MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
SECRETARIA DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
CURSO DE MESTRADO EM ENGENHARIA ELÉTRICA

ANTÓNIO LUÍS LOPES RAMOS

NOVOS ALGORITMOS ADAPTATIVOS QRD-RLS MULTICANAIS RÁPIDOS E SUAS APLICAÇÕES LCMV

Rio de Janeiro
2004

INSTITUTO MILITAR DE ENGENHARIA

ANTÓNIO LUÍS LOPES RAMOS

NOVOS ALGORITMOS ADAPTATIVOS QRD-RLS MULTICANAIS RÁPIDOS E SUAS APLICAÇÕES LCMV

Dissertação de Mestrado apresentada ao Curso de Mestrado em Engenharia Elétrica do Instituto Militar de Engenharia, como requisito parcial para obtenção do título de Mestre em Ciências em Engenharia Elétrica.

Orientador: Prof. José Antonio Apolinário Jr. - D. Sc.

Rio de Janeiro
2004

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80-Praia Vermelha
Rio de Janeiro-RJ CEP 22290-270

M489 Lopes Ramos, António Luís
Novos Algoritmos Adaptativos QRD-RLS Multicanais Rápidos e suas Aplicações LCMV, António Luís Lopes Ramos, Rio de Janeiro, Instituto Militar de Engenharia, 2004.
117 p.:il, graf., tab.

Dissertação (mestrado) – Instituto Militar de Engenharia, Rio de Janeiro, 2004.

1. Filtragem Adaptativa, Multichannel Algorithms, QR Decomposition I. Instituto Militar de Engenharia. II. Título.

CDD 006.454

INSTITUTO MILITAR DE ENGENHARIA

ANTÓNIO LUÍS LOPES RAMOS

NOVOS ALGORITMOS ADAPTATIVOS QRD-RLS MULTICANAIS RÁPIDOS E SUAS APLICAÇÕES LCMV

Dissertação de Mestrado apresentada ao Curso de Mestrado em Engenharia Elétrica do Instituto Militar de Engenharia, como requisito parcial para obtenção do título de Mestre em Ciências em Engenharia Elétrica.

Orientador: Prof. José Antonio Apolinário Jr. - D. Sc.

Aprovada em 02 de Julho de 2004 pela seguinte Banca Examinadora:

Prof. José Antonio Apolinário Jr. - D. Sc. do IME - Presidente

---

Prof. Juraci Ferreira Galdino - D. Sc. do IME

Prof. Marcello L. R. de Campos - Ph. D. da COPPE/UFRJ

Prof. Leonardo S. Resende - D. Sc. da UFSC

Rio de Janeiro
2004

Aos meus pais, Olavo António Ramos (em memória) e Joana Maria Lopes, por tudo que representam na minha vida.

AGRADECIMENTOS

Ao meu orientador, TC Prof. José Antonio Apolinário Jr., pela atenção e profissionalismo com que acompanhou a realização deste trabalho e, sobretudo, pela grande amizade e incentivo que muito contribuíram para que eu o concluísse com êxito.

Ao Cap Prof. Juraci Ferreira Galdino pela colaboração incondicional na leitura do manuscrito original e pelas valiosas sugestões.

Aos meus amigos e colegas de pesquisa, pelo apoio e pelo convívio amigável.

A todos os professores e funcionários do Departamento de Engenharia Elétrica do Instituto Militar de Engenharia que, de alguma forma, contribuíram para a realização deste trabalho.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro.

À minha mãe e aos meus irmãos Titino, Zázá, Té, Lú e meus sobrinhos maravilhosos Nely, Romy, Nany, Junior, Ruben e Rui que, mesmo à distância, foram verdadeiras fontes de apoio e inspiração.

À minha Tia Dora pelo apoio ao longo destes anos da minha estadia no Brasil.

Finalmente, um agradecimento muito especial à minha esposa Vera, companheira, amiga e colega, pelo apoio e compreensão pelos infindáveis momentos de ausência durante a realização deste trabalho.

SUMÁRIO

LISTA DE ILUSTRAÇÕES 9

LISTA DE TABELAS 10

LISTA DE ABREVIATURAS E SÍMBOLOS 11

1 INTRODUÇÃO 15
1.1 Introdução 15
1.2 Objetivo da Dissertação 16
1.3 Estado da Arte 16
1.4 Contribuições da Dissertação 17
1.5 Organização da Dissertação 18

2 FILTRAGEM ADAPTATIVA 19
2.1 Definição do Problema 19
2.2 O Algoritmo LMS 22
2.3 O Algoritmo RLS 24
2.4 Algoritmos da Família RLS Baseados na Decomposição QR 26
2.4.1 O Algoritmo QRD-RLS Convencional 26
2.4.2 O Algoritmo QRD-RLS Inverso 32
2.4.3 Interpretando as Variáveis Internas 32
2.4.4 O problema das predições progressiva e regressiva 34
2.4.4.1 A Predição Progressiva 35
2.4.4.2 A Predição Regressiva 35
2.4.5 Algoritmos QRD-RLS Rápidos 36
2.5 Resumo 38

3 FILTRAGEM ADAPTATIVA COM RESTRIÇÕES LINEARES 39
3.1 Introdução 39
3.2 Definição do Problema 39
3.3 O Algoritmo RLS com Restrições Lineares 42
3.4 Estruturas Alternativas aos Algoritmos com Restrições Lineares 42

3.4.1 A Estrutura GSC 43
3.4.2 A Estrutura *Householder Constrained* 45
3.4.3 Estrutura GSC × Estrutura Householder Constrained 46
3.4.4 O Algoritmo Householder Constrained *Conjugate Gradient* 47
3.5 Resumo 48

4 SOBRE O DESEMPENHO DOS ALGORITMOS ADAPTATIVOS DE CONVERGÊNCIA RÁPIDA COM RESTRIÇÕES 50
4.1 Introdução 50
4.2 O *Beamforming* 50
4.2.1 Uma visão Geral 50
4.2.2 O *Beamforming* Digital 52
4.2.3 O *Beamforming* Adaptativo 52
4.2.4 Arranjos Adaptativos de Antenas 53
4.2.5 Algumas definições fundamentais 54
4.3 Avaliação do Desempenho dos Vários Algoritmos 55
4.4 Comparando a performance dos algoritmos robustos 56
4.5 Resumo 59

5 ALGORITMOS MULTICANAIS RÁPIDOS 61
5.1 Introdução 61
5.2 O Algoritmo FQRD Multicanal Baseado em erros de Predição regressiva: Equações Básicas 62
5.3 O Algoritmo *Fast* QRD Multicanal Baseado na atualização do erro de predição regressiva *a priori* (MCFQRD_PRI_B) 67
5.4 O Algoritmo *Fast* QRD Multicanal Baseado na atualização do erro de predição regressiva *a posteriori* (MCFQRD_POS_B) 69
5.5 O Novo Algoritmo: MCFQD_POS_B: Versão em Treliça 71
5.6 O Algoritmo **FQRD** Multicanal para Canais de Ordens Diferentes 73
5.6.1 Aspectos Gerais 73
5.6.2 O Algoritmo Baseado na Atualização do vetor de erros *a posteriori* regressivo — Estrutura Transversal 77

5.6.3 O Algoritmo Baseado na Atualização do vetor de erros *a priori* regressivo ... 83
5.7 Estrutura em Treliça do Algoritmo Baseado na Atualização do vetor de erros *a posteriori* regressivo ... 85
5.8 Resumo ... 86

6 DESEMPENHO DOS ALGORITMOS PROPOSTOS ... 93
6.1 Complexidade Computacional dos Algoritmos Multicanais Rápidos ... 93
6.2 Resultados das Simulações ... 94
6.2.1 Simulações em Ambiente *Beamforming* ... 94
6.2.2 Filtragem de Volterra ... 97
6.2.2.1 O Problema da Filtragem não Linear ... 97
6.2.2.2 Expansão da Série de Volterra ... 98
6.2.2.3 A Filtragem de Volterra como um problema multicanal ... 99
6.2.2.4 Simulações para a Filtragem de Volterra ... 100
6.3 Resumo ... 102

7 CONCLUSÕES E COMENTÁRIOS FINAIS ... 104
7.1 Conclusões ... 104
7.2 Trabalhos Futuros ... 105
7.3 Comentários Finais ... 106

8 REFERÊNCIAS BIBLIOGRÁFICAS ... 107

9 APÊNDICES ... 111
9.1 APÊNDICE 1: Tópicos Sobre a Implementação dos Algoritmos que Fazem Uso da Decomposição QR ... 112
9.2 APÊNDICE 2: O Gradiente Complexo ... 115

LISTA DE ILUSTRAÇÕES

FIG.2.1 Configuração Básica de um Filtro Adaptativo. 20
FIG.2.2 Filtro Adaptativo FIR. 21
FIG.2.3 Filtro Adaptativo Multicanal. 22

FIG.3.1 O processo de transformação. 44
FIG.3.2 A Estrutura GSC mais usual. 45
FIG.3.3 Rotação do vetor de coeficientes. 46
FIG.3.4 Filtragem adaptativa com restrições usando a transformação de Householder vista como uma estrutura GSC. 47

FIG.4.1 Uma frente de onda plana chegando a um arranjo de antenas 52
FIG.4.2 Aproximação *far field*. 53
FIG.4.3 Convergência dos diversos algoritmos tipo RLS, com res-trições, GSC e Householder Constrained. 56
FIG.4.4 Curva de aprendizagem durante o transiente 57
FIG.4.5 *Beam–Pattern* para o caso de 20 iterações com um único experimento. 59

FIG.5.1 Filtro Adaptativo Multicanal 63
FIG.5.2 Composição do novo vetor de entrada para $N_1 = 4$, $N_2 = 3$ e $N_3 = 2$. 75
FIG.5.3 Finalizando a triangularização 81

FIG.6.1 Curva de convergência dos algoritmos multicanais rápidos num ambiente de *beamforming*. 95
FIG.6.2 A filtragem de Volterra de segunda ordem como um problema multicanal com $L = 4$. 99
FIG.6.3 O vetor de entrada $\boldsymbol{x}_N$. 101
FIG.6.4 Curva de convergência do problema da filtragem de Volterra de segunda ordem para alguns algoritmos. 102

FIG.9.1 As diferentes triangularizações de $\boldsymbol{U}(k)$: (a) SUPERIOR e (b) INFERIOR. 112

LISTA DE TABELAS

TAB.2.1 O algoritmo LMS 23
TAB.2.2 O Algoritmo RLS 26
TAB.2.3 Equações do algoritmo QRD-RLS convencional. 29
TAB.2.4 Equações do algoritmo QRD-RLS Inverso. 33
TAB.2.5 Classificação dos algoritmos QRD rápidos (FQRD) 37

TAB.3.1 O Algoritmo CRLS básico 43
TAB.3.2 Cálculo de $\bar{\boldsymbol{x}}(k) = \boldsymbol{Q}\boldsymbol{x}(k)$. 48
TAB.3.3 Construindo $\boldsymbol{V}$ e Obtendo $\boldsymbol{Q}$. 48
TAB.3.4 O algoritmo *HCCG*. 49

TAB.4.1 Complexidade computacional dos algoritmos mais robustos. 58
TAB.4.2 Complexidade computacional (exemplo numérico). 58

TAB.5.1 Equações do Algoritmo MCFQRD_PRI_B. 87
TAB.5.2 Equações do Algoritmo MCFQRD_POS_B. 88
TAB.5.3 As Equações do MCFQRD_POS_B Versão em Treliça. 89
TAB.5.4 Pseudo-código da etapa 4: obtendo ${\boldsymbol{Q}'_{\theta f}}^{(N-i+1)}(k+1)$. 90
TAB.5.5 O Algoritmo Multicanal rápido para canais de ordens distintas (*erro a posteriori*)—Estrutura Transversal. 91
TAB.5.6 O Algoritmo Multicanal rápido para canais de ordens diferentes (*erro a posteriori*)—Estrutura em Treliça. 92

TAB.6.1 Complexidade Computacional dos Algoritmos Multicanais Rápidos 94
TAB.6.2 Complexidade Computacional dos Algoritmos Multicanais Rápidos no ambiente *beamforming*: exemplo numérico. 96
TAB.6.3 Comp. Comput. dos Algoritmos MCFQRD-RLS: exemplo numérico para $M = 4$ e $N = 50$. 97
TAB.6.4 Complexidade Computacional dos Algoritmos Multicanais Rápidos na Filtragem de Volterra: exemplo numérico. 103

TAB.9.1 Pseudo-código do algoritmo QRD-RLS. 114

ABREVIATURAS

CG	-	Conjugate Gradient
CCG	-	Constrained Conjugate Gradient
CDMA	-	Code Division Multiple Access
CIQRD	-	Constrained Inverse QR Decomposition
CLMS	-	Constrained Least Mean Square
CQN	-	Constrained *Quasi* Newton
CQRD	-	Constrained QR Decomposition
CRLS	-	Constrained Recursive Least Squares
dB	-	decibel
FDMA	-	Frequency Division Multiple Access
FIR	-	Finite Impulse Response
FQRD	-	Fast QR Decomposition
GSC	-	Generalized Sidelobe Canceller
HCCG	-	Householder Constrained Conjugate Gradient
HCIQRD	-	Householder Constrained Inverse QR Decomposition
HCQN	-	Householder Constrained *Quasi* Newton
HCRLS	-	Householder Constrained Recursive Least Squares
IIR	-	Infinite Impulse Response
IQRD	-	Inverse QR Decomposition
LCMV	-	Linearly Constrained Minimum Variance
LMS	-	Least Mean Square
LS	-	Least Squares
MCFQRD	-	Multichannel Fast QR Decomposition
MSE	-	Mean Square Error
MVDR	-	Minimum Variance Distortionless Response
NLMS	-	Normalized Least Mean Square
QN	-	*Quasi* Newton
QRD	-	QR Decomposition
RLS	-	Recursive Least Squares

RSR	-	Razão Sinal Ruido
SDMA	-	Space Division Multiple Access
TDMA	-	Time Division Multiple Access

RESUMO

Recentemente, o processamento digital de sinais multicanais usando filtros adaptativos tem encontrado uma vasta gama de novas aplicações, incluindo processamento de imagens em cores, biomedicina, equalização de canais, cancelamento de eco estereofônico, processamento de sinais multi–dimensionais, identificação de sistemas não–lineares do tipo Volterra e melhoria de sinais de voz. Este número crescente de aplicações tem despertado o interesse pela busca de algoritmos multicanais mais eficientes.

A classe de algoritmos adaptativos, conhecidos como algoritmos multicanais rápidos (MCFQRD-RLS – *multichannel Fast* QRD-RLS) baseados na atualização de erros regressivos, tornou-se uma opção atraente por causa da sua convergência rápida, estabilidade e complexidade computacional reduzida. O foco principal desta dissertação são estes algoritmos e a suas aplicações LCMV. Com base em simulações, usando versões com e sem restrições (neste último caso, fazendo uso de estruturas apropriadas que permitem o uso de algoritmos sem restrições para resolver problemas com restrições) disponíveis atualmente na literatura de filtragem adaptativa, as melhores para este tipo de aplicação são apontadas.

Três novas versões do algoritmo multicanal rápido (MCFQRD-RLS) baseado na atualização do erro regressivo ou *backward* são propostas. Estas novas versões mostraram-se igualmente estáveis, robustas e de rápida convergência porém com uma complexidade computacional menor em relação às versões atualmente disponíveis na literatura. Dois destes novos algoritmos possuem a vantagem de poderem ser utilizados em aplicações em que se requer canais com a mesma ordem ou com ordens distintas.

ABSTRACT

Digital processing of multichannel signals using adaptive filters has recently found a variety of new applications including color image processing, biomedicine, channel equalization, stereophonic echo cancellation, multidimensional signal processing, Volterra–type nonlinear system identification, and speech enhancement. This increased number of applications has spawned a renewed interest in efficient multichannel algorithms.

One class of algorithms, known as multichannel Fast QRD-RLS adaptive algorithms based on backward error updating, has become an attractive option because of their fast convergence, stability, and reduced computational complexity. This dissertation is concerned with this class of multichannel algorithms and their applications to Linear Constrained Minimum Variance (LCMV) adaptive filtering. Simulations are carried out using constrained and unconstrained adaptive filtering algorithms currently available in the adaptive filtering literature and the best options for this kind of application are pointed out.

Based on those results, three new versions of the multichannel Fast QRD-RLS algorithm based on backward error updating are proposed. These new algorithms exhibit good properties concerning stability, robustness, speed of convergence, and lower computational complexity when compared to other multichannel fast QRD-RLS algorithms available in the literature. Two of them can be considered as more general solutions for they can be used in multichannel adaptive filtering applications where either equal or unequal channel orders are required.

1.1 INTRODUÇÃO

A filtragem adaptativa constitui-se, atualmente, num campo de grande importância na área de Processamento de Sinais. O uso de filtros adaptativos torna-se particularmente necessário em ambientes onde as características do sinal de interesse são desconhecidas ou quando as estatísticas deste ambiente variam com o tempo. São inúmeras as aplicações em que o uso de filtros adaptativos se faz necessário e, graças à grande quantidade de trabalhos desenvolvidos ao longo das últimas três décadas nesta área, muitas soluções já foram apresentadas para este tipo de problema. Porém, a busca por soluções cada vez mais robustas e rápidas, a um custo computacional aceitável, tem motivado a continuação de pesquisas na área para o desenvolvimento de novos algoritmos.

Um filtro é, basicamente, um dispositivo de *hardware* ou de *software* cujos parâmetros são projetados para extrair alguma característica de interesse de um sinal a partir de dados disponíveis até um determinado instante k. Quando estes parâmetros são variantes no tempo, o dispositivo é geralmente conhecido como um filtro adaptativo. A atualização destes parâmetros (coeficientes do filtro) é feita por meio de um algoritmo que busca minimizar uma função custo previamente estabelecida.

Na comparação da grande variedade de algoritmos adaptativos disponíveis na literatura, são levados em consideração os critérios seguintes (HAYKIN, 1996).

- **Estrutura**. Diz respeito à maneira pela qual o algoritmo é implementado que, para os filtros adaptativos de resposta ao impulso finita, FIR (*finite impulse response*), pode ser dividida em dois tipos: transversal (conhecido no inglês por *tapped delay line*) e treliça (*lattice* em inglês). Existem ainda os filtros adaptativos de resposta ao impulso infinita, IIR (*infinite impulse response*), que podem ser realizados numa grande variedade de estruturas; tais filtros estão fora do escopo deste trabalho.

- **Taxa de convergência**: diz respeito a quão rápido os coeficientes do filtro se aproximam da solução ótima que, em situações práticas, nunca é alcançada; **desajuste**: é a medida de quão próximo os coeficientes estimados se encontram

dos ótimos; e **acompanhamento**: diz respeito à capacidade dos coeficientes estimados pelo algoritmo acompanharem os ótimos em ambientes não-estacionários.

- **Aspectos computacionais**: podem ser incluidos a complexidade computacional e a imunidade a erros numéricos, principalmente em ambientes de precisão finita que, geralmente, geram erros de quantização e, por vezes, instabilidade numérica.

O foco principal deste trabalho são os algoritmos de convergência rápida baseados no algoritmo RLS (*Recursive Least Squares*) e suas aplicações em LCMV (*Linearly Constrained Minimum Variance*). Estes algoritmos, por fazerem uso da decomposição QR baseada em rotações de Givens, que são numericamente estáveis, foram desenvolvidos com a finalidade de resolver o já bem conhecido problema de instabilidade do algoritmo RLS.

