might be caused by the same type of sensor device being used in all locations, while $\sigma^2_{w_b}$ accounts for the noise variance differences among nodes.

Thus, the noise signal $\boldsymbol{w}[k]$ in (4.16) employed for each simulation presented in the next subsections is defined as zero-mean Gaussian according to three scenarios:

(i) $\sigma_{w_a}^2 = 0.001$ and $\sigma_{w_b}^2 = 0.000$;

(ii) $\sigma_{w_a}^2 = 0.010$ and $\sigma_{w_b}^2 = 0.000$; and

(iii) $\sigma_{w_a}^2 = 0.005$ and $\sigma_{w_b}^2 = 0.010$.

6.2.2 Convergence Speed and Complexity Comparison

Based on the noise scenarios (i) and (iii), numerical simulations are performed to analyze the overall performance of the proposed GSP NLMS algorithm in comparison to the LMS [15] and RLS [18] strategies. For each noise scenario the three adaptive GSP algorithms are applied, with respective convergence/forgetting factors adjusted in order to provide a similar $\text{MSD}_{\text{G}}^{*}$. At each simulation run we evaluate 5000 iterations, where the reference GS $\mathbf{x}_{\text{o}}$ is scaled by a 1.2 factor at $k = 2500$ to observe the algorithms' tracking abilities. Then, it is assumed that: the algorithm has converged after reaching $1.025 \cdot \text{MSD}_{\text{G}}^{*}$ for the first time, the steady-state FoMs are computed using the last 1000 iterations of each run, and the update time uses the "*tic/toc*" MATLAB functions to provide the reader with a rough idea of how long it takes to compute $\hat{\mathbf{x}}_{\text{o}}[k+1]$ for each algorithm simulated. Based on the average values of a 1000-run ensemble, the simulations numerical results are summarized in Table 6.1.

From Table 6.1 one observes that the GSP NLMS algorithm converges considerably (more than 10 times) faster than the LMS algorithm, but slightly (about twice) slower than the RLS algorithm. This convergence speed comparison is made clear by the $\text{MSD}_{\text{G}}[k]$ plots in Figures 6.4 and 6.5, where we only compare the NLMS and LMS methods in Figure 6.4 and display the three algorithms in Figure 6.5. A particular point about Figure 6.5 is that the transition at $k = 2500$ indicates that the NLMS algorithm behaves like the RLS for large k, as pointed out in Subsection 5.1.5. Another conclusion from Table 6.1 is that the computation complexity for performing the NLMS update is comparable to the one for the LMS, being much smaller than the required for the RLS algorithm.[1]

6.2.3 Steady-State FoM Predictions

Next, we investigate the accuracy of the steady-state $\text{MSE}_{\text{G}}^{*}$ and $\text{MSD}_{\text{G}}^{*}$ predicted for the NLMS algorithm in Subsection 5.1.4 and for the LMS and RLS algorithms

[1] Although these complexity results cannot be directly compared to the FLOPs estimation in Subsection 5.1.3, they corroborate the idea from Subsection 5.1.3 that the GSP NLMS complexity is similar to the one for the GSP LMS algorithm.

Table 6.1: MSD^*_{G}, iterations until convergence and time for computing $\hat{\mathbf{x}}_{\text{o}}[k+1]$ for adaptive GSP simulation scenarios

Entry	Simulation Setup			Simulation Results		
	Alg.	Factor	$\mathbf{C}_w$	MSD^*_{G}	Converg.	Upd. Time
(L.I)	LMS	0.280	(i)	0.0313	1727 iter.	32 μs
(R.I)	RLS	0.930	(i)	0.0313	65 iter.	1952 μs
(N.I)	NLMS	0.070	(i)	0.0314	137 iter.	31 μs
(L.II)	LMS	0.280	(iii)	0.3121	1479 iter.	31 μs
(R.II)	RLS	0.930	(iii)	0.3107	58 iter.	1916 μs
(N.II)	NLMS	0.070	(iii)	0.3115	114 iter.	31 μs
(L.III)	LMS	0.721	(iii)	1.0034	533 iter.	32 μs
(R.III)	RLS	0.792	(iii)	0.9922	18 iter.	1941 μs
(N.III)	NLMS	0.208	(iii)	0.9961	34 iter.	32 μs

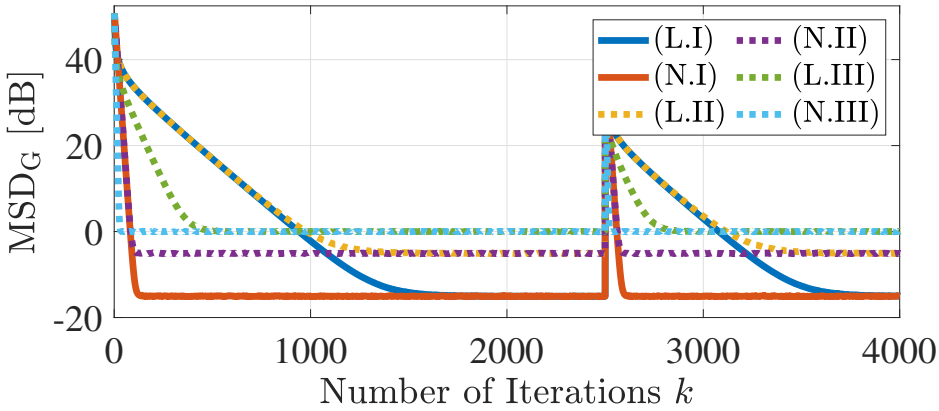


Figure 6.4: $\text{MSD}_{\text{G}}[k]$ behavior when applying the GSP LMS and NLMS algorithms to different simulation scenarios described in Table 6.1.

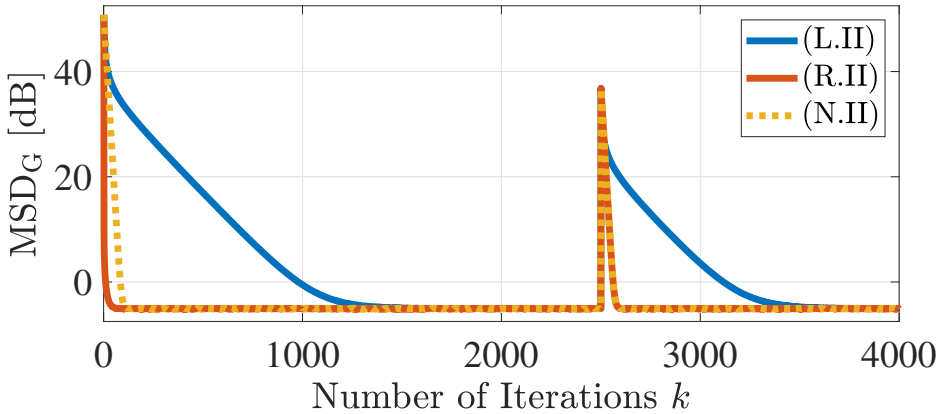


Figure 6.5: $\text{MSD}_{\text{G}}[k]$ behavior when applying the GSP LMS, RLS, and NLMS algorithms to the simulation scenario (ii).

in Section 5.2. By adopting the noise scenarios (ii) and (iii) described in Subsection 6.2.1, we use different convergence/forgetting factors (μ_{N}, μ_{L}, and β_{R}) to assess

Table 6.2: Theoretical and experimental $\text{MSE}_{\text{G}}^{*}$ and $\text{MSD}_{\text{G}}^{*}$, and their respective REs, for the GSP NLMS, LMS, and RLS algorithms in noise scenario (ii)

Algor.	Factor	$\text{MSE}_{\text{G}}^{*}$			$\text{MSD}_{\text{G}}^{*}$		
		Theory	Simul.	RE[%]	Theory	Simul.	RE[%]
NLMS	0.05	2.1513	2.1512	0.005	0.2217	0.2212	0.226
	0.10	2.2053	2.2052	0.005	0.4550	0.4549	0.022
	0.25	2.3857	2.3855	0.008	1.2350	1.2358	-0.065
	0.50	2.7667	2.7665	0.007	2.8817	2.8822	-0.017
LMS	0.20	2.2600	2.2599	0.004	0.2160	0.2159	0.046
	0.50	2.5717	2.5711	0.023	0.6179	0.6179	0.000
	1.00	3.4617	3.4638	-0.061	1.6803	1.6815	-0.071
RLS	0.95	2.1513	2.1516	-0.014	0.2217	0.2217	0.000
	0.90	2.2053	2.2050	0.014	0.4550	0.4551	-0.022
	0.75	2.3857	2.3857	0.000	1.2350	1.2362	-0.097

the theoretical predictions in diverse conditions. At each simulation run we evaluate 3000, 12000, and 1500 iterations for the NLMS, LMS, and RLS algorithms, respectively, where the last 1000 iterations of each run are assumed to be part of the steady state. By taking an average of the $\text{MSE}_{\text{G}}[k]$ and $\text{MSD}_{\text{G}}[k]$ measurements at steady state for an ensemble of 1000 runs, the obtained experimental results are presented in Tables 6.2 and 6.3 for noise scenarios (ii) and (iii), respectively. These results are compared to the $\text{MSE}_{\text{G}}^{*}$ and $\text{MSD}_{\text{G}}^{*}$ theoretical predictions presented in: (5.19) and (5.20) for the NLMS algorithm; (5.32) and (5.33) for the LMS algorithm; and (5.38) and (5.39) for the RLS algorithm. Additionally, for clarifying how close the specified theoretical and simulated FoMs are, in Tables 6.2 and 6.3 it is also included a relative error (RE) metric computed as

$$\text{Relative error (RE)} = \frac{\text{Theory value} - \text{Simul. result}}{\text{Simul. result}}\,. \tag{6.3}$$

According to Tables 6.2 and 6.3, one verifies that the $\text{MSE}_{\text{G}}^{*}$ and $\text{MSD}_{\text{G}}^{*}$ predictions provided for the NLMS, LMS, and RLS algorithms are very accurate across all different simulation scenarios. In particular, all results yield an RE smaller than 0.25% with respect to their theoretical estimates. Although both the $\text{MSE}_{\text{G}}^{*}$ and $\text{MSD}_{\text{G}}^{*}$ predictions have been obtained in this work for all three adaptive GSP algorithms, it is worth mentioning that the $\text{MSD}_{\text{G}}^{*}$ for the LMS and RLS algorithms has been previously presented in [19]. However, the analysis for the LMS algorithm requires μ_{L} to be small and it presents an approximation that provides worse estimates of $\text{MSD}_{\text{G}}^{*}$ as μ_{L} increases. On the other hand, from Tables 6.2 and 6.3 one

Table 6.3: Theoretical and experimental $\text{MSE}_{\text{G}}^{*}$ and $\text{MSD}_{\text{G}}^{*}$, and their respective REs, for the GSP NLMS, LMS, and RLS algorithms in noise scenario (iii)

Algor.	Factor	$\text{MSE}_{\text{G}}^{*}$			$\text{MSD}_{\text{G}}^{*}$		
		Theory	Simul.	RE[%]	Theory	Simul.	RE[%]
NLMS	0.05	2.1464	2.1460	0.019	0.2202	0.2200	0.091
	0.10	2.2003	2.2004	-0.005	0.4520	0.4522	-0.044
	0.25	2.3804	2.3801	0.013	1.2268	1.2264	0.033
	0.50	2.7607	2.7605	0.007	2.8626	2.8623	0.010
LMS	0.20	2.2600	2.2598	0.009	0.2160	0.2161	-0.046
	0.50	2.5673	2.5674	-0.004	0.6171	0.6171	0.000
	1.00	3.4585	3.4595	-0.029	1.6793	1.6804	-0.065
RLS	0.95	2.1513	2.1512	0.005	0.2217	0.2217	0.000
	0.90	2.1999	2.1998	0.005	0.4501	0.4501	0.000
	0.75	2.3794	2.3795	-0.004	1.2216	1.2221	-0.041

concludes that the accurate predictions for the LMS algorithm using (5.33) do not degrade with large μ_{L} factors.