1.2 OBJETIVO DA DISSERTAÇÃO

O objetivo desta dissertação é fazer uma avaliação e comparação do desempenho dos algoritmos de convergência rápida disponíveis na literatura no que tange à estabilidade, velocidade de convergência e complexidade computacional em ambientes de *beamforming*. A utilização dos algoritmos multicanais rápidos baseados no RLS neste tipo de problema é investigada e novas versões eficientes destes algoritmos são derivadas.

1.3 ESTADO DA ARTE

A chamada filtragem espacial, ou *beamforming*, tem sido uma técnica usada há vários anos, na sua versão analógica, para solucionar vários tipos de problemas como (VAN VENN, 1988): radar (controle de tráfego), sonar (classificação de fonte), comunicações (transmissão direcional, rádio–difusão setorizado em comunicações via satélite), imagem (tomografia, ultra–som), exploração geofísica (exploração de petróleo, mapeamento), exploração astrofísica (imagem de alta resolução do universo) e biomedicina (monitoração do coração dos fetos, ajuda auditiva).

Um *beamformer* é implementado com o auxílio de um arranjo de sensores convenientemente espaçados cuja finalidade é receber um sinal proveniente de uma determindada direção de interesse e rejeitar sinais interferentes proveniente de direções indesejáveis. A

viabilização da aplicação prática desta técnica em sistemas de comunicações digitais depara com alguns obstáculos tais como a necessidade de um processador digital adequado e, conseqüentemente, de um algoritmo adaptativo apropriado que seja suficientemente robusto, rápido e com uma complexidade computacional atraente. A necessidade de um algoritmo rápido se torna crítica quando é considerado o caso em que pelo menos um dos pontos em um enlace de comunicação é móvel requerendo, assim, rastreamento e adaptação constantes, como é o caso do SDMA (*Space–Division Multiple Access*) (LITVA, 1996).

Normalmente, os algoritmos adaptativos a serem utilizados nestes casos devem ser tais que a adaptação seja feita sujeita a restrições convenientemente impostas. A maioria dos algoritmos que incorporam restrições lineares na sua solução, quando usados para solucionar problemas em aplicação LCMV, apresentam problemas de estabilidade (RAMOS, 2004c). Até o presente, o uso de estruturas que permitem a utilização de algoritmos sem restrições estáveis e robustos se apresentam como soluções viáveis, pelo menos quanto à robustez.

Os algoritmos multicanais rápidos apresentam bom desempenho neste tipo de aplicação, pelo menos quanto à robustez e à velocidade de convergência, porém, a uma complexidade computacional relativamente alta.

1.4 CONTRIBUIÇÕES DA DISSERTAÇÃO

Nesta dissertação é feita uma avaliação do desempenho dos algoritmos de convergência rápida com restrições em aplicações LCMV. Com base em simulações, são apresentadas as melhores opções de algoritmos disponíveis na literatura até o momento. Como será visto, a utilização de estruturas como a GSC (*Generalized Sidelobe Canceller*) (GRIFFITHS, 1982) e a Householder Constrained (DE CAMPOS, 2002) são alternativas viáveis por permitirem que se utilize algoritmos sem restrições e estáveis para solucionar este tipo de problema.

São introduzidas versões sem restrições otimizadas dos algoritmos multicanais rápidos baseados na decomposição QR são introduzidas: uma versão em treliça para o algoritmo multicanal em bloco (os sinais de entrada dos canais são processados de uma única vez) mais duas versões, uma transversal e outra em treliça, para o algoritmo multicanal sequencial (os sinais de entrada dos canais são processados seqüencialmente).

No Capítulo 2, são apresentados alguns conceitos básicos de filtragem adaptativa. No Capítulo 3 a filtragem adaptativa com restrições é revista e o algoritmo RLS com restrições é apresentado na sua versão básica. O desempenho dos algoritmos adaptativos de convergência rápida com restrições em aplicações LCMV é avaliado no Capítulo 4. No Capítulo 5, são abordados os algoritmos multicanais rápidos baseados no RLS, onde são também introduzidas três novas versões robustas destes algoritmos. A avaliação do desempenho destes novos algoritmos em aplicações LCMV é feita no Capítulo 6, onde também é considerada uma aplicação de fitragem de Volterra, dada às características particularmente favoráveis a estas aplicações das versões sequenciais dos algoritmos multicanais abordados neste trabalho. Conclusões e comentátios finais são apresentados no Capítulo 7.

Neste capítulo são abordados, de forma resumida, alguns conceitos básicos de filtragem adaptativa. Na Seção 2.1 serão apresentados os principais fundamentos, incluindo a solução de Wiener. Os algoritmos LMS (*Least Mean–Square*) e RLS (*Recursive Least Squares*) são vistos nas Seções 2.2 e 2.3, respectivamente. Na Seção 2.4, são introduzidos os algoritmos baseados no RLS que fazem uso da decomposição QR (QRD-RLS), onde são também revistos alguns tópicos básicos para o entendimento dos algoritmos da família QRD-RLS rápidos.

2.1 DEFINIÇÃO DO PROBLEMA

Uma configuração básica de um filtro adaptativo discreto no tempo está representada na FIG. 2.1. Observamos nesta figura que $\boldsymbol{x}(k)$ é o vetor sinal de entrada, $d(k)$ é o sinal "desejado" (ou de referência), $y(k)$ é a saída do filtro adaptativo e $\varepsilon(k)$ é o erro entre o sinal desejado e a saída do filtro. Este sinal de erro é empregado para ajustar o vetor de coeficientes do filtro, $\boldsymbol{w}(k)$, por meio de um algoritmo de adaptação. O processo de adaptação dos coeficientes de um filtro adaptativo é realizado visando a minimização de uma determinada função custo. A função custo de uso mais comum é o erro médio quadrático, MSE, do inglês *Mean-Square Error*, que é definida como

$$\xi(k) = E[|\varepsilon^2(k)|] = E[\varepsilon(k)\varepsilon^*(k)], \tag{2.1}$$

na qual $E[\cdot]$ denota o operador valor esperado e

$$\varepsilon(k) = d(k) - \boldsymbol{w}^H(k-1)\boldsymbol{x}(k) \tag{2.2}$$

é o erro *a priori* (antes da atualização dos coeficientes) no instante k.

Em muitas aplicações é usada uma estrutura de um filtro de resposta ao impulso finita, FIR, do inglês *Finite Impulse Response*, onde o vetor de entrada $\boldsymbol{x}(k)$ é composto por uma amostra mais recente e os demais elementos são amostras dos instantes de tempo passados até um retardo máximo, como na ilustração da FIG. 2.2, para o caso monocanal. Neste caso, a saída $y(k)$ é dada por

$$y(k) = \sum_{j=1}^{N} w_j^*(k-1)x(k-j+1) = \boldsymbol{w}^H(k-1)\boldsymbol{x}(k). \tag{2.3}$$

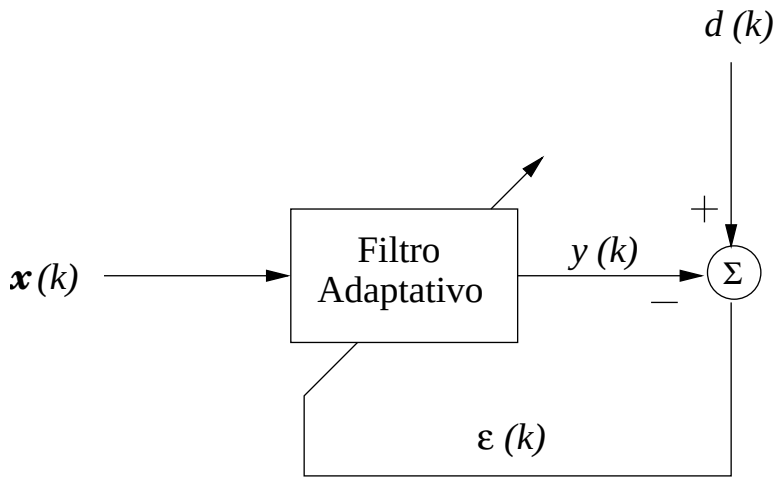

FIG. 2.1: Configuração Básica de um Filtro Adaptativo.

A filtragem adaptativa multicanal fica caracterizada quando se deseja processar, simultaneamente, M sinais captados através de um arranjo (*array*) de M sensores, como na FIG. 2.3. Nesta situação, a saída pode ser expressa como

$$\begin{aligned} y(k) &= \sum_{i=1}^{N} \boldsymbol{w}_i^H(k-1)\boldsymbol{x}_M(k-i+1) \\ &= \boldsymbol{w}^H(k-1)\boldsymbol{x}(k), \end{aligned} \tag{2.4}$$

onde

$$\begin{aligned} \boldsymbol{w}(k-1) &= \left[\begin{array}{cccc} \boldsymbol{w}_1^T(k-1) & \boldsymbol{w}_2^T(k-1) & \cdots & \boldsymbol{w}_N^T(k-1) \end{array} \right]^T, \\ \boldsymbol{x}(k) &= \left[\begin{array}{cccc} \boldsymbol{x}_M^T(k) & \boldsymbol{x}_M^T(k-1) & \cdots & \boldsymbol{x}_M^T(k-N+1) \end{array} \right]^T \text{ e} \\ \boldsymbol{x}_M(k) &= \left[\begin{array}{cccc} x_1(k) & x_2(k) & \cdots & x_M(k) \end{array} \right]^T. \end{aligned} \tag{2.5}$$

e os $\boldsymbol{w}_j(k-1)$, com $j = 1, 2, \cdots, N$ são dados por

$$\boldsymbol{w}_j(k-1) = \left[\begin{array}{cccc} w_1(k-j) & w_2(k-j) & \cdots & w_N(k-j) \end{array} \right]^T.$$

Se a entrada do filtro adaptativo for multicanal com M canais e não houver retardos em cada canal, este filtro é conhecido como combinador linear adaptativo e a saída do mesmo é dada por

$$y(k) = \sum_{j=1}^{M} w_j^*(k-1)x_j(k) = \boldsymbol{w}^H(k-1)\boldsymbol{x}(k), \tag{2.6}$$

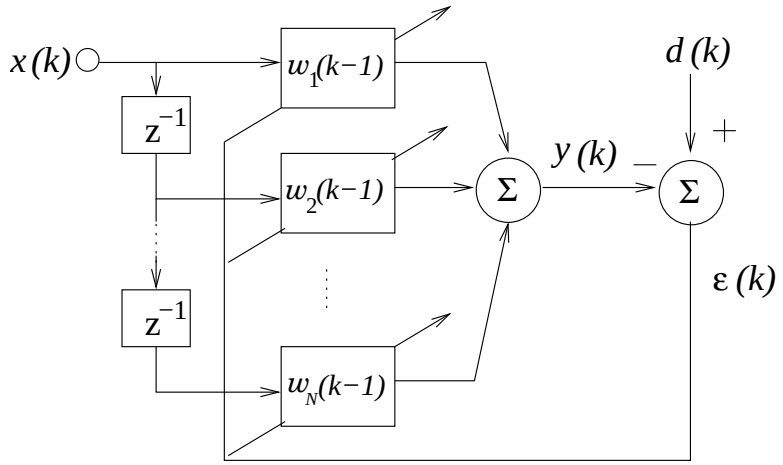

FIG. 2.2: Filtro Adaptativo FIR.

onde

$$\boldsymbol{x}(k) = [x_1(k) \quad x_2(k) \quad \cdots \quad x_M(k)]^T$$

com $x_i(k)$ correspondendo ao *i-ésimo* canal de entrada.

Em qualquer caso, substituindo a EQ. 2.2 na EQ. 2.1 obtém-se

$$\begin{aligned}\xi(k) &= E\left[\left\{d(k) - \boldsymbol{w}^H(k-1)\boldsymbol{x}(k)\right\}\left\{d^*(k) - \boldsymbol{x}^H(k)\boldsymbol{w}(k-1)\right\}\right] \\ &= E[d(k)d^*(k)] - E[d(k)\boldsymbol{x}^H(k)\boldsymbol{w}(k-1)] - E[d^*(k)\boldsymbol{w}^H(k-1)\boldsymbol{x}(k)] \\ &\quad + E[\boldsymbol{w}^H(k-1)\boldsymbol{x}(k)\boldsymbol{x}^H(k)\boldsymbol{w}(k-1)]. \qquad (2.7)\end{aligned}$$

Sejam a matriz de autocorrelação do sinal de entrada, $\boldsymbol{R}$, e o vetor de correlação cruzada entre o sinal de desejado e o de entrada, $\boldsymbol{p}$, dados, respectivamente, por

$$\boldsymbol{R} = E[\boldsymbol{x}(k)\boldsymbol{x}^H(k)] \qquad (2.8)$$

e

$$\boldsymbol{p} = E[d^*(k)\boldsymbol{x}(k)]. \qquad (2.9)$$

Considerando o vetor de coeficientes, $\boldsymbol{w}(k-1)$, como sendo fixo (e determinístico), $\boldsymbol{w}$, o gradiente em relação ao mesmo, $\nabla_{\boldsymbol{w}}\xi(k)$, da EQ. 2.7 resulta em (Apêndice 1)

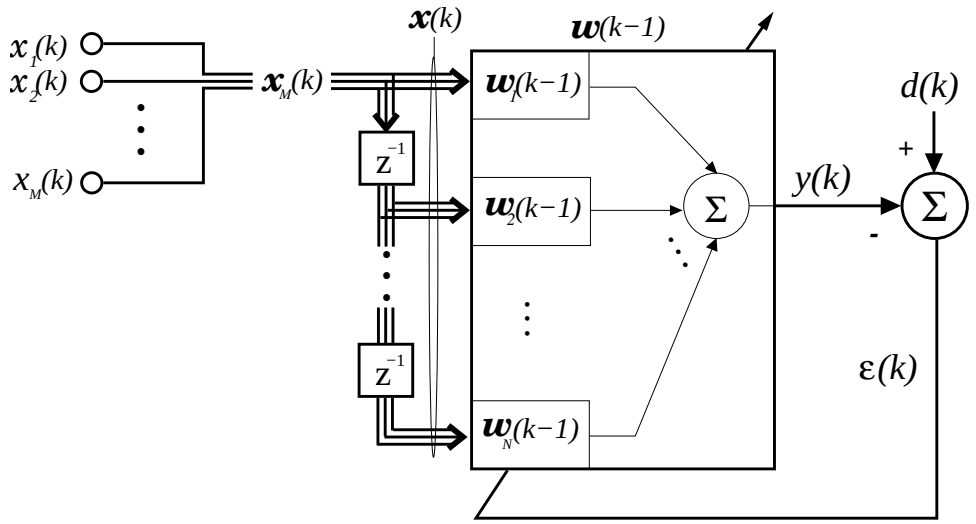

FIG. 2.3: Filtro Adaptativo Multicanal.

$$\nabla_{\boldsymbol{w}}\xi(k) = -2\boldsymbol{p} + 2\boldsymbol{R}\boldsymbol{w}. \tag{2.10}$$

Igualando a última equação a zero, e assumindo que a matriz $\boldsymbol{R}$ seja não singular, o vetor de coeficientes ótimo que minimiza a função custo fica dado por

$$\boldsymbol{w}_{opt} = \boldsymbol{R}^{-1}\boldsymbol{p}. \tag{2.11}$$

A EQ. 2.11 é conhecida como a solução de Wiener (HAYKIN, 1996). Na prática, porém, é mais comum a obtenão do vetor de coeficientes, $\boldsymbol{w}$, por meio de um algoritmo adaptativo.

2.2 O ALGORITMO LMS

O algoritmo LMS (*Least Mean Square*) continua sendo o mais usado em aplicações práticas por duas razões fundamentais: a sua simplicidade, implicando em um baixo custo computacional, e a garantia de sua convergência para escolha apropriada do seu passo de adaptação.

O algoritmo LMS é baseado no algoritmo *steepest–descent* cuja adaptação do vetor de coeficientes é dada por (HAYKIN, 1996; DINIZ, 2002):

$$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) - \mu\nabla_{\boldsymbol{w}}\xi(k), \tag{2.12}$$

TAB. 2.1: O algoritmo LMS

LMS

Inicializar:
$\boldsymbol{x}(0) = \boldsymbol{w}(0) = \mathbf{0}$
for $k = 1, 2, \ldots$
{
$\varepsilon(k) = d(k) - \boldsymbol{w}^H(k-1)\boldsymbol{x}(k)$
$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) + \mu\varepsilon^*(k)\boldsymbol{x}(k)$
}

onde μ é conhecido por *step-size* ou largura de passo de adaptação. O valor de μ deve ser criteriosamente escolhido para garantir a convergência do algoritmo. Uma abordagem deste assunto pode ser encontrado em (GALDINO, 2003).

Na prática, o gradiente $\nabla_{\boldsymbol{w}}\xi(k) = -2\boldsymbol{p} + 2\boldsymbol{R}\boldsymbol{w}$ não pode ser obtido com exatidão. Assim, usando estimativas muito simples de $\boldsymbol{p}$ e de $\boldsymbol{R}$ dadas por

$$\hat{\boldsymbol{p}} = \boldsymbol{x}(k)d^*(k) \tag{2.13}$$

e

$$\hat{\boldsymbol{R}} = \boldsymbol{x}(k)\boldsymbol{x}^H(k), \tag{2.14}$$

pode-se obter uma estimativa despolarizada do gradiente como

$$\begin{aligned}\hat{\nabla}_{\boldsymbol{w}}\xi(k) &= -2\boldsymbol{x}(k)d^*(k) + 2\boldsymbol{x}(k)\boldsymbol{x}^H(k)\boldsymbol{w}(k-1) \\ &= -2\boldsymbol{x}(k)\left[d^*(k) - \boldsymbol{x}^H(k)\boldsymbol{w}(k-1)\right] \\ &= -2\varepsilon^*(k)\boldsymbol{x}(k).\end{aligned} \tag{2.15}$$

Substituindo a estimativa na EQ. 2.15 na EQ. 2.12 do algoritmo *steepest–descent* obtemos a seguinte relação para a atualização do vetor de coeficientes correspondente ao algoritmo LMS.

$$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) + 2\mu\varepsilon^*(k)\boldsymbol{x}(k) \tag{2.16}$$

As equações referentes ao algoritmo LMS estão na TAB. 2.1, onde o *step-size* incorpora o número 2, pois, na prática, μ é inicializado apropriadamente para que não seja necessário multiplicá-lo por 2 a cada iteração.

O algoritmo RLS (*Recursive Least Squares*) pertence à classe dos algoritmos de convergência. Porém, além de uma elevada complexidade computacional, ele apresenta sérios problemas de instabilidade que por vezes impedem o seu uso em aplicações práticas. Mesmo assim, ele serviu de base para a derivação de vários outros algoritmos mais robustos, em particular os algoritmos que fazem uso da decomposição QR, obtida através das rotações de Givens que são numericamente estáveis (GOLUB, 1983; LING, 1991).

Para esta classe de algoritmos, a função custo é definida como

$$\xi(k) = \sum_{i=0}^{k} \lambda^{k-i} |e(i)|^2 = \sum_{i=0}^{k} \lambda^{k-i} |d(i) - \boldsymbol{w}^H(k)\boldsymbol{x}(i)|^2 = \boldsymbol{e}^H(k)\boldsymbol{e}(k), \tag{2.17}$$

sendo λ o fator de esquecimento e $\boldsymbol{e}(k)$ o vetor de erros dado por

$$\boldsymbol{e}(k) = \boldsymbol{d}(k) - \boldsymbol{X}^T(k)\boldsymbol{w}^*(k) \tag{2.18}$$

ou

$$\boldsymbol{e}^*(k) = \boldsymbol{d}^*(k) - \boldsymbol{X}^H(k)\boldsymbol{w}(k). \tag{2.19}$$

Na equação acima, $\boldsymbol{d}(k)$ é o vetor com o sinal de referência (desejado) ponderado, $\boldsymbol{w}(k)$ é o vetor de coeficientes do filtro e $\boldsymbol{X}(k)$ é a matriz de informação de entrada dados por

$$\boldsymbol{d}(k) = \begin{bmatrix} d(k) \\ \lambda^{1/2} d(k-1) \\ \vdots \\ \lambda^{k/2} d(0) \end{bmatrix}, \tag{2.20}$$

$$\boldsymbol{w}(k) = \begin{bmatrix} w_1(k) \\ w_2(k) \\ \vdots \\ w_N(k) \end{bmatrix} \tag{2.21}$$

e

$$\boldsymbol{X}^H(k) = \begin{bmatrix} \boldsymbol{x}^H(k) \\ \lambda^{1/2} \boldsymbol{x}^H(k-1) \\ \vdots \\ \lambda^{k/2} \boldsymbol{x}^H(0) \end{bmatrix}. \tag{2.22}$$

Aqui, N representa a ordem do filtro (e também o número de coeficientes) e $\boldsymbol{x}(k) = [x(k) \quad x(k-1) \quad \cdots \quad x(k-N+1)]$ é o vetor de entrada.[1]

A solução de Wiener para a EQ. 2.17 é obtida de forma similar ao que foi feito na Seção 2.1, isto é,

$$\boldsymbol{w}(k) = \boldsymbol{R}^{-1}(k)\boldsymbol{p}(k), \tag{2.23}$$

com $\boldsymbol{R}(k)$ e $\boldsymbol{p}(k)$ sendo dados, respectivamente, por

$$\begin{aligned} \boldsymbol{R}(k) &= \boldsymbol{X}(k)\boldsymbol{X}^H(k) \\ &= \sum_{i=0}^{k} \lambda^{k-i}\boldsymbol{x}(i)\boldsymbol{x}^H(i) \end{aligned} \tag{2.24}$$

e

$$\begin{aligned} \boldsymbol{p}(k) &= \boldsymbol{X}(k)\boldsymbol{d}^*(k) \\ &= \sum_{i=0}^{k} \lambda^{k-i}\boldsymbol{x}(i)d^*(i). \end{aligned} \tag{2.25}$$

Ao invés das expressões dadas pela EQ. 2.24 e pela EQ. 2.25, $\boldsymbol{R}(k)$ e $\boldsymbol{p}(k)$ podem ser obtidos de forma recursiva, como segue.

$$\begin{aligned} \boldsymbol{R}(k) &= \sum_{i=0}^{k} \lambda^{k-i}\boldsymbol{x}(i)\boldsymbol{x}^H(i) \\ &= \boldsymbol{x}(k)\boldsymbol{x}^H(k) + \lambda\boldsymbol{R}(k-1) \end{aligned} \tag{2.26}$$

$$\begin{aligned} \boldsymbol{p}(k) &= \sum_{i=0}^{k} \lambda^{k-i}\boldsymbol{x}(i)d^*(i) \\ &= \boldsymbol{x}(k)d^*(k) + \lambda\boldsymbol{p}(k-1) \end{aligned} \tag{2.27}$$

Substituindo a EQ. 2.27 na EQ. 2.23 e usando também a EQ. 2.26, após algumas manipulações algébricas, chega-se a

$$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) + \varepsilon^*(k)\boldsymbol{R}^{-1}(k)\boldsymbol{x}(k), \tag{2.28}$$

na qual, $\varepsilon(k)$ é o erro *a priori* definido como

$$\varepsilon(k) = d(k) - \boldsymbol{w}^H(k-1)\boldsymbol{x}(k). \tag{2.29}$$

[1] É abordado aqui, de forma resumida, o caso monocanal mas as mesmas equações se aplicam ao caso multicanal.

TAB. 2.2: O Algoritmo RLS

RLS

Inicializações:
$\boldsymbol{w}(0) = \mathbf{0}$ e $\boldsymbol{R}^{-1}(0) = \eta \boldsymbol{I}$ onde η = número pequeno
for $k = 1, 2, \cdots$
{
$\varepsilon(k) = d(k) - \boldsymbol{w}^H(k-1)\boldsymbol{x}(k)$
$\boldsymbol{k}(k) = \boldsymbol{R}^{-1}(k-1)\boldsymbol{x}(k)$
$\boldsymbol{\kappa}(k) = \frac{\boldsymbol{k}(k)}{\lambda + \boldsymbol{x}^H(k)\boldsymbol{k}(k)}$
$\boldsymbol{R}^{-1}(k) = \frac{1}{\lambda}\left[\boldsymbol{R}^{-1}(k-1) - \frac{\boldsymbol{k}(k)\boldsymbol{k}^H(k)}{\lambda + \boldsymbol{x}^H(k)\boldsymbol{k}(k)}\right]$
$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) + \varepsilon^*(k)\boldsymbol{\kappa}(k)$
}

A inversão de matriz na EQ. 2.28 pode ser evitada se aplicarmos o Lema de Inversão de Matrizes à EQ. 2.26 (DINIZ, 2002)[2], obtendo-se:

$$\boldsymbol{R}^{-1}(k) = \frac{1}{\lambda}\left[\boldsymbol{R}^{-1}(k-1) - \frac{\boldsymbol{R}^{-1}(k-1)\boldsymbol{x}(k)\boldsymbol{x}^H(k)\boldsymbol{R}^{-1}(k-1)}{\lambda \boldsymbol{x}^H(k)\boldsymbol{R}^{-1}(k-1)\boldsymbol{x}(k)}\right] \tag{2.30}$$

O algoritmo RLS completo está na TAB. 2.2. Note que, para facilitar a notação, a EQ. 2.30 foi definida em função de duas variáveis auxiliares: $\boldsymbol{k}(k) = \boldsymbol{R}^{-1}(k-1)\boldsymbol{x}(k)$, conhecido como o ganho de Kalmam e $\boldsymbol{\kappa}(k) = \frac{\boldsymbol{k}(k)}{\lambda + \boldsymbol{x}^H(k)\boldsymbol{k}(k)}$.

2.4 ALGORITMOS DA FAMÍLIA RLS BASEADOS NA DECOMPOSIÇÃO QR

Como foi mencionado anteriormente, o algoritmo RLS possui uma rápida convergência mas é instável (KIM, 1991). Para solucionar o problema da instabilidade, foram desenvolvidos novos algoritmos baseados no RLS básico fazendo uso da decomposição QR (QRD) da matriz de informação por meio das rotações de Givens que são numericamente estáveis. Nesta seção, abordaremos os algoritmos QRD-RLS básico, QRD-RLS inverso e os da família QRD-RLS rápidos.

2.4.1 O ALGORITMO QRD-RLS CONVENCIONAL

O algoritmo QRD-RLS minimiza a mesma função custo que o algoritmo RLS dada pela EQ. 2.17. A decomposição QR, aplicada à matriz de informação, $\boldsymbol{X}(k)$, efetua sua

[2] $[\boldsymbol{A} + \boldsymbol{BCD}]^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{B}[\boldsymbol{DA}^{-1}\boldsymbol{B} + \boldsymbol{C}^{-1}]^{-1}\boldsymbol{DA}^{-1}$

triangularização e evita a necessidade de se lidar com sua inversa, fonte de instabilidade numérica. Seja $\boldsymbol{Q}(k)$ a matriz ortogonal com as rotações de Givens envolvidas neste processo; pré-multiplicando a EQ. 2.19 por $\boldsymbol{Q}(k)$, a norma da função custo não é alterada. Assim,

$$\boldsymbol{Q}(k)\boldsymbol{e}^*(k) = \begin{bmatrix} \boldsymbol{e}_{q1}(k) \\ \boldsymbol{e}_{q2}(k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{d}_{q1}(k) \\ \boldsymbol{d}_{q2}(k) \end{bmatrix} - \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{U}(k) \end{bmatrix} \boldsymbol{w}(k) \tag{2.31}$$

onde $\boldsymbol{e}_q$ e $\boldsymbol{d}_q$ são, respectivamente, o conjugado erro e do sinal desejado rotacionados; $\boldsymbol{U}(k)$ é o fator de Cholesky de $\boldsymbol{R}(k) = \boldsymbol{X}(k)\boldsymbol{X}^H(k)$, isto é, $\boldsymbol{U}^H(k)\boldsymbol{U}(k) = \boldsymbol{X}(k)\boldsymbol{X}^H(k)$. Os sub-índices 1 e 2 indicam os primeiros $k - N + 1$ e últimos N elementos do vetor, respectivamente.

A minimização da função custo é conseguida escolhendo-se um $\boldsymbol{w}(k)$ tal que $\boldsymbol{e}_{q2} = \boldsymbol{d}_{q2} - \boldsymbol{U}(k)\boldsymbol{w}(k) = 0$. Logo,

$$\boldsymbol{w}(k) = \boldsymbol{U}^{-1}(k)\boldsymbol{d}_{q2}(k). \tag{2.32}$$

Se necessário, o vetor de coeficientes $\boldsymbol{w}(k)$ pode ser calculado usando um procedimento de substituição regressiva ou *backward substitution* (HAYKIN, 1996; DINIZ, 2002).

Pelo fato de $\boldsymbol{Q}(k)$ ser ortogonal e usando a definição de $\boldsymbol{X}(k)$ na EQ. 2.22, o produto $\boldsymbol{Q}(k)\boldsymbol{X}^H(k)$ pode ser escrito como segue

$$\boldsymbol{Q}(k) \underbrace{\begin{bmatrix} 1 & \boldsymbol{0}^T \\ \boldsymbol{0} & \boldsymbol{Q}^H(k-1) \end{bmatrix} \begin{bmatrix} 1 & \boldsymbol{0}^T \\ \boldsymbol{0} & \boldsymbol{Q}(k-1) \end{bmatrix}}_{\boldsymbol{I}} \underbrace{\begin{bmatrix} \boldsymbol{x}^H(k) \\ \lambda^{1/2}\boldsymbol{X}^H(k-1) \end{bmatrix}}_{\boldsymbol{X}^H(k)} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{U}(k) \end{bmatrix}. \tag{2.33}$$

Designando o produto das duas primeiras matrizes na EQ. 2.33 por $\tilde{\boldsymbol{Q}}(k)$, tem-se

$$\tilde{\boldsymbol{Q}}(k) \begin{bmatrix} \boldsymbol{x}^H(k) \\ \boldsymbol{0} \\ \lambda^{1/2}\boldsymbol{U}(k-1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{U}(k) \end{bmatrix}. \tag{2.34}$$

Da EQ. 2.34, pode-se concluir que o produto de rotações de Givens em $\tilde{\boldsymbol{Q}}(k)$ são responsáveis, apenas, por zerar o vetor de entrada no instante k sobre $\lambda^{1/2}\boldsymbol{U}(k-1)$. Ainda da EQ. 2.33, pode ser obtida uma expressão recursiva para $\boldsymbol{Q}(k)$ como na equação seguinte.

$$\boldsymbol{Q}(k) = \tilde{\boldsymbol{Q}}(k) \begin{bmatrix} 1 & \boldsymbol{0}^T \\ \boldsymbol{0} & \boldsymbol{Q}(k-1) \end{bmatrix} \tag{2.35}$$

A EQ. 2.34 sugere um aumento na ordem a cada iteração uma vez que a estrutura de $\tilde{\boldsymbol{Q}}(k)$ inclui uma sub-matriz $\boldsymbol{I}_{k-N}$. Mas isso pode ser evitado se essa sub-matriz for excluída da estrutura de $\tilde{\boldsymbol{Q}}(k)$, reescrevendo a EQ. 2.34 da seguinte forma

$$\begin{bmatrix} \mathbf{0}^T \\ \boldsymbol{U}(k) \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} \boldsymbol{x}^H(k) \\ \lambda^{1/2}\boldsymbol{U}(k-1) \end{bmatrix}, \tag{2.36}$$

onde $\boldsymbol{Q}_\theta(k) = \prod_{i=N}^{1} \boldsymbol{Q}_{\theta_i}(k)$ é uma matriz de ordem fixa obtida de $\tilde{\boldsymbol{Q}}(k)$ após a remoção da sub-matriz $\boldsymbol{I}_{k-N}$.[3]

Recordando a EQ. 2.31, pode-se observar que $\boldsymbol{e}_{q1}(k) = \boldsymbol{d}_{q1}(k)$. Logo, o produto $\boldsymbol{Q}(k)\boldsymbol{d}^*(k)$ pode ser escrito como

$$\begin{aligned} \begin{bmatrix} \boldsymbol{e}_{q1}(k) \\ \boldsymbol{d}_{q2}(k) \end{bmatrix} &= \tilde{\boldsymbol{Q}}(k) \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}(k-1) \end{bmatrix} \begin{bmatrix} d^*(k) \\ \lambda^{1/2}\boldsymbol{d}^*(k-1) \end{bmatrix} \\ &= \tilde{\boldsymbol{Q}}(k) \begin{bmatrix} d^*(k) \\ \lambda^{1/2}\boldsymbol{e}_{q1}(k-1) \\ \lambda^{1/2}\boldsymbol{d}_{q2}(k-1) \end{bmatrix} \\ &= \begin{bmatrix} e_{q1}(k) \\ \lambda^{1/2}\boldsymbol{e}_{q1}(k-1) \\ \boldsymbol{d}_{q2}(k) \end{bmatrix}. \end{aligned} \tag{2.37}$$

Na última multiplicação da equação acima, levou-se em consideração o fato de que $\tilde{\boldsymbol{Q}}(k)$ altera apenas o primeiro e os últimos N elementos do vetor.

À semelhança do que foi feito anteriormente, a seção crescente de $\tilde{\boldsymbol{Q}}(k)$ na EQ. 2.37 pode ser removida resultado em

$$\begin{bmatrix} e_{q1}(k) \\ \boldsymbol{d}_{q2}(k) \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} d^*(k) \\ \lambda^{1/2}\boldsymbol{d}_{q2}(k-1) \end{bmatrix}, \tag{2.38}$$

onde $e_{q1}(k)$ é o primeiro elemento do vetor de erro *rotacionado*, $\boldsymbol{e}_{q1}(k)$, e $\boldsymbol{d}_{q2}(k)$ é um vetor com os últimos N elementos do vetor sinal desejado *rotacionado*. O processo na EQ. 2.38 é referido na literatura como *estimação conjunta*.

Para ter todas as equações do algoritmo QRD-RLS convencional, é necessária ainda uma expressão para a atualização do vetor erro. Para obter-se tal expressão, é necessário

[3] $\boldsymbol{Q}_\theta(k) = \boldsymbol{Q}_{\theta_N}(k) \cdots \boldsymbol{Q}_{\theta_2}(k)\boldsymbol{Q}_{\theta_1}(k)$.

TAB. 2.3: Equações do algoritmo QRD-RLS convencional.

QRD-RLS
para cada k { Obtendo $\boldsymbol{Q}_\theta(k)$ e atualizando $\boldsymbol{U}(k)$: $\begin{bmatrix} \mathbf{0}^T \\ \boldsymbol{U}(k) \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} \boldsymbol{x}^H(k) \\ \lambda^{1/2}\boldsymbol{U}(k-1) \end{bmatrix}$ Obtendo $\gamma(k)$: $\gamma(k) = \prod_{i=0}^{N} \cos\theta_i(k)$ Obtendo $e_{q1}(k)$ e atualizando $\boldsymbol{d}_{q2}(k)$: $\begin{bmatrix} e_{q1}(k) \\ \boldsymbol{d}_{q2}(k) \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} d^*(k) \\ \lambda^{1/2}\boldsymbol{d}_{q2}(k-1) \end{bmatrix}$ Obtendo o erro *a priori*: $\varepsilon(k) = e^*_{q1}(k)/\gamma(k)$ }

entender a estrutura da matriz $\boldsymbol{Q}_\theta(k)$ que depende do tipo de triangularização (inferior ou superior) a ser obtida para $\boldsymbol{U}(k)$. Mas, independentemente dessa triangularização, ela pode ser particionada como (ver Apêndice 1)

$$\boldsymbol{Q}_\theta(k) = \begin{bmatrix} \gamma(k) & \boldsymbol{g}^H(k) \\ \boldsymbol{f}(k) & \boldsymbol{E}(k) \end{bmatrix}, \tag{2.39}$$

onde $\gamma(k) = \prod_{i=1}^{N} \cos\theta_i(k)$ e as partições $\boldsymbol{f}(k)$, $\boldsymbol{g}(k)$ e $\boldsymbol{E}(k)$ dependem do tipo de triangularização e serão abordadas mais detalhadamente nas seções que se seguem.

Para obter a última expressão do algoritmo QRD-RLS convencional, basta multiplicar o vetor transposto $\boldsymbol{e}_q^H(k)\boldsymbol{Q}(k)$ pelo vetor $[1 \quad 0 \quad \cdots \quad 0]^T$, como segue.

$$\boldsymbol{e}_q^H(k)\boldsymbol{Q}(k) \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \boldsymbol{e}^H(k)\boldsymbol{Q}^H(k)\boldsymbol{Q}(k) \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = e(k) \tag{2.40}$$

sendo $e(k)$ o erro *a posteriori* (aquele calculado após a atualização do vetor de coeficientes).

Das EQ. 2.35 e EQ. 2.39, e usando o fato de $\boldsymbol{Q}_\theta(k)$ ser igual a $\tilde{\boldsymbol{Q}}(k)$ após a remoção das $k - N$ linhas e colunas crescentes, a EQ. 2.40 pode ser reescrita como

$$e(k) = \boldsymbol{e}_q^H(k)\tilde{\boldsymbol{Q}}^H(k)\begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}(k-1) \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{2.41}$$

$$= \begin{bmatrix} \boldsymbol{e}_{q1}^H(k) & \boldsymbol{e}_{q2}^H(k) \end{bmatrix}\begin{bmatrix} \gamma(k) & \mathbf{0} & \boldsymbol{g}^H(k) \\ 0 & \boldsymbol{I}_{k-N-1} & 0 \\ \boldsymbol{f}(k) & \mathbf{0} & \boldsymbol{E}(k) \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{2.42}$$

$$= \begin{bmatrix} \boldsymbol{e}_{q1}^H(k) & \boldsymbol{e}_{q2}^H(k) \end{bmatrix}\begin{bmatrix} \gamma(k) \\ 0 \\ \vdots \\ 0 \\ \boldsymbol{f}(k) \end{bmatrix}. \tag{2.43}$$

Finalmente, pelo fato de $\boldsymbol{w}(k)$ ser escolhido tal que $\boldsymbol{e}_{q2}(k) = 0$, pode-se concluir da equação acima e da EQ. 2.40 que o erro *a posteriori* é dado por

$$e(k) = e_{q1}^*(k)\gamma(k). \tag{2.44}$$

Verifica-se que o erro *a priori* se relaciona com o erro *a posteriori* (DINIZ, 2002) de acordo com

$$\varepsilon(k) = e(k)/\gamma^2(k) = e_{q1}^*(k)/\gamma(k). \tag{2.45}$$

A EQ. 2.44 deixa claro que não é necessário o cálculo do vetor de coeficientes, $\boldsymbol{w}(k)$, para a obtenção do erro *a priori*, $\varepsilon(k)$, para esta família de algoritmos. As equações do algoritmo QRD-RLS convencional estão resumidas na TAB. 2.3.

Considerações Sobre a Obtenção da Matriz $\boldsymbol{Q}_\theta$

A matriz $\boldsymbol{Q}_\theta(k)$ na EQ. 2.36 contém os parâmetros, senos e cossenos, das rotações de Givens responsáveis por rotacionar os elementos do vetor $\boldsymbol{x}(k)$ sobre a diagonal de uma matriz triangular inferior. Esses parâmetros são calculados de acordo com o elemento a ser rotacionado.