6.2.4 Update Rate Steady-State Predictions

Finally, we perform a few numerical simulations to analyze the behavior of the CW-EC and ℓ_2N-EC data-selective strategies from Subsections 5.3.1 and 5.3.2, respectively, assessing the accuracy of update rate expressions (5.42) and (5.47). For simplicity, the CW-EC and ℓ_2N-EC strategies are referred to as the DS schemes (I) and (II), respectively. By using the noise scenarios (ii) and (iii) described in Subsection 6.2.1, different values of update factor κ are used for each DS scheme. For each of these environments we apply either the GSP NLMS, LMS, or RLS as the adaptive algorithm with factors $\mu_{\text{N}} = 0.1$, $\mu_{\text{L}} = 0.5$, and $\beta_{\text{R}} = 0.9$, in a total of 2500, 10000, and 2000 iterations for run, respectively. To compute P_{up} only the last 1000 iterations of each run are considered, and the update rate is evaluated by taking the average across an ensemble with 1000 runs. The scenario descriptions, theoretical predictions, and experimental results for the DS adaptive GSP strategies are shown in Tables 6.4 and 6.5.

Based on these tables, one concludes that the estimates (5.42) and (5.47) provide fair predictions about the update probability, where the largest RE evaluated is less than 2.5%. For both DS strategies, it is observed that a κ increase implies in a reduction of P_{up}, which is an interesting feature since it enhances the overall computational complexity reduction. However, the trade-off for increasing κ is a reduction in the algorithm convergence speed, as can be noted in the MSD_{G} behavior

Table 6.4: Stationary P_{up} and RE for the adaptive GSP algorithms using the CW-EC and ℓ_2N-EC data-selective (DS) strategies in noise scenario (ii)

DS	κ	Theory	NLMS		LMS		RLS	
		P_{up}[%]	P_{up}[%]	RE[%]	P_{up}[%]	RE[%]	P_{up}[%]	RE[%]
CW-EC	3.00	43.319	43.314	0.012	43.300	0.044	43.325	-0.014
	3.50	9.310	9.322	-0.129	9.339	-0.311	9.319	-0.097
	3.75	3.646	3.658	-0.328	3.640	0.165	3.635	0.303
ℓ_2N-EC	1.00	48.702	48.599	0.212	48.781	-0.162	48.744	-0.086
	1.10	15.276	15.301	-0.163	15.375	-0.644	15.214	0.408
	1.15	6.701	6.682	0.284	6.841	-2.046	6.724	-0.342

Table 6.5: Stationary P_{up} and RE for the adaptive GSP algorithms using the CW-EC and ℓ_2N-EC data-selective (DS) strategies in noise scenario (iii)

DS	κ	Theory	NLMS		LMS		RLS	
		P_{up}[%]	P_{up}[%]	RE[%]	P_{up}[%]	RE[%]	P_{up}[%]	RE[%]
CW-EC	3.00	43.319	43.322	-0.007	43.343	-0.055	43.382	-0.145
	3.50	9.310	9.283	0.291	9.244	0.714	9.272	0.410
	3.75	3.646	3.647	-0.027	3.650	-0.110	3.678	-0.870
ℓ_2N-EC	1.00	48.702	48.682	0.041	48.645	0.117	48.718	-0.033
	1.10	15.276	15.317	-0.268	15.356	-0.521	15.296	-0.131
	1.15	6.701	6.730	-0.431	6.791	-1.325	6.733	-0.475

for the CW-EC and ℓ_2N-EC strategies in Figures 6.6 and 6.7, respectively. Therefore, as Tables 6.4 and 6.5, and Figures 6.6 and 6.7 indicate, one may reduce the overall complexity of the adaptive GSP algorithm by increasing the update factor of the DS strategy, at the cost of slowing down its convergence speed.

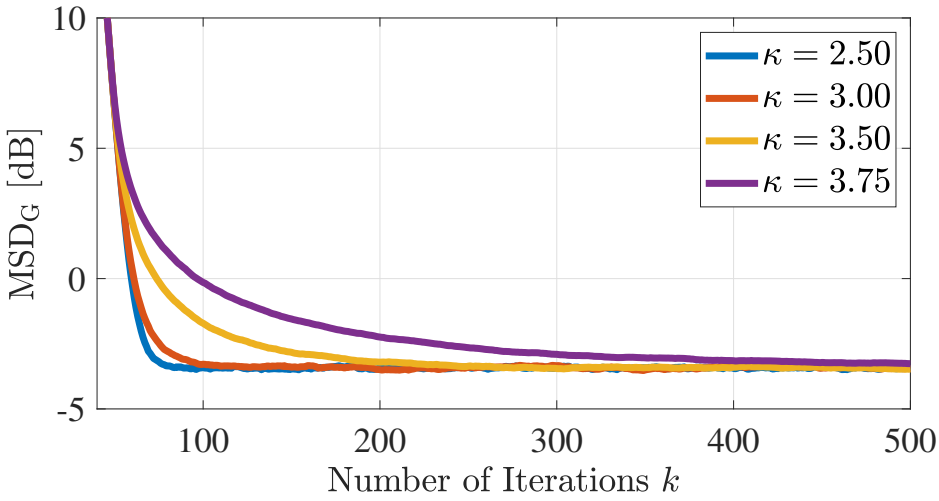


Figure 6.6: $\mathrm{MSD}_{\mathrm{G}}[k]$ of the NLMS algorithm when using different factors κ for the CW-EC DS strategy.

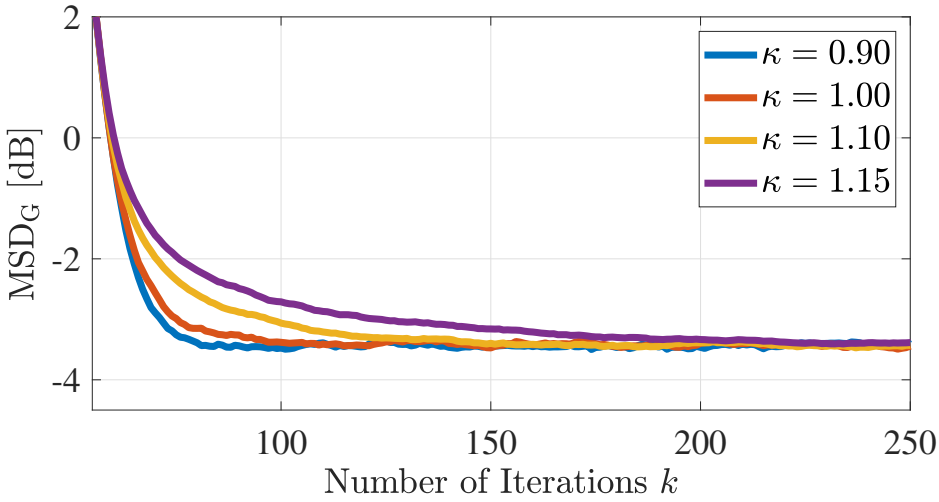


Figure 6.7: $\mathrm{MSD}_{\mathrm{G}}[k]$ of the NLMS algorithm when using different factors κ for the ℓ_2N-EC DS strategy.