Exemplo: Seja o vetor de números complexos $\boldsymbol{z} = [a \quad b]^T$. Deseja-se obter os parâmetros necessário para rotacionar o elemento a (anulando-o) sobre o b de tal forma que a norma de $\boldsymbol{z}$ seja mantida.

Para um caso como este, $\boldsymbol{Q}_\theta$ teria a seguinte estrutura:

$$\boldsymbol{Q}_\theta = \begin{bmatrix} \cos\theta & -\sin^*\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \tag{2.46}$$

onde o *asterisco* denota o complexo conjugado; $\cos\theta$ e $\sin\theta$ são dados por

$$\cos\theta = \frac{|b|}{\sqrt{|a|^2 + |b|^2}} \tag{2.47}$$

e

$$\sin\theta = \left[\frac{a\cos\theta}{b}\right]^*. \tag{2.48}$$

A operação completa fica como segue.

$$\begin{aligned} \begin{bmatrix} 0 \\ r \end{bmatrix} &= \boldsymbol{Q}_\theta \begin{bmatrix} a \\ b \end{bmatrix} \\ &= \begin{bmatrix} \cos\theta & -\sin^*\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}, \end{aligned} \tag{2.49}$$

onde $r = \sqrt{|a|^2 + |b|^2}$.

Se $\boldsymbol{z}$ for real, $\cos\theta$ e $\sin\theta$ podem ser calculados usando simplesmente as seguintes relações:

$$\cos\theta = \frac{b}{r} \tag{2.50}$$

e

$$\sin\theta = \frac{a}{r}. \tag{2.51}$$

Para o caso da EQ. 2.36, o procedimento acima é repetido o número de vezes necessário para anular todos os elementos de $\boldsymbol{x}(k)$. Este tópico encontra-se mais detalhado no Apêndice 1.

2.4.2 O ALGORITMO QRD-RLS INVERSO

O algoritmo QRD-RLS Inverso, proposto em (ALEXANDER, 1993), é baseado na atualização da inversa do fator de Cholesky, $\boldsymbol{U}^{-1}(k-1)$. A vantagem desse algoritmo em relação ao QRD-RLS convencional é que ele obtém o vetor de coeficientes sem a necessidade do procedimento de substituição regressiva. As equações básicas deste algoritmo serão vistas de forma resumida nesta seção.

Começando pela solução de Wiener, $\boldsymbol{w}(k) = \boldsymbol{R}^{-1}(k)\boldsymbol{p}(k)$, e fazendo uso de $[\boldsymbol{x}(k) \quad \lambda^{1/2}\boldsymbol{X}(k-1)]$ ou invés de $\boldsymbol{X}(k)$ nas definições de $\boldsymbol{R}(k)$ e $\boldsymbol{p}(k)$, é possível mostrar que (vide EQ. 2.28 com $\boldsymbol{R}(k) = \boldsymbol{U}^H(k)\boldsymbol{U}(k)$)

$$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) + \boldsymbol{U}^{-1}(k)\boldsymbol{U}^{-H}(k)\boldsymbol{x}(k)\varepsilon^*(k), \tag{2.52}$$

onde o vetor

$$\boldsymbol{U}^{-1}(k)\boldsymbol{U}^{-H}(k)\boldsymbol{x}(k) = \boldsymbol{R}^{-1}(k)\boldsymbol{x}(k)$$

é conhecido como ganho de Kalman.

O vetor de coeficientes é atualizado por (ALEXANDER, 1993)

$$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) - \gamma(k)\boldsymbol{u}(k)\varepsilon^*(k), \tag{2.53}$$

onde $\boldsymbol{u}(k) = -\lambda^{-1/2}\gamma(k)\boldsymbol{U}^{-1}(k-1)\boldsymbol{a}(k)$ com $\boldsymbol{a}(k) = \lambda^{-1/2}\boldsymbol{U}^{-H}(k-1)\boldsymbol{x}(k)$.

As equações do algoritmo IQRD-RLS, nas suas formas matriciais, estão resumidas na TAB. 2.4. Este algoritmo, bem com o QRD-RLS convencional, possui uma complexidade computacional de ordem $O[N^2]$ (GHIRNIKAR, 1992; ALEXANDER, 1993; DINIZ, 2002).

2.4.3 INTERPRETANDO AS VARIÁVEIS INTERNAS

Como visto anteriormente, a matriz $\boldsymbol{Q}_\theta(k)$ pode ser particionada como

$$\boldsymbol{Q}_\theta(k) = \begin{bmatrix} \gamma(k) & \boldsymbol{g}^H(k) \\ \boldsymbol{f}(k) & \boldsymbol{E}(k) \end{bmatrix} \tag{2.54}$$

onde $\gamma(k) = \prod_{i=1}^{N} \cos\theta_i(k)$; $\mathbf{E}(k)$ não possui nenhum significado relevante de interesse; os significados das partições $\boldsymbol{f}(k)$ e $\mathbf{g}(k)$ serão vistos na discussão que se segue. Essas partições dependem do tipo de triangularização a ser utilizada– no nosso caso, triangularização inferior, correspondendo a erros de predição *backward* (APOLINÁRIO JR., 1997).

TAB. 2.4: Equações do algoritmo QRD-RLS Inverso.

IQRD-RLS
para cada k
{ Obtendo $\boldsymbol{a}(k)$:
$\boldsymbol{a}(k) = \lambda^{-1/2}\boldsymbol{U}^{-H}(k-1)\boldsymbol{x}(k)$
Obtendo $\boldsymbol{Q}_\theta(k)$ e $\gamma(k)$:
$\begin{bmatrix} 1/\gamma(k) \\ \mathbf{0} \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} 1 \\ -\boldsymbol{a}(k) \end{bmatrix}$
Obtendo $\boldsymbol{u}(k)$ e atualizando $\boldsymbol{U}^{-H}(k)$:
$\begin{bmatrix} \boldsymbol{u}^H(k) \\ \boldsymbol{U}^{-H}(k) \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} \mathbf{0}^T \\ \lambda^{-1/2}\boldsymbol{U}^{-H}(k-1) \end{bmatrix}$
Obtendo $e(k)$, o erro a *priori*:
$\varepsilon(k) = d(k) - \boldsymbol{w}^H(k-1)\boldsymbol{x}(k)$
Atualizando o vetor de coeficientes:
$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) - \gamma(k)\boldsymbol{u}(k)\varepsilon^*(k)$
}

Da EQ. 2.36 e da EQ. 2.54, é possível escrever

$$\begin{aligned} \begin{bmatrix} \mathbf{0}^T \\ \boldsymbol{U}(k) \end{bmatrix} &= \boldsymbol{Q}_\theta(k) \begin{bmatrix} \boldsymbol{x}^H(k) \\ \lambda^{1/2}\boldsymbol{U}(k-1) \end{bmatrix} \\ &= \begin{bmatrix} \gamma(k) & \boldsymbol{g}^H(k) \\ \boldsymbol{f}(k) & \boldsymbol{E}(k) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}^H(k) \\ \lambda^{1/2}\boldsymbol{U}(k-1) \end{bmatrix} \end{aligned} \tag{2.55}$$

Sabe-se que $\boldsymbol{Q}_\theta(k)$ é uma matriz ortogonal. Logo,

$$\begin{aligned} \boldsymbol{I}_{N+1} &= \boldsymbol{Q}_\theta(k)\boldsymbol{Q}_\theta^H(k) = \boldsymbol{Q}_\theta^H(k)\boldsymbol{Q}_\theta(k) \\ &= \begin{bmatrix} \gamma(k) & \boldsymbol{g}^H(k) \\ \boldsymbol{f}(k) & \boldsymbol{E}(k) \end{bmatrix} \begin{bmatrix} \gamma(k) & \boldsymbol{f}^H(k) \\ \boldsymbol{g}(k) & \boldsymbol{E}^H(k) \end{bmatrix} \\ &= \begin{bmatrix} \gamma(k) & \boldsymbol{f}^H(k) \\ \boldsymbol{g}(k) & \boldsymbol{E}^H(k) \end{bmatrix} \begin{bmatrix} \gamma(k) & \boldsymbol{g}^H(k) \\ \boldsymbol{f}(k) & \boldsymbol{E}(k) \end{bmatrix} \end{aligned} \tag{2.56}$$

As seguintes relações podem ser obtidas das EQ. 2.55 e EQ. 2.56.

$$\boldsymbol{f}(k)\boldsymbol{x}^H(k) + \lambda^{1/2}\boldsymbol{E}(k)\boldsymbol{U}(k-1) = \boldsymbol{U}(k) \tag{2.57}$$

e

$$\gamma(k)\boldsymbol{f}(k) + \boldsymbol{E}(k)\boldsymbol{g}(k) = \boldsymbol{0} \tag{2.58}$$

Da EQ. 2.55, vê-se que $\boldsymbol{U}(k)$ é o fator de Cholesky de

$$\left[\begin{array}{cc} \boldsymbol{x}(k) & \lambda^{1/2}\boldsymbol{U}^H(k-1) \end{array}\right] \left[\begin{array}{c} \boldsymbol{x}^H(k) \\ \lambda^{1/2}\boldsymbol{U}(k-1) \end{array}\right].$$

Assim sendo, pré-multiplicando EQ. 2.55 pela sua transposta obtém-se

$$\boldsymbol{U}^H(k)\boldsymbol{U}(k) = \boldsymbol{x}(k)\boldsymbol{x}^H(k) + \lambda\boldsymbol{U}^H(k-1)\boldsymbol{U}(k-1). \tag{2.59}$$

Pré-multiplicando a EQ. 2.57 por $\boldsymbol{U}^H(k)$ e comparando o resultado com a EQ. 2.59, chega-se às duas relações seguintes.

$$\boldsymbol{f}(k) = \boldsymbol{U}^{-H}(k)\boldsymbol{x}(k) \tag{2.60}$$

$$\boldsymbol{E}(k) = \lambda^{1/2}\boldsymbol{U}^{-H}(k)\boldsymbol{U}^H(k-1) \tag{2.61}$$

Substituindo as EQ. 2.60 e EQ. 2.61 na EQ. 2.58 obtém-se

$$\begin{aligned} \boldsymbol{g}(k) &= -\gamma(k)\boldsymbol{U}^{-H}(k-1)\boldsymbol{x}(k)/\sqrt{\lambda} \\ &= -\gamma(k)\boldsymbol{a}(k) \end{aligned} \tag{2.62}$$

As últimas três equações são importantes para o entendimento dos algoritmos rápidos da família QR.

2.4.4 O PROBLEMA DAS PREDIÇÕES PROGRESSIVA E REGRESSIVA

As soluções dos problemas da predição progressiva e regressiva são úteis na derivação dos algoritmos QRD-RLS rápidos que são introduzidos na Seção 2.4.5. Por questão de simplificação, será assumido uma vez mais o caso monocanal. Essa discrição pode ser expandida para contemplar o caso multicanal a ser abordado no Capítulo 5.

2.4.4.1 A PREDIÇÃO PROGRESSIVA

Na predição progressiva ou *forward*, o sinal desejado é $x(k)$ e a predição será realizada, para um determinado instante k, a partir do vetor de entrada num instante anterior, $\boldsymbol{x}(k-1)$. O vetor erro de predição progressiva, $\boldsymbol{e}_f(k)$, pode ser expresso como

$$\begin{aligned}\boldsymbol{e}_f(k) &= \boldsymbol{d}_f(k) - \begin{bmatrix} \boldsymbol{X}^T(k-1) \\ \mathbf{0}^T \end{bmatrix} \boldsymbol{w}_f^*(k) \\ &= \begin{bmatrix} x(k) \\ \lambda^{1/2}x(k-1) \\ \vdots \\ \lambda^{k/2}x(0) \end{bmatrix} - \begin{bmatrix} \boldsymbol{X}^T(k-1) \\ \mathbf{0}^T \end{bmatrix} \boldsymbol{w}_f^*(k), \end{aligned} \tag{2.63}$$

onde $\boldsymbol{w}_f(k)$ é o vetor de coeficientes da predição progressiva.

A EQ. 2.63 pode ser escrita em termos de $\boldsymbol{X}^{(N+1)}(k)$, a matriz de informação de entrada de ordem $N+1$.

$$\boldsymbol{e}_f(k) = \begin{bmatrix} \boldsymbol{d}_f(k) & \boldsymbol{X}^T(k-1) \\ & \mathbf{0}^T \end{bmatrix} \begin{bmatrix} 1 \\ -\boldsymbol{w}_f^*(k) \end{bmatrix} = \boldsymbol{X}^{T_{(N+1)}}(k) \begin{bmatrix} 1 \\ -\boldsymbol{w}_f^*(k) \end{bmatrix} \tag{2.64}$$

O vetor erro de predição progressiva rotacionado é definido como

$$\boldsymbol{e}_{f_q}(k) = \begin{bmatrix} \boldsymbol{Q}(k-1) & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \boldsymbol{e}_f^*(k) = \begin{bmatrix} \boldsymbol{e}_{f_{q1}}(k) & \mathbf{0} \\ \boldsymbol{d}_{f_{q2}}(k) & \boldsymbol{U}(k-1) \\ \lambda^{1/2}x(0) & \mathbf{0}^T \end{bmatrix} \begin{bmatrix} 1 \\ -\boldsymbol{w}_f(k) \end{bmatrix}. \tag{2.65}$$

2.4.4.2 A PREDIÇÃO REGRESSIVA

No problema da predição regressiva ou *backward*, o objetivo é obter-se, num determinado instante k, uma estimativa de uma amostra passada de uma determinada seqüencia de entrada a partir da informação disponível naquele instante. Assim, o sinal desejado é $x(k-N)$ e a predição é feita com base no vetor $\boldsymbol{x}(k)$. O vetor de erros regressivos ponderado é dado por

$$\boldsymbol{e}_b(k) = \boldsymbol{d}_b(k) - \boldsymbol{X}^T(k)\boldsymbol{w}_b^*(k) = \begin{bmatrix} x(k-N) \\ \lambda^{1/2}x(k-N-1) \\ \vdots \\ \lambda^{(k-N)/2}x(0) \\ \mathbf{0}_N \end{bmatrix} - \boldsymbol{X}^T(k)\boldsymbol{w}_b^*(k), \tag{2.66}$$

onde $\boldsymbol{w}_b(k)$ é o vetor de coeficientes da predição regressiva.

A EQ. 2.66 pode ser reescrita em função de $\boldsymbol{X}^{(N+1)}(k)$, a matriz de informação de ordem $N+1$.

$$\boldsymbol{e}_b(k) = \begin{bmatrix} \boldsymbol{X}^T(k) & \boldsymbol{d}_b(k) \end{bmatrix} \begin{bmatrix} -\boldsymbol{w}_b^*(k) \\ 1 \end{bmatrix} = \boldsymbol{X}^{T_{(N+1)}}(k) \begin{bmatrix} -\boldsymbol{w}_b^*(k) \\ 1 \end{bmatrix} \tag{2.67}$$

O vetor de erro de predição regressiva rotacionado é definido como:

$$\boldsymbol{e}_{b_q} = \boldsymbol{Q}(k)\boldsymbol{e}_b^*(k) = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{e}_{b_{q1}}(k) \\ \boldsymbol{U}(k) & \boldsymbol{d}_{b_{q2}}(k) \end{bmatrix} \begin{bmatrix} -\boldsymbol{w}_b(k) \\ 1 \end{bmatrix}. \tag{2.68}$$

2.4.5 ALGORITMOS QRD-RLS RÁPIDOS

Usando as soluções das predições progressiva e regressiva, alguns algoritmos denominados de QRD-RLS rápidos foram desenvolvidos. Estes algoritmos podem ser classificados de acordo com o tipo de triangularização aplicada à matriz de informação (superior ou inferior) e com o tipo de vetor de erro utilizado no processo de adaptação (*a priori* ou *a posteriori*), como será visto a seguir.

Usando a ortogonalização de Gram-Schmidt para $\boldsymbol{U}(k)$ triangular inferior, juntamente com a solução do problema da predição regressiva, é possível demonstrar que $\boldsymbol{f}(k)$ pode ser expresso por

$$\boldsymbol{f}(k) = \begin{bmatrix} e_b^{(N)}(k)/\|\boldsymbol{e}_b^{(N)}(k)\| \\ e_b^{(N-1)}(k)/\|\boldsymbol{e}_b^{(N-1)}(k)\| \\ \vdots \\ e_b^{(0)}(k)/\|\boldsymbol{e}_b^{(0)}(k)\| \end{bmatrix}, \tag{2.69}$$

onde $\boldsymbol{e}_b$ é o vetor de erros de predição regressiva *a posteriori*. Conseqüentemente, $\boldsymbol{f}(k)$ é referido como o vetor de erros de predição regressiva *a posteriori* normalizado no instante k.

Aplicando o mesmo procedimento anterior, pode-se demonstrar também que

TAB. 2.5: Classificação dos algoritmos QRD rápidos (FQRD)

Tipo de erro	Predição	
	Progressiva (*Forward*)	Regressiva (*Backward*)
A Posteriori	**FQRD_POS_F**	**FQRD_POS_B**
A Priori	**FQRD_PRI_F**	**FQRD_PRI_B**

$$\boldsymbol{g}(k) = -\gamma(k)\boldsymbol{a}(k) = -\gamma(k)\lambda^{-1/2} \begin{bmatrix} e_b^{\prime(N)}(k)/\|\boldsymbol{e}_b^{(N)}(k)\| \\ e_b^{\prime(N-1)}(k)/\|\boldsymbol{e}_b^{(N-1)}(k)\| \\ \vdots \\ e_b^{\prime(0)}(k)/\|\boldsymbol{e}_b^{(0)}(k)\| \end{bmatrix}, \tag{2.70}$$

onde $\boldsymbol{e}_b'(k)$ corresponde ao vetor de erros de predição regressiva *a priori* e $\|\cdot\|$ denota norma. Assim, $\boldsymbol{a}(k)$ é referido com o vetor de erros de predição regressiva *a priori* normalizado pelas energias do vetor de erros de predição regressiva *a posteriori* e ponderado por $\lambda^{-1/2}$.

A atualização de $\boldsymbol{a}(k)$ ou de $\boldsymbol{f}(k)$ determina a classificação do algoritmo quanto ao tipo de erro (*a priori* ou *a posteriori*, respectivamente) e a triangularização aplicada à matriz de informação (inferior ou superior) determina a classificação quanto ao tipo de predição (regressiva ou progressiva, respectivamente) – vide TAB. 2.5 (APOLINÁRIO JR., 1997).

Dessa família de algoritmos rápidos, sabe-se que aqueles que usam a atualização do vetor de erro de predição progressiva (*forward*), o **FQR_POS_F**, proposto em (CIOFFI, 1990), e **FQR_PRI_F**, proposto em (APOLINÁRIO JR., 1997), apresentam problemas de instabilidade (APOLINÁRIO JR., 1998). O **FQRD_POS_B** e o **FQRD_PRI_B** foram propostos em (REGALIA, 1991; APOLINÁRIO JR., 2000a) e em (MIRANDA, 1995; RONTOGIANNIS, 1995) (versões distintas), respectivamente. A principal vantagem desta classe de algoritmos é uma redução na ordem da complexidade computacional em relação ao QRD-RLS convencional e ao QRD-RLS Inverso de $O[N^2]$ para $O[N]$.

As versões multicanal destes algoritmos podem ser consideradas como generalizações dos monocanais e serão abordadas detalhadamente no Capítulo 5.

2.5 RESUMO

Neste capítulo, foi feita uma introdução à Filtragem Adaptativa. Algoritmos básicos como o LMS e o RLS foram revistos. Foi feita, também, uma introdução aos algoritmos baseados no RLS que usam a decomposição QR. Foram igualmente abordados, de forma resumida, alguns tópicos relevantes ao entendimento dos algoritmos da família QRD-RLS rápidos a serem detalhados no Capítulo 5.