Chapter 7

Conclusion and Future Works

7.1 Concluding Remarks

This dissertation extended some current adaptive filtering algorithms for handling the problem of online estimation of bandlimited graph signals from sampled noisy measurements. Considering that a recent line of research has recast the traditional LMS and RLS adaptive filtering algorithms into the GSP context, this work further explored this idea and proposed the respective NLMS algorithm and data-selective strategies for the same estimation scenario. The GSP NLMS algorithm has been designed such that it improves the convergence speed of the GSP LMS procedure, while requiring noticeable computational savings in comparison to the GSP RLS method. In particular, it has been verified that the GSP NLMS algorithm demands the same computational complexity of the GSP LMS. Moreover, we proposed a steady-state analysis of the NLMS method that provides closed-form expressions for evaluating important figures of merit that describe the algorithm behavior. This analysis has been extended to cover the previously suggested GSP LMS and RLS algorithms, complementing previous studies in the literature. At last, this dissertation proposed two alternative data-selective strategies to be implemented along the GSP adaptive algorithms in order to reduce the overall computational complexity and yield power savings. By adopting some particular choices of internal parameters, it has been demonstrated through a steady-state analysis that these data-selective strategies produce convenient formulas for obtaining specific update rates when using either the GSP LMS, the GSP RLS, or the GSP NLMS algorithms.

Along the way to describe the aforementioned contributions, this dissertation reviewed some traditional adaptive filtering concepts and algorithms, including the motivation and derivation of the LMS, RLS, NLMS, AP, and proportionate AP algorithms. Inspired by the appeal of reducing the computational complexity of adaptive procedures, the data selection idea has been introduced by adopting two different

approaches: the simpler data-selective strategy, later used along the adaptive GSP algorithms, and the set-membership approach. In particular, this work revisited a set-membership algorithm named SM-PAPA and presented some contributions for improving its practical performance. Additionally, this dissertation stated the motivation behind GSP, defining its basic elements, describing some framework tools, and discussing the extended signal processing concepts more relevant to the scope of this work. Focusing on bandlimited graph signals, the problem of online recovering the original signal from sampled noisy measurements has been stated and the recently proposed methods based on the LMS and RLS algorithms have been described. Then, after reviewing the most important topics on both traditional adaptive filtering and graph signal processing, this dissertation built on previous works and proposed the main theoretical contributions of this work, detailed in the last paragraph.

Simulation results indicated that the proposed GSP NLMS algorithm is indeed much faster than, while being approximately as complex as, the original GSP LMS method, which suggests an interesting alternative to the GSP RLS algorithm if one wishes to employ an adaptive GSP procedure that yields a reduced convergence time with low complexity. Moreover, the numerical results also demonstrated that both the steady-state metrics estimates in the analyses for the GSP LMS, RLS, and NLMS and the update probability predicted for the data-selective techniques are very accurate. This outcome strongly corroborated the complete analysis proposed and the expressions obtained, highlighting the analytical importance of the contributions provided in this dissertation. At last, for the data-selective strategies it has been noticed that the overall computational complexity can be reduced by increasing a simple internal parameter, but the intrinsic trade-off of this operation is an increase in the algorithm convergence time.

It is worth mentioning that the implementation of this work's proposed techniques and assessed simulation scenarios have been made available at [42].

7.2 Future Research Directions

As future works, there are two clear research directions: deriving alternative methods based on traditional adaptive filtering algorithms and implementing the data selection idea by following a different data selection strategy, such as the set-membership approach. The motivation supporting the first suggestion comes from the desire of exploring particular features in applications, which is the same inspiration for obtaining new adaptive filtering algorithms and has the potential of providing improvements in certain figures of merit. On the other hand, the first steps on the second clear research direction are suggested in this work (Subsection 5.1.1) for deriving an

SM-NLMS algorithm, that might outperform the data-selective approach considered in this work for implementing the data selection concept. Moreover, another data selection approach that can be adopted for generalizing the GSP RLS algorithm is the RLS BEACON in [35].

Alternatively, other possible non-trivial contributions involve improving the GSP NLMS algorithm by considering time-varying graph structures and sampling sets, and deriving distributed implementations. Furthermore, one can explore new real-world scenarios that might benefit from the use of the GSP framework and investigate the impacts of using the online adaptive estimation in comparison to traditional reconstruction techniques.

Bibliography

[1] DINIZ, P. S. R., DA SILVA, E. A. B., NETTO, S. L. *Digital signal processing: system analysis and design*. Cambridge University Press, 2010. doi: 10.1017/CBO9780511781667.

[2] CHEN, Y., KAR, S., MOURA, J. M. F. "The internet of things: secure distributed inference", *IEEE Signal Processing Magazine*, v. 35, n. 5, pp. 64–75, Sep. 2018. ISSN: 1053-5888. doi: 10.1109/MSP.2018.2842097.

[3] SANDRYHAILA, A., MOURA, J. M. F. "Big data analysis with signal processing on graphs: representation and processing of massive data sets with irregular structure", *IEEE Signal Processing Magazine*, v. 31, n. 5, pp. 80–90, Sept 2014. ISSN: 1053-5888. doi: 10.1109/MSP.2014.2329213.

[4] ORTEGA, A., FROSSARD, P., KOVAČEVIĆ, J., et al. "Graph signal processing: overview, challenges, and applications", *Proceedings of the IEEE*, v. 106, n. 5, pp. 808–828, May 2018. ISSN: 0018-9219. doi: 10.1109/JPROC.2018.2820126.

[5] BONDY, J. A., MURTY, U. S. R. *Graph theory with applications*. Macmillan, 1976.

[6] SHUMAN, D. I., NARANG, S. K., FROSSARD, P., et al. "The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains", *IEEE Signal Processing Magazine*, v. 30, n. 3, pp. 83–98, May 2013. ISSN: 1053-5888. doi: 10.1109/MSP.2012.2235192.

[7] SANDRYHAILA, A., MOURA, J. M. F. "Discrete signal processing on graphs", *IEEE Transactions on Signal Processing*, v. 61, n. 7, pp. 1644–1656, April 2013. ISSN: 1053-587X. doi: 10.1109/TSP.2013.2238935.