3.1 INTRODUÇÃO

Algoritmos adaptativos com restrições são aplicados em várias áreas de comunicações e processamento de sinais incluindo *beamforming*, estimação espectral e detecção multi–usuário para sistemas de comunicações, entre outras. Neste tipo de aplicação, normalmente estamos interessados que o vetor de coeficientes $\boldsymbol{w}$ satisfaça um sistema de equações lineares. Este tipo de solução é conhecido pela sua sigla em inglês **LCAF** (*Linearly Constrained Adaptive Filtering*).

Um algoritmo muito simples e robusto, incorporando restrições na solução, foi originalmente introduzido em (FROST III, 1972). Este algoritmo, conhecido como *Constrained Least Mean Square* (CLMS), bem como o seu correspondente sem restrições, o algoritmo LMS, visto no Capítulo 2 (Seção 2.2), tem uma séria desvantagem no que diz respeito à velocidade de convergência: é muito lenta. Tentando solucionar este problema, vários algoritmos do tipo RLS com restrições (*Constrained Recursive Least Squares*–CRLS) foram propostos recentemente. Em função da reconhecida instabilidade numérica do algoritmo RLS convencional, alguns dos algoritmos com restrições baseados nele usam fatores de correção (RESENDE, 1996) ou rotações numericamente estáveis (CHERN, 2002) para tentar garantir a estabilidade.

Neste capítulo, abordaremos de forma resumida os conceitos básicos de filtragem adaptativa com restrições lineares. Na Seção 3.2 é definido o problema da filtragem adaptativa com restrições lineares. O Algoritmo RLS com restrições é revisto na Seção 3.3. Na Seção 3.4 são revistas duas estruturas alternativas ao uso de algoritmos com restrições: a estrutura GSC (*Generalized Sidelobe Canceller*) (GRIFFITHS, 1982) e a estrutura Householder Constrained (DE CAMPOS, 2002).

3.2 DEFINIÇÃO DO PROBLEMA

O problema da filtragem adaptativa sujeita à imposição de restrições lineares pode ser expresso da seguinte maneira:

$$\underset{\boldsymbol{w}}{\text{minimizar}} \quad E\left[e(k)e^*(k)\right] \quad \text{sujeito a} \quad \boldsymbol{C}^H\boldsymbol{w} = \boldsymbol{f}, \tag{3.1}$$

onde $e(k) = d(k) - \boldsymbol{w}^H \boldsymbol{x}(k)$ corresponde ao erro no instante de tempo k, $\boldsymbol{C}$ é a matriz de restrições e $\boldsymbol{f}$ é o vetor de ganho.

Para o caso de um filtro adaptativo usado para processar M sinais captados através de um arranjo (*array*) de M sensores, como na FIG. 2.3, a saída pode ser expressa como na EQ. 2.4. Seja p o número de restrições lineares; neste caso particular em que $\boldsymbol{w}(k)$ (EQ. 2.5) e $\boldsymbol{f}$ são vetores de tamanho $MN \times 1$ e $p \times 1$, respectivamente, $\boldsymbol{C}$ possui dimensões $MN \times p$.

Em muitas situações de interesse prático, $d(k) = 0$. Nessas situações, o filtro adaptativo é conhecido como **LCMV** (do termo em inglês *Linearly-Constrained Minimum-Variance*). Outra situação particular de interesse ocorre quando $d(k) = 0$ e $\mathbf{f} = 1$, sendo, nesta situação, o filtro conhecido como **MVDR** (do termo em Inglês *Minimum Variance Distortionless Response*).

A solução ótima do problema apresentado na EQ. 3.1 para os coeficientes do filtro adaptativo com restrições, $\boldsymbol{w}_{opt}$, também conhecida como solução de Wiener, é obtida usando multiplicadores de Lagrange (FROST III, 1972) e (HAYKIN, 1996). Considerando o caso geral, isto é, $d(k) \neq 0$, uma nova função objetivo pode ser definida como

$$\xi(k) = E\left[e(k)e^*(k)\right] + \boldsymbol{\mathcal{L}}_R^T \Re\left[\boldsymbol{C}^H \boldsymbol{w} - \boldsymbol{f}\right] + \boldsymbol{\mathcal{L}}_I^T \Im\left[\boldsymbol{C}^H \boldsymbol{w} - \boldsymbol{f}\right], \tag{3.2}$$

onde os operados $\Re[\cdot]$ e $\Im[\cdot]$ correspondem, respectivamente, às partes reais e imaginárias de um número complexo.

Na obtenção da expressão acima foi levada em consideração a intenção de se minimizar um valor real. Agora, seja $\boldsymbol{\mathcal{L}} = \boldsymbol{\mathcal{L}}_R + j\boldsymbol{\mathcal{L}}_I$, e sabendo que $\Re[z] = \frac{z}{2} + \frac{z^*}{2}$, pode-se reescrever a EQ. 3.2 como segue.

$$\begin{aligned} \xi(k) &= E\left[e(k)e^*(k)\right] + \Re\left[\boldsymbol{\mathcal{L}}^H(\boldsymbol{C}^H\boldsymbol{w} - \boldsymbol{f})\right] \\ &= E\left[e(k)e^*(k)\right] + \frac{1}{2}\boldsymbol{\mathcal{L}}^H(\boldsymbol{C}^H\boldsymbol{w} - \boldsymbol{f}) + \frac{1}{2}\boldsymbol{\mathcal{L}}^T(\boldsymbol{w}^H\boldsymbol{C} - \boldsymbol{f}^*) \end{aligned} \tag{3.3}$$

Para encontrar a solução ótima, basta calcular o gradiente de $\xi(k)$, $\nabla_w \xi(k)$, e igualá-lo a zero.

$$\begin{aligned} \nabla_w \xi(k) &= \nabla_w \left[E\left[e(k)e^*(k)\right] + \frac{1}{2}\boldsymbol{\mathcal{L}}^H(\boldsymbol{C}^H\boldsymbol{w} - \boldsymbol{f}) + \frac{1}{2}\boldsymbol{\mathcal{L}}^T(\boldsymbol{w}^H\boldsymbol{C} - \boldsymbol{f}^*)\right] \\ &= E\left[-2\boldsymbol{x}(k)e^*(k)\right] + \mathbf{0} + \boldsymbol{C}\boldsymbol{\mathcal{L}} \\ &= E\left[-2\boldsymbol{x}(k)d^*(k)\right] + 2E\left[\boldsymbol{x}(k)\boldsymbol{x}^H(k)\right]\boldsymbol{w} + \boldsymbol{C}\boldsymbol{\mathcal{L}} \end{aligned} \tag{3.4}$$

Onde foi levado em consideração que $e(k) = d(k) - \boldsymbol{w}^H \boldsymbol{x}(k)$.

A expressão acima pode ser igualada a zero, levando em consideração as definições da matriz de autocorrelação $\boldsymbol{R} = E[\boldsymbol{x}(k)\boldsymbol{x}^H(k)]$ e do vetor de correlação cruzada $\boldsymbol{p} = E[d^*(k)\boldsymbol{x}(k)]$, como segue.

$$-2\boldsymbol{p} + 2\boldsymbol{R}\boldsymbol{w} + \boldsymbol{C}\boldsymbol{\mathcal{L}} = \boldsymbol{0} \tag{3.5}$$

Resolvendo a última equação em relação a $\boldsymbol{w}$ obtém-se

$$\boldsymbol{w} = \frac{1}{2}\boldsymbol{R}^{-1}(2\boldsymbol{p} - \boldsymbol{C}\boldsymbol{\mathcal{L}}). \tag{3.6}$$

Levando em consideração que $\boldsymbol{C}^H\boldsymbol{w} = \boldsymbol{f}$, pode-se pré-multiplicar a EQ. 3.6 por $\boldsymbol{C}^H$ e chegar à expressão seguinte para $\boldsymbol{\mathcal{L}}$.

$$\boldsymbol{\mathcal{L}} = 2(\boldsymbol{C}^H\boldsymbol{R}^{-1}\boldsymbol{C})^{-1}(\boldsymbol{C}^H\boldsymbol{R}^{-1}\boldsymbol{p} - \boldsymbol{f}) \tag{3.7}$$

Finalmente, substituindo a EQ. 3.7 na EQ. 3.6, obtém-se a solução ótima para o vetor de coeficientes que minimiza $E[e(k)e^*(k)]$ sujeito a $\boldsymbol{C}^H\boldsymbol{w} = \boldsymbol{f}$.

$$\boldsymbol{w}_{opt} = \boldsymbol{R}^{-1}\boldsymbol{p} + \boldsymbol{R}^{-1}\boldsymbol{C}(\boldsymbol{C}^H\boldsymbol{R}^{-1}\boldsymbol{C})^{-1}(\boldsymbol{f} - \boldsymbol{C}^H\boldsymbol{R}^{-1}\boldsymbol{p}) \tag{3.8}$$

A EQ. 3.8 é conhecida como a solução de Wiener para a problemas com restrições lineares. É fácil se observar que esta mesma equação pode ser vista como a soma de duas parcelas: a primeira, $\boldsymbol{w}_{sr} = \boldsymbol{R}^{-1}\boldsymbol{p}$, corresponde à solução de Wiener sem restrição e a segunda, $\boldsymbol{w}_{cr} = \boldsymbol{R}^{-1}\boldsymbol{C}(\boldsymbol{C}^H\boldsymbol{R}^{-1}\boldsymbol{C})^{-1}(\boldsymbol{f} - \boldsymbol{C}^H\boldsymbol{R}^{-1}\boldsymbol{p})$, é a parcela responsável por fazer com que a solução satisfaça as restrições inicialmente impostas. Esta última parcela pode ser ainda escrita como $\boldsymbol{w}_{cr} = \boldsymbol{R}^{-1}\boldsymbol{C}(\boldsymbol{C}^H\boldsymbol{R}^{-1}\boldsymbol{C})^{-1}(\boldsymbol{f} - \boldsymbol{C}^H\boldsymbol{w}_{sr})$. Assim, a EQ. 3.8 pode ser reescrita de forma compacta como

$$\boldsymbol{w}_{opt} = \boldsymbol{w}_{sr} + \boldsymbol{w}_{cr}. \tag{3.9}$$

É interessante notar que quando não existe um sinal de referência, isto é, $d(k) = 0$, como é o caso das aplicações **LCMV**, isto implica em que $\boldsymbol{p} = E[d^*(k)\boldsymbol{x}(k)] = 0$ e a EQ. 3.8 se simplifica para

$$\boldsymbol{w}_{opt} = \boldsymbol{R}^{-1}\boldsymbol{C}(\boldsymbol{C}^H\boldsymbol{R}^{-1}\boldsymbol{C})^{-1}\boldsymbol{f}. \tag{3.10}$$

3.3 O ALGORITMO RLS COM RESTRIÇÕES LINEARES

A partir da solução de Wiener para o problema de filtragem adaptativa sujeito a restrições lineares dada pela EQ. 3.8, e resumida na EQ. 3.9, pode-se, usando as equações do algoritmo RLS vistas no Capítulo 2 (Seção 2.3), chegar facilmente ao algoritmo RLS com restrições lineares. Para tanto, basta recordar que a EQ. 3.9 nos diz claramente que este problema se resolve somando à solução sem restrições, $\boldsymbol{w}_{sr}$, a parcela $\boldsymbol{w}_{cr}$. Assim, a parcela sem restrição continua sendo atualizada como

$$\boldsymbol{w}_{sr}(k) = \boldsymbol{w}_{sr}(k-1) + e^{*}_{sr}(k)\boldsymbol{\kappa}(k). \tag{3.11}$$

Agora, fazendo $\boldsymbol{\Gamma}(k) = \boldsymbol{R}^{-1}(k)\boldsymbol{C}$ e $\boldsymbol{\Psi}(k) = \boldsymbol{C}^{H}\boldsymbol{R}^{-1}(k)\boldsymbol{C} = \boldsymbol{C}^{H}\boldsymbol{\Gamma}(k)$ a parcela com restrições pode ser reescrita como

$$\boldsymbol{w}_{cr}(k) = \boldsymbol{\Gamma}(k)\boldsymbol{\Psi}^{-1}(k)\left[\boldsymbol{f} - \boldsymbol{C}^{H}\boldsymbol{w}_{sr}(k)\right]. \tag{3.12}$$

Finalmente, a solução completa, obtida somando-se a EQ. 3.11 à EQ. 3.12, é

$$\begin{aligned} \boldsymbol{w}(k) &= \boldsymbol{w}_{sr}(k) + \boldsymbol{w}_{cr}(k) \\ &= \boldsymbol{w}_{sr}(k) + \boldsymbol{\Gamma}(k)\boldsymbol{\Psi}^{-1}(k)\left[\boldsymbol{f} - \boldsymbol{C}^{H}\boldsymbol{w}_{sr}(k)\right]. \end{aligned} \tag{3.13}$$

Obviamente, para uma aplicação **LCMV**, a equação anterior se reduz a

$$\boldsymbol{w}(k) = \boldsymbol{\Gamma}(k)\boldsymbol{\Psi}^{-1}(k)\boldsymbol{f}. \tag{3.14}$$

Vale a pena ressaltar que o cálculo de $\boldsymbol{\Psi}^{-1}(k)$ é simples para o caso de se ter apenas uma restrição quando $\boldsymbol{\Psi}(k)$ é escalar. Para o caso de mais de uma restrição, é possível determinar uma expressão recursiva para $\boldsymbol{\Psi}^{-1}(k)$, evitando-se assim o cálculo da inversa da matriz. O algoritmo RLS com restrições lineares (CRLS) básico está descrito na TAB. 3.1 (RESENDE, 1996).

3.4 ESTRUTURAS ALTERNATIVAS AOS ALGORITMOS COM RESTRIÇÕES LINEARES

Uma boa alternativa ao uso de algoritmos com restrições lineares é o uso de estruturas que permitem a utilização de algoritmos sem restrição para solucionar problemas com restrições. Estas estruturas garantem que as restrições continuem sendo satisfeitas de acordo com a EQ. 3.1, ou seja, que $\boldsymbol{C}^{H}\boldsymbol{w} = \boldsymbol{f}$.

TAB. 3.1: O Algoritmo CRLS básico

CRLS
Inicializações:
$\boldsymbol{w}_{sr}(0)$ e $\boldsymbol{R}^{-1}(0) = \mu\boldsymbol{I}$ onde μ = número pequeno
for $k = 1, 2, \cdots$
{
$\boldsymbol{k}(k) = \boldsymbol{R}^{-1}(k-1)\boldsymbol{x}(k)$
$\boldsymbol{\kappa}(k) = \frac{\boldsymbol{k}(k)}{\lambda + \boldsymbol{x}^H(k)\boldsymbol{k}(k)}$
$\boldsymbol{R}^{-1}(k) = \frac{1}{\lambda}\left[\boldsymbol{R}^{-1}(k-1) - \frac{\boldsymbol{k}(k)\boldsymbol{k}^H(k)}{\lambda + \boldsymbol{x}^H(k)\boldsymbol{k}(k)}\right]$
$e_{sr}(k) = d(k) - \boldsymbol{w}_{sr}^H(k-1)\boldsymbol{x}(k)$
$\boldsymbol{w}_{sr}(k) = \boldsymbol{w}_{sr}(k-1) + e_{sr}^*(k)\boldsymbol{\kappa}(k)$
$\boldsymbol{\Gamma}(k) = \boldsymbol{R}^{-1}(k)\boldsymbol{C}$
$\boldsymbol{\Psi}(k) = \boldsymbol{C}^H\boldsymbol{\Gamma}(k)$
$\boldsymbol{w}(k) = \boldsymbol{w}_{sr}(k) + \boldsymbol{\Gamma}(k)\boldsymbol{\Psi}^{-1}(k)\left[\boldsymbol{f} - \boldsymbol{C}^H\boldsymbol{w}_{sr}(k)\right]$
}

A principal vantagem da utilização destas estruturas reside no fato de que existem vários algoritmos sem restrições com convergência e estabilidades garantidas, o que não se verifica com a grande maioria dos algoritmos com restrições conhecidos. Nesta seção, são abordadas, de forma resumida, duas dessas estruturas: a GSC (*Generalized Sidelobe Canceller*) (GRIFFITHS, 1982) e a Householder Constrained (DE CAMPOS, 2002).

3.4.1 A ESTRUTURA GSC

Para satisfazer as restrições impostas por um determinado problema, usando-se algoritmos sem restrições, a estrutura GSC aplica uma transformação $\boldsymbol{T}$ ao vetor de coeficientes definida da seguinte maneira

$$\boldsymbol{T} = [\boldsymbol{C} \quad \boldsymbol{B}]\,. \tag{3.15}$$

tal que

$$\begin{aligned} \boldsymbol{w}(k) &= \boldsymbol{T}\overline{\mathbf{w}}(k) \\ &= [\boldsymbol{C} \quad \boldsymbol{B}] \begin{bmatrix} \overline{\mathbf{w}}_S(k) \\ -\overline{\mathbf{w}}_I(k) \end{bmatrix} \\ &= \boldsymbol{C}\overline{\mathbf{w}}_S(k) - \boldsymbol{B}\overline{\mathbf{w}}_I(k) \end{aligned} \tag{3.16}$$

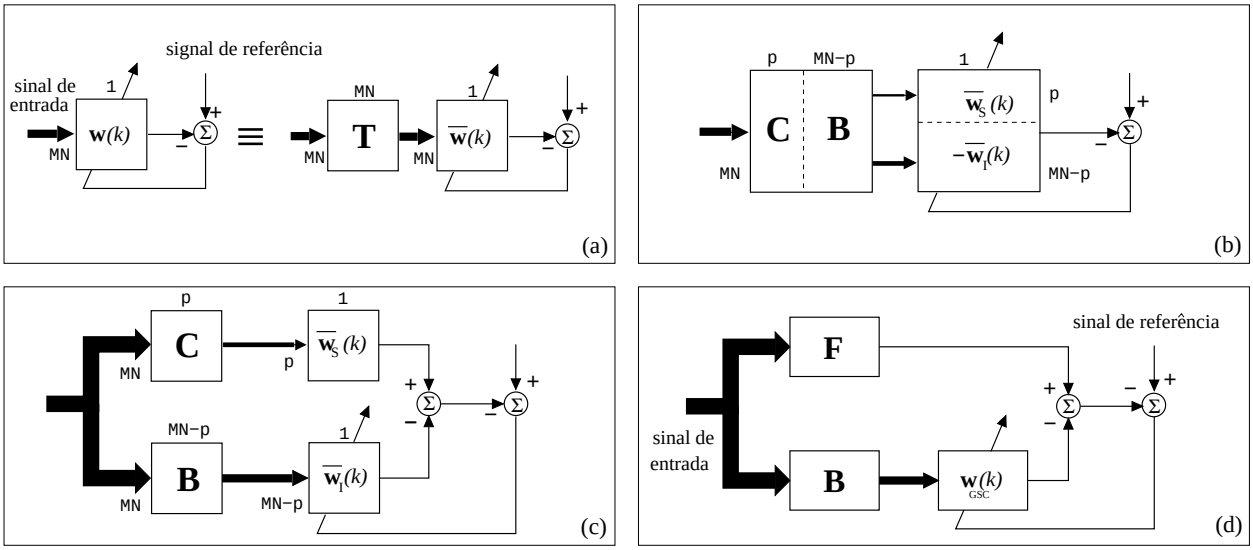

FIG. 3.1: O processo de transformação.