[8] SANDRYHAILA, A., MOURA, J. M. F. "Discrete signal processing on graphs: frequency analysis", *IEEE Transactions on Signal Processing*, v. 62, n. 12, pp. 3042–3054, June 2014. ISSN: 1053-587X. doi: 10.1109/TSP.2014.2321121.

[9] ANIS, A., GADDE, A., ORTEGA, A. "Towards a sampling theorem for signals on arbitrary graphs". In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3864–3868, May 2014. doi: 10.1109/ICASSP.2014.6854325.

[10] CHEN, S., SANDRYHAILA, A., KOVAČEVIĆ, J. "Sampling theory for graph signals". In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3392–3396, April 2015. doi: 10.1109/ICASSP.2015.7178600.

[11] CHEN, S., VARMA, R., SANDRYHAILA, A., et al. "Discrete signal processing on graphs: sampling theory", *IEEE Transactions on Signal Processing*, v. 63, n. 24, pp. 6510–6523, Dec 2015. ISSN: 1053-587X. doi: 10.1109/TSP.2015.2469645.

[12] CHAMON, L. F. O., RIBEIRO, A. "Greedy sampling of graph signals", *IEEE Transactions on Signal Processing*, v. 66, n. 1, pp. 34–47, Jan 2018. ISSN: 1053-587X. doi: 10.1109/TSP.2017.2755586.

[13] CHEN, S., SANDRYHAILA, A., MOURA, J. M. F., et al. "Signal recovery on graphs: variation minimization", *IEEE Transactions on Signal Processing*, v. 63, n. 17, pp. 4609–4624, Sep. 2015. ISSN: 1053-587X. doi: 10.1109/TSP.2015.2441042.

[14] CHEN, S., VARMA, R., SINGH, A., et al. "Signal recovery on graphs: fundamental limits of sampling strategies", *IEEE Transactions on Signal and Information Processing over Networks*, v. 2, n. 4, pp. 539–554, Dec 2016. ISSN: 2373-776X. doi: 10.1109/TSIPN.2016.2614903.

[15] LORENZO, P. D., BARBAROSSA, S., BANELLI, P., et al. "Adaptive least mean squares estimation of graph signals", *IEEE Transactions on Signal and Information Processing over Networks*, v. 2, n. 4, pp. 555–568, Dec 2016. ISSN: 2373-776X. doi: 10.1109/TSIPN.2016.2613687.

[16] LORENZO, P. D., BANELLI, P., BARBAROSSA, S., et al. "Distributed adaptive learning of graph signals", *IEEE Transactions on Signal Processing*, v. 65, n. 16, pp. 4193–4208, Aug 2017. ISSN: 1053-587X. doi: 10.1109/TSP.2017.2708035.

[17] LORENZO, P. D., BANELLI, P., BARBAROSSA, S. "Optimal sampling strategies for adaptive learning of graph signals". In: *2017 25th European Signal Processing Conference (EUSIPCO)*, pp. 1684–1688, Aug 2017. doi: 10.23919/EUSIPCO.2017.8081496.

[18] LORENZO, P. D., ISUFI, E., BANELLI, P., et al. "Distributed recursive least squares strategies for adaptive reconstruction of graph signals". In: *2017 25th European Signal Processing Conference (EUSIPCO)*, pp. 2289–2293, Aug 2017. doi: 10.23919/EUSIPCO.2017.8081618.

[19] LORENZO, P. D., BANELLI, P., ISUFI, E., et al. "Adaptive graph signal processing: algorithms and optimal sampling strategies", *IEEE Transactions on Signal Processing*, v. 66, n. 13, pp. 3584–3598, July 2018. ISSN: 1053-587X. doi: 10.1109/TSP.2018.2835384.

[20] LORENZO, P., BARBAROSSA, S., BANELLI, P. "Chapter 9 - Sampling and recovery of graph signals". In: Djurić, P. M., Richard, C. (Eds.), *Cooperative and graph signal processing*, Academic Press, pp. 261 – 282, 2018. ISBN: 978-0-12-813677-5. doi: https://doi.org/10.1016/B978-0-12-813677-5.00009-2. Disponível em: <`http://www.sciencedirect.com/science/article/pii/B9780128136775000092`>.

[21] LORENZO, P. D., CECI, E. "Online recovery of time-varying signals defined over dynamic graphs". In: *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 131–135, Sep. 2018. doi: 10.23919/EUSIPCO.2018.8553473.

[22] MOHAN, D. M., ASIF, M. T., MITROVIC, N., et al. "Wavelets on graphs with application to transportation networks". In: *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 1707–1712, Oct 2014. doi: 10.1109/ITSC.2014.6957939.

[23] HUANG, W., GOLDSBERRY, L., WYMBS, N. F., et al. "Graph frequency analysis of brain signals", *IEEE Journal of Selected Topics in Signal Processing*, v. 10, n. 7, pp. 1189–1203, Oct 2016. ISSN: 1932-4553. doi: 10.1109/JSTSP.2016.2600859.

[24] FRACASTORO, G., THANOU, D., FROSSARD, P. "Graph transform learning for image compression". In: *2016 Picture Coding Symposium (PCS)*, pp. 1–5, Dec 2016. doi: 10.1109/PCS.2016.7906368.

[25] THANOU, D., DONG, X., KRESSNER, D., et al. "Learning heat diffusion graphs", *IEEE Transactions on Signal and Information Processing over Networks*, v. 3, n. 3, pp. 484–499, Sep. 2017. ISSN: 2373-776X. doi: 10.1109/TSIPN.2017.2731164.

[26] WIDROW, B., HOFF, M. E. "Adaptive switching circuits", *WESCOM Conv. Rec.*, , n. pt 4, pp. 96–140, Aug 1960. ISSN: 0018-9219. doi: 10.1109/PROC.1976.10286.

[27] WIDROW, B., MCCOOL, J. M., LARIMORE, M. G., et al. "Stationary and nonstationary learning characteristics of the LMS adaptive filter ", *Proceedings of the IEEE*, v. 64, n. 8, pp. 1151–1162, Aug 1976. ISSN: 0018-9219. doi: 10.1109/PROC.1976.10286.

[28] SAYED, A. H. *Adaptive filters*. John Wiley & Sons, 2011.

[29] DINIZ, P. S. R. *Adaptive filtering: algorithms and practical implementation*. Springer, 2013.

[30] ELEFTHERIOU, E., FALCONER, D. "Tracking properties and steady-state performance of RLS adaptive filter algorithms", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 34, n. 5, pp. 1097–1110, October 1986. ISSN: 0096-3518. doi: 10.1109/TASSP.1986.1164950.

[31] SPELTA, M. J. M., MARTINS, W. A. "Online temperature estimation using graph signals". In: *XXXVI Simpósio Brasileiro de Telecomunicações e Processamento de Sinais - SBrT2018*, pp. 154–158, 2018.

[32] NAGUMO, J., NODA, A. "A learning method for system identification", *IEEE Transactions on Automatic Control*, v. 12, n. 3, pp. 282–287, June 1967. ISSN: 0018-9286. doi: 10.1109/TAC.1967.1098599.

[33] SLOCK, D. T. M. "On the convergence behavior of the LMS and the normalized LMS algorithms ", *IEEE Transactions on Signal Processing*, v. 41, n. 9, pp. 2811–2825, Sep. 1993. ISSN: 1053-587X. doi: 10.1109/78.236504.

[34] WERNER, S., DINIZ, P. S. R. "Set-membership affine projection algorithm", *IEEE Signal Processing Letters*, v. 8, n. 8, pp. 231–235, Aug 2001. ISSN: 1070-9908. doi: 10.1109/97.935739.

[35] 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, Nov 1999. ISSN: 1053-587X. doi: 10.1109/78.796429.

[36] DINIZ, P. S. R. "On data-selective adaptive filtering", *IEEE Transactions on Signal Processing*, v. 66, n. 16, pp. 4239–4252, Aug 2018. ISSN: 1053-587X. doi: 10.1109/TSP.2018.2847657.

[37] 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, Aug 2016. ISSN: 1053-587X. doi: 10.1109/TSP.2016.2546225.

[38] BERBERIDIS, D. K., KEKATOS, V., WANG, G., et al. "Adaptive censoring for large-scale regressions". In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5475–5479, April 2015. doi: 10.1109/ICASSP.2015.7179018.

[39] SPELTA, M. J. M., MARTINS, W. A. "Optimal constraint vectors for set-membership proportionate affine projection algorithms". In: *2018 IEEE Statistical Signal Processing Workshop (SSP)*, pp. 523–527, June 2018. doi: 10.1109/SSP.2018.8450820.

[40] SPELTA, M. J. M. "Brazilian Weather Stations". `http://github.com/mspelta/brazilian-weather-stations`, 2018.

[41] INSTITUTO NACIONAL DE METEOROLOGIA (INMET) . "Normais climatológicas do Brasil". `http://www.inmet.gov.br/portal/index.php?r=clima/normaisClimatologicas`.

[42] SPELTA, M. J. M. "MSc Dissertation". `https://github.com/mspelta/msc_dissertation`, 2019.

[43] BOYD, S., VANDENBERGUE, L. *Convex optimization*. Cambridge University Press, 2004.

[44] NETTO, S. L., DINIZ, P. S. R., AGATHOKLIS, P. "Adaptive IIR filtering algorithms for system identification: a general framework", *IEEE Transactions on Education*, v. 38, n. 1, pp. 54–66, Feb 1995. ISSN: 0018-9359. doi: 10.1109/13.350221.

[45] PAPOULIS, A. *Probability, Random Variables, and Stochastic Processes*. McGraw Hill, 1991.

[46] 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.

[47] BENESTY, J., GAY, S. L. "An improved PNLMS algorithm". In: *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '02)*, v. 2, pp. 1881–1884. IEEE, 2002.

[48] BENESTY, J., GÄNSLER, T., D.MORGAN, et al. *Advances in network and acoustic echo cancellation*. Springer, 2001.

[49] HOSHUYAMA, O., GOUBRAN, R. A., SUGIYAMA, A. "A generalized proportionate variable step-size algorithm for fast changing acoustic environments". In: *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '04)*, v. 4, pp. 161–164. IEEE, 2004.

[50] GÄNSLER, T., GAY, S. L., SONDHI, M. M., et al. "Double-talk robust fast converging algorithms for network echo cancellation", *IEEE Transactions on Speech and Audio Processing*, v. 8, n. 6, pp. 656–663, November 2000.

[51] 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. 2017, n. 1, pp. 72, Oct 2017.

[52] 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.

[53] 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.

[54] GALDINO, J. F., APOLINÁRIO JR., J. A., DE CAMPOS, M. L. R. "A set-membership NLMS algorithm with time-varying error bound". In: *Proceedings of the IEEE Int. Symp. Circuits Syst.*, pp. 277–280. IEEE, 2006.

[55] 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. ISSN: 1070-9908. doi: 10.1109/97.668945.

[56] LIMA, M. V. S., FERREIRA, T. N., MARTINS, W. A., et al. "Sparsity-aware data-selective adaptive filters", *IEEE Trans. Signal Process.*, v. 62, n. 17, pp. 4557–4572, September 2014.

[57] LIMA, M. V. S., DINIZ, P. S. R. "Steady-state analysis of the set-membership affine projection algorithm". In: *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3802–3805, 2010.

[58] 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, pp. 1811–1837, 2013.

[59] LIMA, M. V. S., DINIZ, P. S. R. "Fast learning set theoretic estimation". In: *Proceedings of the 21st European Signal Processing Conference (EUSIPCO 2013)*, pp. 1–5. IEEE, 2013.

[60] JIANG, K., SUN, D., TOH, K.-C. "An inexact accelerated proximal gradient method for large scale linearly constrained convex SDP", *SIAM J. optim*, v. 22, n. 3, pp. 1042–1064, 2012.

[61] GRANT, M., BOYD, S. "CVX MATLAB software for disciplined convex programming, version 2.0 beta". 2014. Disponível em: <`http://cvxr.com/cvx/`>.

[62] ANTONIOU, A., LU, W. *Practical optimization: algorithms and engineering applications*. Springer, 2007.

[63] BERTSEKAS, D. P. *Nonlinear programming*. Athena Scientific, 1999.

[64] NOCEDAL, J., WRIGHT, S. J. *Numerical optimization*. Springer, 2006.

[65] SANTIN, O., HAVLENA, V. "Combined gradient and newton projection quadratic programming solver for MPC". In: *Proceedings of the 18th World Congress - The International Federation of Automatic Control*, v. 44, pp. 5567–5572, 2011.