A matriz $\boldsymbol{B}$ é designada por sua expressão em inglês *Blocking Matrix* e os vetores $\overline{\mathbf{w}}_S(k)$ e $\overline{\mathbf{w}}_I(k)$ representam as partes superiores e inferiores de $\overline{\mathbf{w}}(k)$, respectivamente. Recordando que as restrições são tais que $\boldsymbol{C}^H\boldsymbol{w}(k) = \boldsymbol{f}$, pode-se escrever, usando a EQ. 3.16, que

$$\boldsymbol{C}^H\boldsymbol{C}\overline{\mathbf{w}}_S(k) - \boldsymbol{C}^H\boldsymbol{B}\overline{\mathbf{w}}_I(k) = \boldsymbol{f}. \tag{3.17}$$

Agora, se a condição $\boldsymbol{B}^H\boldsymbol{C} = 0$, ou a sua equivalente $\boldsymbol{C}^H\boldsymbol{B} = \mathbf{0}$, for imposta, podemos escrever, da EQ. 3.17, que

$$\boldsymbol{C}^H\boldsymbol{C}\overline{\mathbf{w}}_S(k) = \boldsymbol{f} \quad \Longrightarrow \quad \overline{\mathbf{w}}_S(k) = (\boldsymbol{C}^H\boldsymbol{C})^{-1}\boldsymbol{f}.$$

Da relação anterior, percebe-se, claramente, que $\overline{\mathbf{w}}_S(k)$ é uma parte constante de $\overline{\boldsymbol{w}}(k)$, o que significa, em outras palavras, que ela permanecerá com o valor de inicialização. A parte que sofrerá atualização, $\overline{\mathbf{w}}_I(k)$, será designada de agora em diante por $\mathbf{w}_{GSC}(k)$. A parte que não sofrerá atualização corresponde ao vetor $\boldsymbol{F} = \boldsymbol{C}\overline{\boldsymbol{w}}_S(k) = \boldsymbol{C}(\boldsymbol{C}^H\boldsymbol{C})^{-1}\boldsymbol{f}$. Todo o processo explicado até aqui está resumido de forma esquemática na FIG. 3.1.

É interessante mencionar que, em muitas aplicações práticas, não existe um sinal de referência, isto é, $d(k) = 0$, tal que a FIG. 3.1 (d) se reduz ao diagrama da FIG. 3.2. Nesta última figura, as variáveis internas estão detalhadas. Observa-se facilmente que o

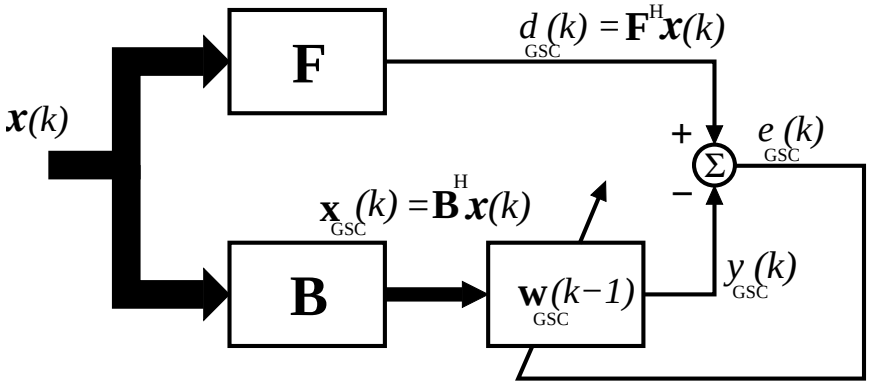

FIG. 3.2: A Estrutura GSC mais usual.

vetor de coeficientes completo, $\boldsymbol{w}(k)$, é dado por

$$\boldsymbol{w}(k) = \boldsymbol{F} - \boldsymbol{B}\mathbf{w}_{GSC}(k). \tag{3.18}$$

Observações acerca da *Blocking Matrix B*

Como foi visto anteriormente, a única restrição para a matriz $\boldsymbol{B}$ é que ela seja ortogonal em relação à matriz de restrições $\boldsymbol{C}$, isto é $\boldsymbol{C}^H\boldsymbol{B} = \mathbf{0}$. Sendo assim, as possibilidades são amplas, mas o custo computacional e o tipo de aplicação são fatores que influenciarão nessa escolha. Para alguns casos simples, a matriz $\boldsymbol{B}$ pode ser construída a partir de funções de Walsh (GRIFFITHS, 1982). Em outros casos, pode ser obtido usando as decomposições QR ou SVD, gerando, no entanto, matrizes que implicam em um custo computacional mais alto.

3.4.2 A ESTRUTURA *HOUSEHOLDER CONSTRAINED*

Considerando um caso mais geral para o problema de otimização sujeita a restrições, usando-se uma matriz $\boldsymbol{B}$ ortogonal (não trivial), a multiplicação $\mathbf{x}_{GSC}(k) = \boldsymbol{B}^H\boldsymbol{x}(k)$, como mostrado na FIG. 3.2, pode acarretar um aumento considerável na complexidade computacional do algoritmo. Para casos como este, a estrutura *Householder Constrained*, proposta em (DE CAMPOS, 2002), apresenta uma solução mais eficiente.

A estrutura *Householder Constrained* pode ser encarada como uma estrutura GSC onde a matriz de transformação é uma matriz ortogonal $\boldsymbol{Q}$. Esta matriz é construída com

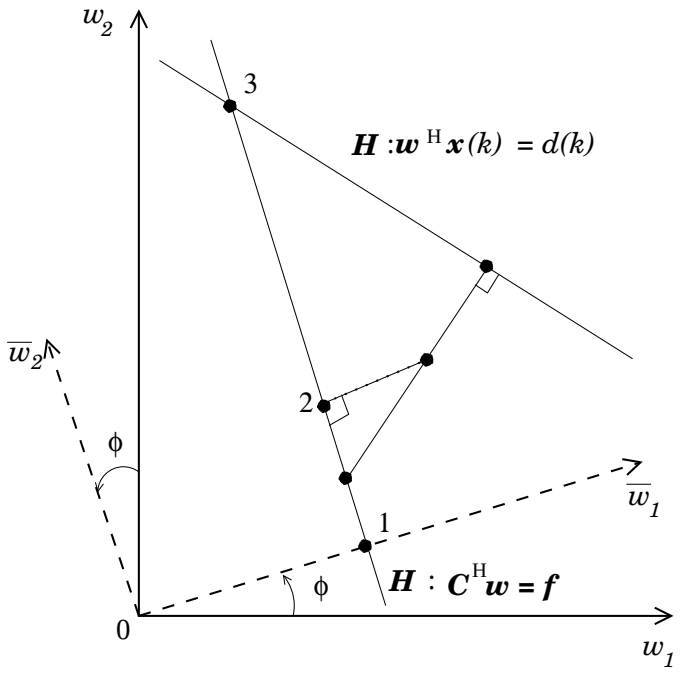

FIG. 3.3: Rotação do vetor de coeficientes.

transformações de Householder sucessivas e efetua uma rotação no vetor de coeficientes $\boldsymbol{w}$ de um ângulo ϕ, dando origem a um vetor de coeficientes transformado $\bar{\mathbf{w}}$ como mostrado na FIG. 3.3.

Matematicamente, $\boldsymbol{w}$ pode ser relacionada à $\bar{\mathbf{w}}$ como segue.

$$\bar{\mathbf{w}}(k) = \boldsymbol{Q}\boldsymbol{w}(k) \Longrightarrow \begin{bmatrix} \bar{w}_1(k) \\ \bar{w}_2(k) \end{bmatrix} = \boldsymbol{Q} \begin{bmatrix} w_1(k) \\ w_2(k) \end{bmatrix} \tag{3.19}$$

Pode-se observar na FIG. 3.3 que o ângulo de rotação ϕ é escolhido tal que o eixo $\bar{w}_2$ fique paralelo ao hiperplano das restrições, $\boldsymbol{C}^H\boldsymbol{w} = \boldsymbol{f}$. Desta forma, a coordenada correspondente a $\bar{w}_1$, após ser inicializada, não precisa de atualização.

3.4.3 ESTRUTURA GSC × ESTRUTURA HOUSEHOLDER CONSTRAINED

A FIG. 3.4 estabelece um paralelo entre essas duas estruturas. Vê-se que se a matriz $\boldsymbol{Q}$ for particionada em duas partes, $\boldsymbol{Q}_S$ e $\boldsymbol{Q}_I$, esta última pode ser encarada como uma *Blocking Matrix* válida, comparável à matriz $\boldsymbol{B}$ da estrutura GSC. Pode ser demonstrado (DE CAMPOS, 2002) que $\boldsymbol{Q}_I\boldsymbol{C} = \boldsymbol{0}$. Ainda da mesma figura, vê-se que $\boldsymbol{Q}_S^T\bar{\mathbf{w}}_S(0) = \boldsymbol{C}(\boldsymbol{C}^H\boldsymbol{C})^{-1}\boldsymbol{f} = \boldsymbol{F}$ corresponde à parte superior da estrutura GSC. Porém, o que torna

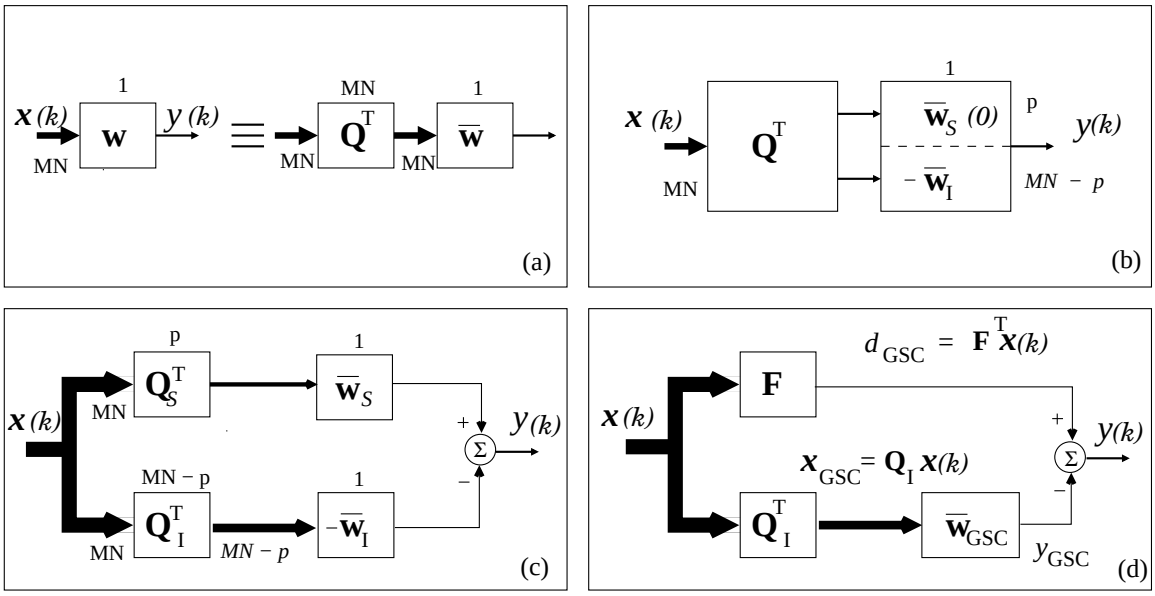

FIG. 3.4: Filtragem adaptativa com restrições usando a transformação de Householder vista como uma estrutura GSC.

a estrutura Householder computacionalmente mais atraente, particularmente nos casos em que não se pode usar uma *Bloking Matrix* $\boldsymbol{B}$ trivial, construída apenas com uns e zeros, é a forma eficiente pela qual se pode computar o vetor transformado do sinal de entrada $\boldsymbol{x}(k)$, $\bar{\boldsymbol{x}}(k)$, usando p transformações de Householder sucessivas (DE CAMPOS, 2002), como na TAB. 3.2, usando um pseudo-código com uma notação similar à usada no Matlab©.

O pseudo-código na TAB. 3.3, usando também uma notação similar à do Matlab©, mostra como construir a matriz $\boldsymbol{V}$ e obter a matriz $\boldsymbol{Q}$.

3.4.4 O ALGORITMO HOUSEHOLDER CONSTRAINED *CONJUGATE GRADIENT*

A estrutura Householder *Constrained* (HC), assim como a estrutura GSC, pode ser usada com qualquer algoritmo adaptativo utilizável num combinador linear adaptativo (isto porque o vetor de entrada transformado conforme indicado anteriormente pode ser visto como a entrada de $MN - p$ canais de um combinador linear adaptativo).

Como exemplo da utilização desta estrutura, é apresentado na TAB. 3.4 o algoritmo Householder *Constrained Conjugate Gradient*. Este algoritmo, resultado da utilização do algoritmo do Gradiente Conjugado (CHANG, 1995, 2000) na estrutura Householder

TAB. 3.2: Cálculo de $\bar{\boldsymbol{x}}(k) = \boldsymbol{Q}\boldsymbol{x}(k)$.

$\bar{\boldsymbol{x}}_k = \boldsymbol{x}(k)$;
for $i = 1 : p$
{
$\quad \bar{\boldsymbol{x}}_k(i : MN) = \bar{\boldsymbol{x}}_k(i : MN)$
$\qquad -2\boldsymbol{V}(i : MN, i)[\boldsymbol{V}^H(i : MN, i)\bar{\boldsymbol{x}}_k(i : MN)]$;
}
$\bar{\boldsymbol{x}}(k) = \bar{\boldsymbol{x}}_k$

TAB. 3.3: Construindo $\boldsymbol{V}$ e Obtendo $\boldsymbol{Q}$.

Inicializar:
$\quad \boldsymbol{V} = \boldsymbol{0}_{MN\times p}$;
$\quad \boldsymbol{A}_{MN\times p} = \boldsymbol{C}\sqrt{(C^H C)^{-1}}$;
$\quad \bar{\boldsymbol{x}}_k = \boldsymbol{x}(k)$;
$\quad$ for $i = 1 : p$
$\qquad$ {
$\qquad \boldsymbol{x} = \boldsymbol{A}(i : MN, i)$
$\qquad \boldsymbol{e}_1 = [\boldsymbol{1} \quad \boldsymbol{0}_{1\times(MN-i)}]^T$;
$\qquad \boldsymbol{v} =$ sign $(\boldsymbol{x}(1))\|\boldsymbol{x}\|\boldsymbol{e}_1 + \boldsymbol{x}$;
$\qquad \boldsymbol{v} = \boldsymbol{v}/\|\boldsymbol{v}\|$;
$\qquad \boldsymbol{A}(i : MN, i : p) = \boldsymbol{A}(i : MN, i : p) - 2\boldsymbol{v}(\boldsymbol{v}^H\boldsymbol{A}(i : MN, i : p))$;
$\qquad \boldsymbol{V}(i : MN, i) = \boldsymbol{v}$;
$\qquad \boldsymbol{Q} = \begin{bmatrix} \boldsymbol{I}_{(i-1)} & \boldsymbol{0}_{(i-1,MN-i+1)} \\ \boldsymbol{0}_{(MN-i+1,i-1)} & \boldsymbol{I}_{(MN-i+1)} - 2\boldsymbol{v}\boldsymbol{v}^H \end{bmatrix}$;
$\quad$ }

Constrained, é um dos vários cujo desempenho é avaliado no próximo capítulo.

3.5 RESUMO

Neste capítulo, foi feita uma introdução à filtragem adaptativa com restrições. Levou-se em consideração os casos mono e multicanal e uma versão básica do algoritmos RLS com restrições foi apresentada. Foram vistas, também, duas estruturas bastante úteis que permitem que sejam utilizados algoritmos sem restrições, robustos, para solucionar problemas em que seja necessária a incorporação de determinadas restrições lineares à solução.

TAB. 3.4: O algoritmo *HCCG*.

HCCG

Disponíveis no instante k:
$\mathbf{x}(k)$, $\mathbf{C}$ $\mathbf{f}$, and $\mathbf{Q}$
Inicializar:
λ, η with $(\lambda - 0.5) \leq \eta \leq \lambda$
δ número pequeno
$\overline{\mathbf{w}}(0) = \mathbf{QF} = \mathbf{QC}(\mathbf{C^H C})^{-1}\mathbf{f}$
$\overline{\mathbf{R}} = \mathbf{I}_{MJ-p}$
$\mathbf{g}(0) = \mathbf{c}(0) = zeros(MN - p, 1)$
para cada k
{
$\overline{\mathbf{x}}(k) = \mathbf{Q}\mathbf{x}(k)$;
$\overline{\mathbf{x}}_S(k) = p$ primeiros elementos de $\overline{\mathbf{x}}(k)$;
$\overline{\mathbf{x}}_I(k) = MN - p$ últimos elementos de $\overline{\mathbf{x}}(k)$;
$\overline{\mathbf{w}}(k) = \begin{bmatrix} \overline{\mathbf{w}}_S(0) \\ -\overline{\mathbf{w}}_I(k-1) \end{bmatrix}$;
$\overline{\mathbf{R}}(k) = \lambda\overline{\mathbf{R}}(k-1) + \overline{\mathbf{x}}_I(k)\overline{\mathbf{x}}_I^H(k)$
$\overline{e}(k) = \overline{\mathbf{w}}_S^H(0)\overline{\mathbf{x}}_S(k) - \overline{\mathbf{w}}_I^H(k-1)\overline{\mathbf{x}}_I(k)$;
$\alpha(k) = \eta\frac{\mathbf{c}^H(k)\mathbf{g}(k-1)}{\mathbf{c}^H(k)\overline{\mathbf{R}(k)}\mathbf{c}(k)+\delta}$
$\mathbf{g}(k) = \lambda\mathbf{g}(k-1) - \alpha(k)\overline{\mathbf{R}}(k)\mathbf{c}(k) - \mathbf{x}_L(k)\overline{e}^*(k)$;
$\overline{\mathbf{w}}_I(k) = \overline{\mathbf{w}}_I(k-1) - \alpha\mathbf{c}(k)$
$\beta(k) = \frac{[\mathbf{g}(k)-\mathbf{g}(k-1)]^H\mathbf{g}(k)}{\mathbf{g}^H(k-1)\mathbf{g}(k-1)+\delta}$;
$\mathbf{c}(k+1) = \mathbf{g}(k) + \beta(k)\mathbf{c}(k)$;
}

4 SOBRE O DESEMPENHO DOS ALGORITMOS ADAPTATIVOS DE CONVERGÊNCIA RÁPIDA COM RESTRIÇÕES

4.1 INTRODUÇÃO

Neste capítulo, é analisado o desempenho dos algoritmos da família RLS com restrições — foram incluídos, ainda, dois algoritmos que não pertencem a essa mesma família, mas que também apresentam convergência rápida: o algoritmo CCG (*Constrained Conjugate Gradient*) (APOLINÁRIO JR, 2000b) e o CQN (*Constrained Quasi-Newton*) (DE CAMPOS, 1998) — e suas correspondentes versões sem restrições usadas nas estruturas GSC e *Householder Constrained*, vistas no Capítulo 3, numa aplicação de *beamforming* adaptativo.

Na Seção 4.2, é apresentada uma breve introdução ao *Beamforming*. Na Seção 4.3, é feita uma descrição das características do ambiente de simulação utilizado. Na seção seguinte, serão apresentados os resultados de simulações realizadas com os diversos algoritmos acima referidos. Esses resultados compreendem uma comparação em termos da velocidade de convergência, estabilidade numérica e complexidade computacional. Na Seção 4.5, são apresentadas as principais conclusões sobre os desempenhos.

4.2 O *BEAMFORMING*

O enfoque principal não é a teoria do *Beamforming*, mas sim o desempenho dos algoritmos de convergência rápida neste tipo de aplicação. Por isso, nesta seção, são apresentados apenas alguns conceitos básicos para o entendimento do experimento realizado neste tipo de ambiente.