[66] KELLEY, C. T. *Iterative methods for optimization*. Society for Industrial and Applied Mathematics, 1999.

[67] KELLEY, C. T. "Gradient projection method MATLAB implementation". 1998. Disponível em: <`http://www.siam.org/books/kelley/fr18/OPT_CODE/gradproj.m`>.

[68] RIBEIRO, G., LIMA, J. "Graph signal processing in a nutshell", *Journal of Communication and Information Systems*, v. 33, n. 1, Jul. 2018. doi: 10.14209/jcis.2018.22. Disponível em: <`https://jcis.sbrt.org.br/jcis/article/view/563`>.

[69] PÜSCHEL, M., MOURA, J. M. F. "Algebraic signal processing theory", *CoRR*, v. abs/cs/0612077, 2006. Disponível em: <`http://arxiv.org/abs/cs/0612077`>.

[70] PÜSCHEL, M., MOURA, J. M. F. "Algebraic signal processing theory: foundation and 1-D time", *IEEE Transactions on Signal Processing*, v. 56, n. 8, pp. 3572–3585, Aug 2008. ISSN: 1053-587X. doi: 10.1109/TSP.2008.925261.

[71] NARANG, S. K., GADDE, A., ORTEGA, A. "Signal processing techniques for interpolation in graph structured data". In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 5445–5449, May 2013. doi: 10.1109/ICASSP.2013.6638704.

[72] PETERSEN, K. B., PEDERSEN, M. S. *The matrix cookbook*, 2012 (accessed November 9, 2018). Disponível em: <`https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf`>.

[73] TREFETHEN, L. N., III, D. B. *Numerical linear algebra*. SIAM: Society for Industrial and Applied Mathematics, 1997.

[74] DAMM, T. "Direct methods and ADI-preconditioned Krylov subspace methods for generalized Lyapunov equations", *Numerical Lin. Alg. with Applic.*, v. 15, pp. 853–871, 2008.

[75] LAUB, A. J. *Matrix analysis for scientists and engineers*. Philadelphia, PA, USA, SIAM: Society for Industrial and Applied Mathematics, 2004. ISBN: 9780898715767.

[76] ANDREWS, L. C. *Special functions of mathematics for engineers*. Oxford Univ. Press, 1998.

[77] HOGG, R. V., CRAIG, A. T. *Introduction to mathematical statistics*. Macmillan Publishing Co, Inc., 1978.

[78] MAHMOUD, H., AKKARI, N. "Shortest path calculation: a comparative study for location-based recommender system". In: *2016 World Symposium on Computer Applications Research (WSCAR)*, pp. 1–5, March 2016. doi: 10.1109/WSCAR.2016.16.

Appendix A

Alternative Derivations of Adaptive Algorithms

A.1 Theoretical SM-PAPA Update Equation

Based on the PAPA derivation presented in Subsection 2.6.1, for solving (3.9) we first write the respective Lagrangian

$$\begin{aligned}\mathcal{L}_{\text{sm}}\big[\hat{\mathbf{h}}[k+1]\big] = \frac{1}{2}\|\hat{\mathbf{h}}[k+1] - \hat{\mathbf{h}}[k]\|^2_{\mathbf{G}^{-1}[k]} + \\ + \boldsymbol{\lambda}^{\text{T}}_{\text{sm}}[k]\cdot\big(\,\mathbf{d}_{\text{dr}}[k] - \mathbf{X}^{\text{T}}_{\text{dr}}[k]\hat{\mathbf{h}}[k+1] - \boldsymbol{\gamma}[k]\,\big)\,,\end{aligned} \tag{A.1}$$

where $\boldsymbol{\lambda}_{\text{sm}}[k] \in \mathbb{R}^{L+1}$ represents the SM-PAPA Lagrange multipliers vector.

Considering that the constraint vector $\boldsymbol{\gamma}[k]$ is not dependent on the value of $\hat{\mathbf{h}}[k+1]$, by taking the gradient of (A.1) with respect to $\hat{\mathbf{h}}[k+1]$ and equating this expression to zero, we obtain that

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]\boldsymbol{\lambda}_{\text{sm}}[k]\,. \tag{A.2}$$

Since (3.9) indicates that $\mathbf{X}^{\text{T}}_{\text{dr}}[k]\hat{\mathbf{h}}[k+1] = \mathbf{d}_{\text{dr}}[k] - \boldsymbol{\gamma}[k]$ and we know that $\mathbf{e}[k] = \mathbf{d}_{\text{dr}}[k] - \mathbf{X}^{\text{T}}_{\text{dr}}[k]\hat{\mathbf{h}}[k]$, from equation (A.2) we find that

$$\begin{aligned}\mathbf{X}^{\text{T}}_{\text{dr}}[k]\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]\boldsymbol{\lambda}_{\text{sm}}[k] &= \mathbf{X}^{\text{T}}_{\text{dr}}[k]\hat{\mathbf{h}}[k+1] - \mathbf{X}^{\text{T}}_{\text{dr}}[k]\hat{\mathbf{h}}[k]\,,\\ \mathbf{X}^{\text{T}}_{\text{dr}}[k]\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]\boldsymbol{\lambda}_{\text{sm}}[k] &= \mathbf{d}_{\text{dr}}[k] - \mathbf{X}^{\text{T}}_{\text{dr}}[k]\hat{\mathbf{h}}[k] - \boldsymbol{\gamma}[k]\,,\\ \mathbf{X}^{\text{T}}_{\text{dr}}[k]\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]\boldsymbol{\lambda}_{\text{sm}}[k] &= \mathbf{e}[k] - \boldsymbol{\gamma}[k]\,.\end{aligned} \tag{A.3}$$