4.2.1 UMA VISÃO GERAL

O aumento na demanda nos sistemas de comunicações sem fio torna necessário o aumento da capacidade destes sistemas, tanto em termos de aumento da velocidade de transmissão de dados quanto em termos de um maior número de usuários.

Isto pode ser conseguido aumentando-se a largura de banda dos canais de comunicação e alocando novas freqüências. Porém, pelo fato do espectro eletromagnético ser limitado,

o aumento da capacidade de um sistema pode rapidamente atingir o limite máximo. Isto motivou o desenvolvimento de técnicas que maximizam o aproveitamento do espectro designado para um determinado serviço. Estas técnicas são conhecidas por técnicas de múltiplo acesso: vários usuários transmitindo para um receptor comum ao mesmo tempo.

No múltiplo acesso, implementado com vários usuários compartilhando uma mesma estação base, o compartilhamento pode se dar a quatro níveis distintos: freqüência; tempo; código ou espaço. No acesso múltiplo por divisão de freqüência (FDMA–*Frequency-Division Multiple Access*) o espectro de freqüência é dividido em porções atribuídas a diferentes usuários. No acesso múltiplo por divisão de tempo (TDMA–*Time Division Multiple access*), possível com o advento de tecnologias digitais, o usuário recebe o acesso à capacidade total de transmissão do sistema por um determinado período de tempo. No acesso múltiplo por divisão de código (CDMA–*Code-Division Multiple Access*), também possível graças ao avanço das tecnologias digitais, a informação é espalhada em todas a faixa de freqüência. Cada sinal transmitido é modulado com um código único que identifica cada usuário. Com isso, cada receptor usa o código apropriado para identificar o sinal de interesse.

A última forma de acesso múltiplo, conhecida como acesso múltiplo por divisão de espaço (SDMA–*Space-Division Multiple Access*) (LITVA, 1996), é conseguido dividindo uma determinada área num grande número de pequenas células. Isso permite que uma mesma portadora possa ser reutilizada em células diferentes, possibilitando, assim, um aumento de capacidade do sistema de comunicações. Obviamente, quanto maior o número de células maior será o aumento de capacidade conseguida. No entanto, para que haja o mínimo de interferência, a distância entre as células deve atender a um determinado valor. Isso impõe um limite ao número de células numa determinada área e, conseqüentemente, no aumento de capacidade que se pode conseguir para um determinado sistema.

O SDMA é pode ser implementado usando a técnica conhecida como *Beamforming*. Embora o *Beamforming* tenha sido implementado inicialmente de forma analógica, o uso de técnicas digitais são necessárias para implementar o chamado *Beamforming* adaptativo. Este, por sua vez, requer a utilização de algoritmos adaptativos para formar, em tempo real, o *Beam Pattern* desejado. Em comunicações, o *Beamforming* tem um apelo maior na redução do nível de interferência e de ruído, dadas as complicações inerentes à implementação prática do SDMA. Na próxima subseção, são apresentados alguns tópicos básicos sobre o *Beamforming* digital.

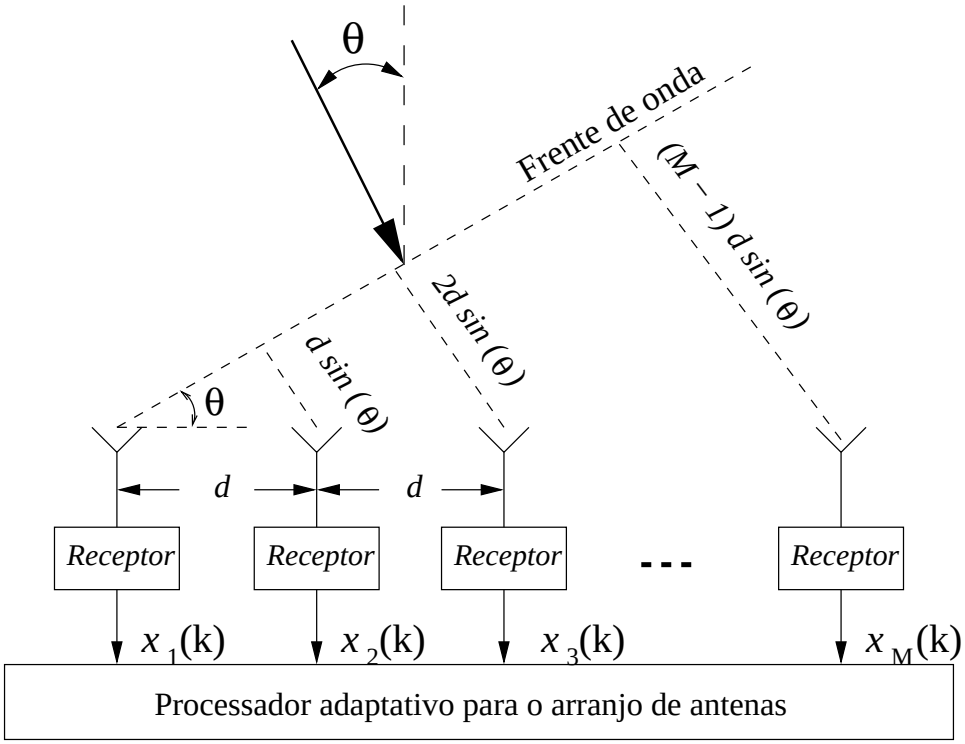

FIG. 4.1: Uma frente de onda plana chegando a um arranjo de antenas

4.2.2 O *BEAMFORMING* DIGITAL

Um dispositivo capaz de separar sinais numa mesma faixa de freqüências mas provenientes de pontos distintos do espaço é chamado de *beamformer*. O *Beamforming* digital foi primeiramente direcionado para sistemas de radar (BARTON, 1980) e sonar (CURTIS, 1980). Mas, com o rápido avanço, tanto teóricos como na criação de dispositivos de *hardware* para o processamento de sinais digitais, *beamformers* digitais, certamente, serão incorporados em sistemas de comunicações futuros visando o aumento de capacidade destes sistemas para atender à demanda crescente.

4.2.3 O *BEAMFORMING* ADAPTATIVO

Num *Beamformer* adaptativo, os pesos do *Beam Pattern* precisam sofrer adaptações de acordo com as condições do ambiente. A necessidade da adaptação do *Beam Pattern*, usando um algoritmo adaptativo, decorre do fato de, normalmente, as estatísticas do sinal de entrada serem desconhecidas. O algoritmo de treinamento usado na adaptação tem

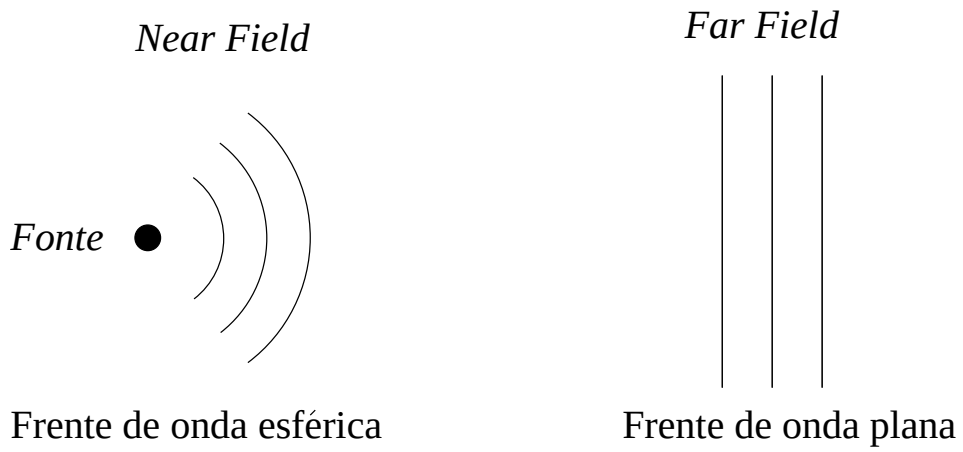

FIG. 4.2: Aproximação *far field*.

que atender determinadas restrições convenientemente impostas para que o sinal desejado seja recebido e os sinais provenientes de direções distintas da de interesse sejam atenuados. Nestes casos, pode-se utilizar algoritmos com restrições diretamente ou algoritmos sem restrições dentro de estruturas como a GSC ou a Householder Constrained, como visto no Capítulo 3.

4.2.4 ARRANJOS ADAPTATIVOS DE ANTENAS

Para a implementação de um *beamformer*, é necessário um arranjo de sensores convenientemente espaçados para receber os sinais provenientes das várias direções. Estes sensores, como indicado na FIG. 4.1, podem ser antenas, para sinais eletromagnéticos, ou microfones, para sinais acústicos. Quando é assumido que o sinal chega ao arranjo de sensores com uma frente de onda plana (*far field*), os cálculos podem ser simplificados. Para o caso de arranjos de antenas em que as distâncias são sempre relativamente grandes é sempre assumido o caso *far field*. O contrário é o *near field*. Estes dois casos estão ilustrados na FIG. 4.2.

São duas as condições para que a aproximação *far field* seja aceita. A primeira é dada por (STUTZMAN, 1998):

$$r > \frac{2D^2}{\tau}, \tag{4.1}$$

com $r \gg D$ e $r \gg \tau$, onde r é a distância à fonte, D é o comprimento do arranjo e τ é o comprimento de onda.

A segunda condição é que o sinal seja de faixa estreita (HIEMSTRA, 2003). Em situações em que não seja possível satisfazer esta última condição, o que, em certos casos, pode tornar difícil de satisfazer a primeira, uma alternativa possível é usar um banco de filtros para dividir o sinal de faixa larga em vários sinais de banda estreita (VAIDYANATHAN, 1993), processá-los separadamente em outros tantos *beamformers* digitais e recombinar as saídas para obter o *beamforming* desejado.

4.2.5 ALGUMAS DEFINIÇÕES FUNDAMENTAIS

Neste ponto, precisamos mencionar algumas definições importantes envolvendo o *Beamforming*.

- **O ruído** – é considerado o primeiro componente indesejável de um sinal. Normalmente, ele é modelado como sendo um processo estocástico, estacionário em sentido amplo, Gaussiano de média nula. Como está associado a um receptor distinto, os processos ruidosos gerados em diferentes receptores podem ser considerados independentes. Neste caso, a sua matriz autocorrelação é dada por

$$\boldsymbol{R}_n = \sigma_n^2 \boldsymbol{I}, \tag{4.2}$$

  onde $\boldsymbol{I}$ é a matriz identidade de tamanho $K \times K$, sendo K o número de elementos no arranjo.

- **Interferência** (sinal não desejado, proposital ou não) [4] – é aqui modelada como uma fonte isolada no espaço; sua matriz de autocorrelação pode ser expressa como

$$\boldsymbol{R}_j = \sigma_j^2 \boldsymbol{v}_{\theta_j} \boldsymbol{v}_{\theta_j}^H, \tag{4.3}$$

  onde σ_j^2 é a potência (variância) do sinal interferidor (de média nula) e $\boldsymbol{v}_{\theta_j}$ é conhecido pela expressão em inglês *array manifold vector* e está associado ao interferidor j que chega ao arranjo proveniente de uma direção θ_j; para o caso de um arranjo linear, $\boldsymbol{v}_\theta$ é dado por

$$\boldsymbol{v}_\theta = \left[e^{i0} \quad \cdots \quad e^{i2\pi \frac{d}{\tau} \sin\theta} \quad e^{i2\pi(N-1)\frac{d}{\tau}\sin\theta} \right]^T, \tag{4.4}$$

  onde d é a distância entre os elementos do arranjo, τ é o comprimento de onda da freqüência central e $i = \sqrt{-1}$.

[4] Na literatura relacionada a radar e Guerra Eletrônica, é comum utilizar-se a expressão *jamming* para designar uma interferência proposital.

- **A razão sinal-ruído** – é definida como

$$RSR = \frac{\sigma_s^2}{\sigma_n^2}, \tag{4.5}$$

onde σ_s^2 é a potência (variância) do sinal desejado (de média nula) que chega a cada elemento do arranjo.

- **A razão interferidor-ruído** – é definida como

$$RJR = \frac{\sigma_j^2}{\sigma_n^2}, \tag{4.6}$$

onde σ_j^2 é a potência do interferidor que chega a cada elemento do arranjo.

4.3 AVALIAÇÃO DO DESEMPENHO DOS VÁRIOS ALGORITMOS

A avaliação leva em consideração um ambiente de *beamforming* a ser descrito nesta seção. São avaliados os seguintes algoritmos: o algoritmo CRLS (*Constrained Recursive Least Squares*) (RESENDE, 1996), CQRD-RLS (*Constrained QRD-RLS*) (TANG, 1991) e o CIQRD-RLS (*Constrained Inverse QRD-RLS*) (CHERN, 2002), da família RLS, e dois outros algoritmos de convergência rápida, o CCC (*Constrained Conjugate Gradient*) (APOLINÁRIO JR, 2000b) e o CQN (*Constrained* Quasi–*Newton*) (DE CAMPOS, 1998). O algoritmo CQRD-RLS (*Constrained QRD-RLS*) (TANG, 1991) é apenas uma solução MVDR (*Minimum Variance Distortionless Response*), assim como o algoritmo de (MOONEN, 2000) que é baseado no QRD-RLS Inverso ao contrário do algoritmo CIQRD-RLS de (CHERN, 2002) que apresenta uma solução generalizada.

Cada um dos algoritmos acima possui correspondentes versões sem restrições que podem ser usados em aplicações em que seja necessária a imposição de restrições lineares com o auxilio das estruturas GSC e Householder Constrained. Nestes casos, eles serão referidos usando o prefixo apropriado (GSC ou HC). Exemplificando, para o algoritmo CRLS tem-se os respectivos correspondentes GSC-RLS e HCRLS. Nestas estruturas, foram utilizados os seguintes algoritmos: o RLS convencional, (DINIZ, 2002), o CG (*Conjugate Gradient*), (CHANG, 2000), o QN (*Quasi–Newton*) (DE CAMPOS, 1997), o QRD-RLS convencional (DINIZ, 2002) e o IQRD-RLS (*Inverse QRD-RLS*) (ALEXANDER, 1993).

O desempenho destes algoritmos é avaliado inicialmente no que diz respeito à sua estabilidade. Na próxima seção, é analisada a complexidade computacional e a capacidade de rejeição de interferências. Neste experimento de *beamforming* foi utilizado, como em

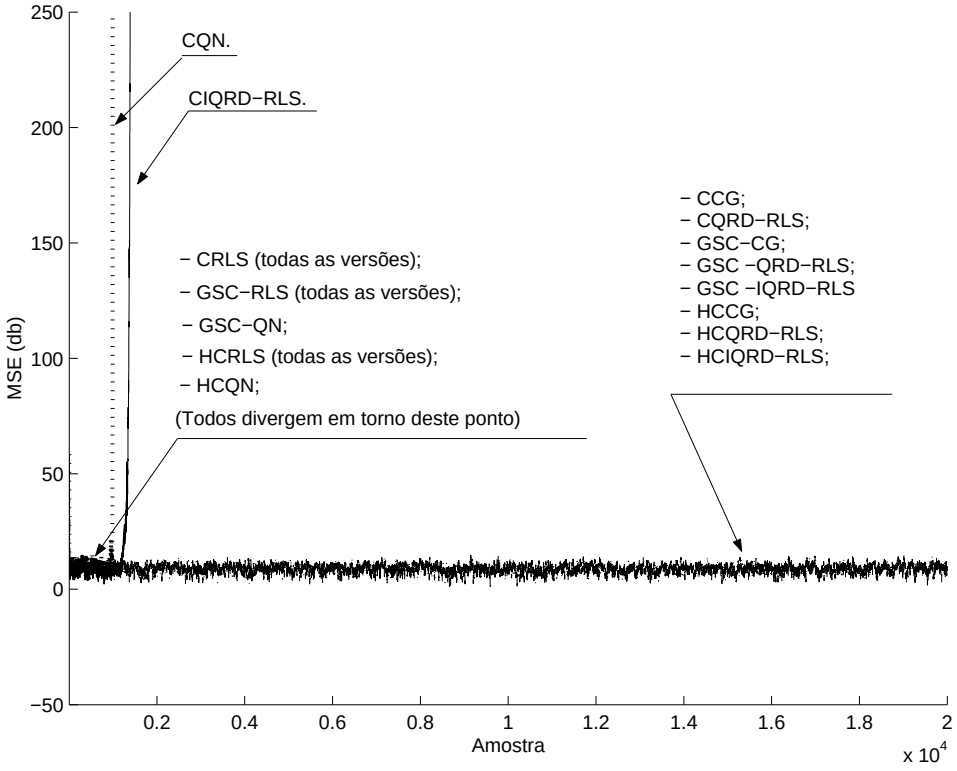

FIG. 4.3: Convergência dos diversos algoritmos tipo RLS, com restrições, GSC e Householder Constrained.

(CHERN, 2002), um arranjo linear de 7 sensores como uma direção de recebimento de $0°$ e três sinais interferidores (*jammers*) com ângulos de incidência a $-25°$, $45°$, e $50°$. A razão sinal-ruído usada é de $0dB$ e a razão interferidor-ruído é de $30dB$. O fator de esquecimento (λ) é de $0,98$.

4.4 COMPARANDO A PERFORMANCE DOS ALGORITMOS ROBUSTOS

Durante o experimento, observou-se que todas as versões do algoritmo CRLS apresentadas em (RESENDE, 1996) são instáveis assim como as suas correspondentes GSC e Householder Constrained. Mesmo o algoritmo CIQRD-RLS (CHERN, 2002), que supostamente seria numericamente mais estáveis que o algoritmo CRLS convencional, apresentou problemas de estabilidade em aplicações LCMV (*Linearly Constrained Minimum variance*).

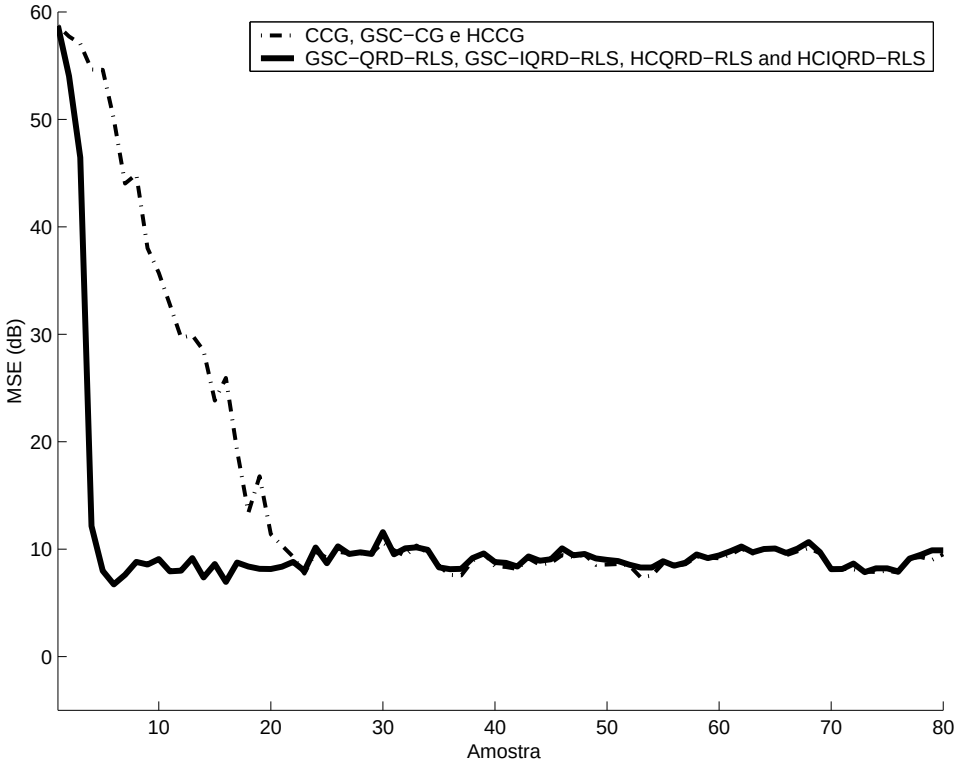

FIG. 4.4: Curva de aprendizagem durante o transiente

Após 10 experimentos independentes de 20.000 amostras cada um, pode-se observar da FIG. 4.3 que poucos dos algoritmos testados permanecem estáveis: o algoritmo CCG (*Constrained Conjugate Gradient*) (APOLINÁRIO JR, 2000b), bem como os seus correspondentes GSC e Householder Constrained (usando a versão sem restrição como descrita em (CHANG, 2000)), o GSC e Householder Constrained QRD-RLS (DINIZ, 2002) e o GSC e Householder Constrained QRD-RLS inverso (ALEXANDER, 1993).