Assuming that $\mathbf{X}_{\text{dr}}[k] \in \mathbb{R}^{(M+1)\times(L+1)}$ has rank $L+1$ (where $L < M$), the matrix $\mathbf{X}^{\text{T}}_{\text{dr}}[k]\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]$ is non-singular. So, we rewrite the SM-PAPA Lagrange multiplier $\boldsymbol{\lambda}_{\text{sm}}[k]$ as

$$\boldsymbol{\lambda}_{\text{sm}}[k] = \Big(\mathbf{X}^{\text{T}}_{\text{dr}}[k]\mathbf{G}[k]\mathbf{X}_{\text{dr}}[k]\Big)^{-1}\big(\mathbf{e}[k] - \boldsymbol{\gamma}[k]\big)\,, \tag{A.4}$$

and substitute this expression in (A.2), resulting in the update equation

$$\hat{\mathbf{h}}[k+1] = \hat{\mathbf{h}}[k] + \mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k] \cdot \Big(\mathbf{X}_{\mathrm{dr}}^{\mathrm{T}}[k]\mathbf{G}[k]\mathbf{X}_{\mathrm{dr}}[k]\Big)^{-1}\big(\mathbf{e}[k] - \boldsymbol{\gamma}[k]\big)\,. \tag{A.5}$$

A.2 GSP RLS Alternative Update Equations

Due to its initialization as a diagonal matrix, from (4.25) one finds that $\mathbf{R}_{\mathrm{R}}[k]$ is symmetric. Then, based on (4.25), by taking $\mathbf{G}[k] = \mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\mathbf{R}_{\mathrm{R}}^{-1}[k-1]\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}$, the matrix inversion lemma states that $\mathbf{R}_{\mathrm{R}}^{-1}[k]$ can be written as

$$\mathbf{R}_{\mathrm{R}}^{-1}[k] = \big[\mathbf{I} - \mathbf{R}_{\mathrm{R}}^{-1}[k-1]\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}(\beta_{\mathrm{R}}\mathbf{C}_w + \mathbf{G}[k])^{-1}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\big]\,\beta_{\mathrm{R}}^{-1}\mathbf{R}_{\mathrm{R}}^{-1}[k-1]\,. \tag{A.6}$$

From (4.7) and $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$ in (4.26), it is clear that $\mathbf{R}_{\mathrm{R}}^{-1}[k]\mathbf{p}_{\mathrm{R}}[k]$ is equal to $\hat{\mathbf{s}}_{\mathcal{F}}[k+1]$. Thus, by multiplying $\mathbf{R}_{\mathrm{R}}^{-1}[k]$ in (A.6) and $\mathbf{p}_{\mathrm{R}}[k]$ from (4.25) we find $\hat{\mathbf{s}}_{\mathcal{F}}[k+1]$. Based on (4.6), $\hat{\mathbf{x}}_{\mathrm{o}}[k+1]$ is

$$\hat{\mathbf{x}}_{\mathrm{o}}[k+1] = \hat{\mathbf{x}}_{\mathrm{o}}[k] + \mathbf{U}_{\mathcal{F}}\mathbf{R}_{\mathrm{R}}^{-1}[k-1]\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}(\beta_{\mathrm{R}}\mathbf{C}_w + \mathbf{G}[k])^{-1}\mathbf{e}[k]\,. \tag{A.7}$$

When right-multiplying (A.6) by $\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_w^{-1}$ it follows that

$$\mathbf{R}_{\mathrm{R}}^{-1}[k]\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_w^{-1} = \mathbf{R}_{\mathrm{R}}^{-1}[k-1]\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}(\beta_{\mathrm{R}}\mathbf{C}_w + \mathbf{G}[k])^{-1}\,,$$

which allows us to rewrite expression (A.7) as

$$\hat{\mathbf{x}}_{\mathrm{o}}[k+1] = \hat{\mathbf{x}}_{\mathrm{o}}[k] + \mathbf{U}_{\mathcal{F}}\mathbf{R}_{\mathrm{R}}^{-1}[k]\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{C}_w^{-1}\mathbf{e}[k]\,. \tag{A.8}$$

A.3 NLMS Algorithm Alternative Derivation

For solving the constrained convex problem (5.8), we consider a Lagrange multipliers vector given by $\boldsymbol{\lambda}_{\mathrm{N}} \in \mathbb{R}^{|\mathcal{F}|}$ and write the Lagrangian function

$$\mathcal{L}_{\mathrm{N}}\big[\hat{\mathbf{s}}_{\mathcal{F}}[k+1]\big] = \|\hat{\mathbf{s}}_{\mathcal{F}}[k+1] - \hat{\mathbf{s}}_{\mathcal{F}}[k]\|_2^2 - \boldsymbol{\lambda}_{\mathrm{N}}^{\mathrm{T}}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[k] - \mathbf{U}_{\mathcal{F}}\hat{\mathbf{s}}_{\mathcal{F}}[k+1])\,. \tag{A.9}$$

Then, when taking the derivate of (A.9) with respect to $\hat{\mathbf{s}}_{\mathcal{F}}[k+1]$ and equating this expression to zero, we obtain that

$$\hat{\mathbf{s}}_{\mathcal{F}}[k+1] - \hat{\mathbf{s}}_{\mathcal{F}}[k] = \frac{1}{2}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}}\boldsymbol{\lambda}_{\mathrm{N}}\,. \tag{A.10}$$

After left-multiplying both sides of (A.10) by $(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})$ and, recalling the a

posteriori error constraint $\mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[k] - \mathbf{U}_{\mathcal{F}}\hat{\mathbf{s}}_{\mathcal{F}}[k+1]) = \mathbf{0}$ in (5.8), we have that

$$\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}(\mathbf{x}_w[k] - \mathbf{U}_{\mathcal{F}}\hat{\mathbf{s}}_{\mathcal{F}}[k]) = \frac{1}{2}(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})^2\boldsymbol{\lambda}_{\mathrm{N}}\,. \tag{A.11}$$

As the *a priori* error is defined as $\mathbf{e}[k] = \mathbf{x}_w[k] - \mathbf{U}_{\mathcal{F}}\hat{\mathbf{s}}_{\mathcal{F}}[k]$, from (A.11) we find that the Lagrange multipliers vector $\boldsymbol{\lambda}_{\mathrm{N}}$ is given by

$$\boldsymbol{\lambda}_{\mathrm{N}} = 2(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})^{-2}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\,\mathbf{e}[k]\,. \tag{A.12}$$

Finally, by replacing expression (A.12) in (A.10) we find the frequency-domain update expression

$$\hat{\mathbf{s}}_{\mathcal{F}}[k+1] = \hat{\mathbf{s}}_{\mathcal{F}}[k] + (\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{S}}\mathbf{U}_{\mathcal{F}})^{-1}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\,\mathbf{e}[k]\,, \tag{A.13}$$

which is identical to the NLMS update equation for graph signal estimation in (5.6).