Embora não mostrado na FIG. 4.3, foram feitos outros 10 experimentos independentes com 6×10^5 amostras cada e os algoritmos *estáveis* não apresentaram sinais de divergência.

Reproduziu-se aqui o experimento de (CHERN, 2002); vale a pena ressaltar que o valor do fator de esquecimento (λ) foi variado e o único efeito observado, para o caso dos algoritmos instáveis da FIG. 4.3, incluído o algoritmo QRD–RLS Inverso com restrição (CIQRD–RLS), foi o retardamento ou adiantamento do momento da divergência. Observa-se que (CHERN, 2002) não menciona o valor de λ usado.

Foi observado, ainda, que o algoritmo CIQRD introduzido em (CHERN, 2002), quando

TAB. 4.1: Complexidade computacional dos algoritmos mais robustos.

ALGORITMO	MULTIP.	DIVIS.	R.QUAD.
CCG	$MJ(6MJ+2p+8)+1$	1	0
HCCG	$(MJ-p)(5MJ-5p+9)-p(p-2MJ-2)+1$	1	0
HCQRD-RLS	$\frac{(MJ-p)}{2}(13MJ-11p+17)+(2MJ-p+2)p+1$	$2(MJ-p)$	$MJ-p$
HCIQRD-RLS	$\frac{(MJ-p)^3}{2}+(MJ-p)(5MJ-5p+7)+p(2MJ-p+1)+2$	$2(MJ-p)+1$	$MJ-p$
GSC-CG	$MJ(7MJ-12p+9)+p(5p-8)+1$	1	0
GSC-QRD-RLS	$(MJ-p)(6MJ-5p+9)+MJ+1$	$2(MJ-p)$	$MJ-p$
GSC-IQRD-RLS	$\frac{(MJ-p)^3}{2}+(MJ-p)(6MJ-5p+7)+MJ+2$	$2(MJ-p)+1$	$MJ-p$

TAB. 4.2: Complexidade computacional (exemplo numérico).

ALGORITMO	MULTIP.	DIVIS.	R.QUAD.
CCG	365	1	0
HCCG	**249**	**1**	**0**
HCQRD-RLS	250	12	6
HCIQRD-RLS	346	13	6
GSC-CG	284	1	0
GSC-QRD-RLS	284	12	6
GSC-IQRD-RLS	381	13	6

comparado ao algoritmo GSC-IQRR-RLS, não segue a mesma curva de aprendizagem para as primeiras amostras (transiente), como seria de se esperar, pelo menos para o caso de ser usada uma matriz $\boldsymbol{B}$ (*Blocking Matrix*) ortogonal. Os demais algoritmos ditos estáveis, neste experimento, apresentaram a mesma curva de aprendizagem para a implementação usando as estruturas GSC e Householder Constrained, desde que $\mathbf{B}^H\mathbf{B}=\mathbf{I}$.

A FIG. 4.4 apresenta as curvas de aprendizagem dos algoritmos mais robustos para as primeiras 250 amostras. Desta figura podemos notar que as implementações GSC e Householder Constrained do algoritmos da família RLS que usam a decomposição QR possuem a melhor taxa de convergência para este experimento.

A complexidade computacional dos algoritmos que convergem e permanecem estáveis está resumida na TAB. 4.1. Nesta tabela, M e J representam o número de canais e o número de coeficientes do filtro, respectivamente, e p é o número de restrições. Para exemplificar a carga computacional para o caso particular da simulação feita aqui, apresentamos na TAB. 4.2 os resultados para $M=7$, $J=1$ e $p=1$.

Observa-se, a partir destas tabelas que, pelo menos para esta aplicação, o algoritmo HCCG (*Householder Constrained Conjugate Gradient*) apresenta a menor complexidade

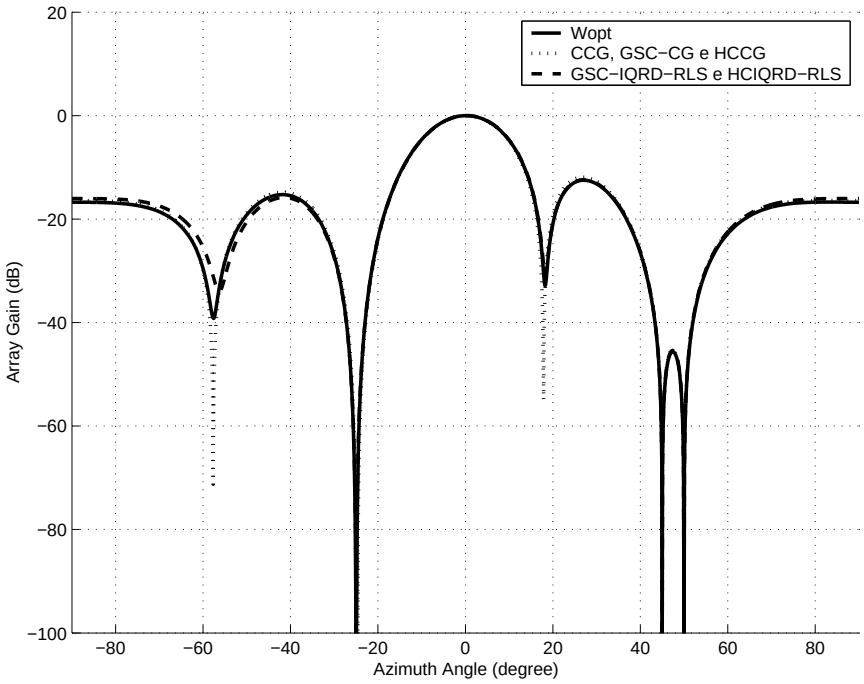

FIG. 4.5: *Beam–Pattern* para o caso de 20 iterações com um único experimento.

computacional. Este algoritmo apresenta não apenas a menor complexidade computacional mas também, como se pode constatar da FIG. 4.5, se ajusta rapidamente à solução ótima, juntamente com os algoritmos CCG (*Constrained Conjugate Gradient*), GSC-CG (*GSC Conjugate Gradient*) e os GSC-IQRD-RLS e HCIQRD-RLS, correspondendo às implementações do algoritmo QRD inverso nas estruturas GSC e Householder Constrained, respectivamente). Esta característica é conhecida em Inglês como *sample support*. O algoritmo QRD-RLS, também utilizado nas mesmas estruturas, não possui uma forma eficiente de determinação do vetor de coeficientes, a não ser por substituição regressiva (DINIZ, 2002), mas apresenta as mesmas características do IQRD-RLS, quanto à estabilidade e à velocidade de convergência.

4.5 RESUMO

Neste capítulo, foi avaliado o desempenho de vários algoritmos com restrições lineares de convergência rápida, especialmente aqueles baseados no algoritmo RLS. Foram incluídos vários algoritmos que usam rotações numericamente estáveis e dois outros não

baseados no RLS mas, igualmente, de convergência rápida.

Após vários testes, com um grande números de amostras, constatou-se que nenhum dos algoritmos com restrições em suas formas diretas (considerando-se apenas os de solução geral para aplicações LCMV), com a exceção do CCG (*Constrained Conjugate Gradient*), possui um desempenho aceitável no tocante à robustez. Isto nos leva à conclusão de que o uso de estruturas como a GSC e Householder Constrained, até o presente momento, são as únicas opções viáveis para a filtragem adaptativa com restrições usando algoritmos da família RLS.

Entre os algoritmos com restrições em suas formas diretas testados aqui, o único que apresentou um desempenho estável, além de uma convergência rápida, foi o algoritmo CCG. O algoritmo HCGG (*Householder Constrained Conjugate Gradient*), pela sua baixa complexidade computacional pode ser considerado uma boa opção para esta aplicação de *beamforming*. Quando implementados utilizando as estruturas GSC e Householder Constrained, os algoritmos CG, QRD-RLS e IQRD-RLS demonstraram boa robustez o que faz com que eles sejam considerados, também, alternativas viáveis para implementações práticas.

5.1 INTRODUÇÃO

Neste capítulo são discutidos algoritmos multicanais rápidos que usam a decomposição QR baseados no algoritmo RLS (MCFQRD-RLS — da sigla em inglês *Multichannel Fast QR Decomposition-Recursive Least Squares*). À semelhança das correspondentes versões monocanais, estes algoritmos podem ser classificados como na TAB. 2.5. Serão abordados aqui, para o caso multicanal, apenas aqueles que usam a atualização do erro de predição regressiva (*backward*), os algoritmos **FQRD_PRI_B** e **FQRD_POS_B**, por serem os casos em que se tem uma garantia de estabilidade (APOLINÁRIO JR., 1997, 2000a).

As primeiras versões multicanais para estes algoritmos foram desenvolvidas com base numa expansão natural do vetor de entrada, antes monocanal, para acomodar os sinais dos M canais de entrada. Como conseqüência disso, a abordagem matemática fica mais complexa. Os algoritmos baseados nesta abordagem, por conveniência, serão referidos doravante como *algoritmos multicanais em bloco*. Uma outra abordagem, possível no processamento de sinais multicanais usando os tipos de algoritmos tratados nesta dissertação, é aquela baseada no processamento seqüencial dos canais. No entanto, esta abordagem seqüencial a ser considerada ainda neste capítulo, usada em (RONTOGIANNIS, 1998), é um pouco mais complexa, porém mais atraente em outros aspectos do que aquela proposta em (BELLANGER, 1991).

As equações básicas para os algoritmos multicanais rápidos são apresentadas na Seção 5.2. Na Seção 5.3 é apresentado o algoritmo Multicanal Rápido *em blocos* baseado na atualização do vetor de erros de predição regressiva *a priori* (RONTOGIANNIS, 1998) e em seguida, na Seção 5.4, é apresentado o algoritmo Multicanal Rápido *em blocos* baseado na atualização do vetor de erros de predição regressiva *a posteriori* (BELLANGER, 1991; MEDINA S., 2002), onde uma nova equação é proposta para otimizar a versão *em blocos* deste algoritmo. Na Seção 5.5, uma versão em treliça para este algoritmo é introduzida. Na Seção 5.6, é visto o caso dos algoritmos multicanais rápidos para canais de ordens diferentes, com base na abordagem do processamento seqüencial, onde também são propostas duas novas versões para o algoritmo multicanal rápido baseado na atualização do

vetor de erros de predição regressiva *a posteriori*.

5.2 O ALGORITMO FQRD MULTICANAL BASEADO EM ERROS DE PREDIÇÃO REGRESSIVA: EQUAÇÕES BÁSICAS

A função objetivo a ser minimizada, de acordo com o algoritmo de mínimos quadrados (*Least Squares*–LS), é definida como

$$\xi_{LS}(k) = \sum_{i=0}^{k} \lambda^{k-i} |e(i)|^2 = \boldsymbol{e}^H(k)\boldsymbol{e}(k), \tag{5.1}$$

onde $\boldsymbol{e}(k) = \begin{bmatrix} e(k) & \lambda^{1/2}e(k-1) & \cdots & \lambda^{k/2}e(0) \end{bmatrix}^T$ é um vetor de erro que pode ser representado como

$$\begin{aligned} \boldsymbol{e}(k) &= \begin{bmatrix} d(k) \\ \lambda^{1/2}d(k-1) \\ \vdots \\ \lambda^{k/2}d(0) \end{bmatrix} - \begin{bmatrix} \boldsymbol{x}_N^T(k) \\ \lambda^{1/2}\boldsymbol{x}_N^T(k-1) \\ \vdots \\ \lambda^{k/2}\boldsymbol{x}_N^T(0) \end{bmatrix} \boldsymbol{w}_N^*(k) \\ &= \boldsymbol{d}(k) - \boldsymbol{X}_N^T(k)\boldsymbol{w}_N^*(k) \end{aligned} \tag{5.2}$$

e o seu complexo conjugado por

$$\begin{aligned} \boldsymbol{e}^*(k) &= \begin{bmatrix} d^*(k) \\ \lambda^{1/2}d^*(k-1) \\ \vdots \\ \lambda^{k/2}d^*(0) \end{bmatrix} - \begin{bmatrix} \boldsymbol{x}_N^H(k) \\ \lambda^{1/2}\boldsymbol{x}_N^H(k-1) \\ \vdots \\ \lambda^{k/2}\boldsymbol{x}_N^H(0) \end{bmatrix} \boldsymbol{w}_N(k) \\ &= \boldsymbol{d}^*(k) - \boldsymbol{X}_N^H(k)\boldsymbol{w}_N(k) \end{aligned} \tag{5.3}$$

onde, como pode ser visto na FIG. 5.1,

$$\boldsymbol{x}_N^H(k) = \begin{bmatrix} \boldsymbol{x}_k^H & \boldsymbol{x}_{k-1}^H & \cdots & \boldsymbol{x}_{k-N+1}^H \end{bmatrix} \tag{5.4}$$

e $\boldsymbol{x}_k^H = [x_1^*(k) \quad x_2^*(k) \quad \cdots \quad x_M^*(k)]$ é o vetor de entrada no instante k. Note que N é definido aqui como a ordem ou o número de coeficientes do filtro por canal, M é o número de canais de entrada e $\boldsymbol{w}_N(k)$ é o vetor de coeficientes de tamanho $MN \times 1$, no instante de tempo k.

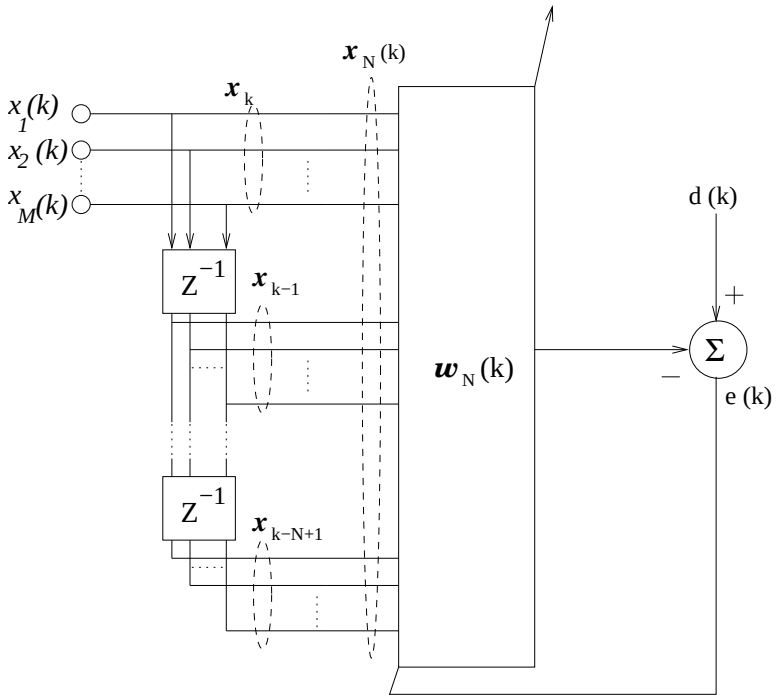

FIG. 5.1: Filtro Adaptativo Multicanal

Se a matriz $\boldsymbol{U}_N(k)$, de tamanho $MN \times MN$, for o fator de Cholesky da matriz de autocorrelação do sinal de entrada $\boldsymbol{R}(k) = \boldsymbol{X}_N(k)\boldsymbol{X}_N^H(k)$, i.e., $\boldsymbol{U}_N^H(k)\boldsymbol{U}_N(k) = \boldsymbol{X}_N(k)\boldsymbol{X}_N^H(k)$, obtido através da matriz de rotações de Givens, $\boldsymbol{Q}_N(k)$, então, a partir da EQ. 5.3, pode-se escrever

$$\begin{aligned} \boldsymbol{e}_q(k) &= \boldsymbol{Q}_N(k)\boldsymbol{e}^*(k) \\ &= \begin{bmatrix} \boldsymbol{e}_{q1}(k) \\ \boldsymbol{e}_{q2}(k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{d}_{q1}(k) \\ \boldsymbol{d}_{q2}(k) \end{bmatrix} - \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{U}_N(k) \end{bmatrix} \boldsymbol{w}_N(k), \end{aligned} \tag{5.5}$$

onde $\boldsymbol{e}_q(k)$ corresponde ao conjugado do vetor de erro rotacionado e $\boldsymbol{d}_q(k) = [\boldsymbol{d}_{q1}(k) \quad \boldsymbol{d}_{q2}(k)]^T$ é o conjugado do vetor com o sinal de referência rotacionado.

Na EQ. 5.5 observa-se que $\boldsymbol{e}_{q1}(k) = \boldsymbol{d}_{q1}(k)$, e que, fazendo $\boldsymbol{e}_{q2}(k) = 0$, obtém-se

$$\boldsymbol{U}_N(k)\boldsymbol{w}_N(k) = \boldsymbol{d}_{q2}(k),$$

o que implica em

$$\boldsymbol{w}_N(k) = \boldsymbol{U}_N^{-1}(k)\boldsymbol{d}_{q2}(k). \tag{5.6}$$

O resultado da EQ. 5.6 é importante e será usado mais adiante. Observe que a matriz resultado da decomposição de Cholesky, $\boldsymbol{U}_N(k)$, é triangular e que, em conseqüência disso, o cálculo da sua inversa não é necessário para se obter $\boldsymbol{w}_N(k)$ (DINIZ, 2002).

A partir da definição do problema da predição progressiva (*forward*) num ambiente multicanal, a matriz $\boldsymbol{X}_{N+1}(k)$ pode ser particionada da seguinte maneira:

$$\boldsymbol{X}_{N+1}^H(k) = \left[\begin{array}{c|c} \boldsymbol{D}_f(k) & \begin{array}{c}\boldsymbol{X}_N^H(k-1)\\ \boldsymbol{0}^T\end{array} \\ \hline \multicolumn{2}{c}{\boldsymbol{0}_{(M-1)\times(MN+M)}} \end{array}\right], \tag{5.7}$$

na qual $\boldsymbol{D}_f(k) = [\boldsymbol{x}_k \quad \lambda^{1/2}\boldsymbol{x}_{k-1} \cdots \lambda^{k/2}\boldsymbol{x}_0]^H$ é o sinal de referência progressiva de tamanho $(k+1)\times M$ e o subscrito $_{N+1}$ corresponde ao problema da $(N+1)$-*ésima* ordem.

O processo de triangularização de $\boldsymbol{X}_{N+1}^H(k)$ levando a $\boldsymbol{U}_{N+1}(k)$ (matriz triangular inferior uma vez que, na notação utilizada nesta dissertação, corresponde à atualização dos erros de predição regressiva ou *backward*) é realizado aplicando rotações de Givens a EQ. 5.7 como segue.

$$\begin{aligned}
\left[\begin{array}{c}\boldsymbol{0}\\ \boldsymbol{U}_{N+1}(k)\end{array}\right] &= \boldsymbol{Q}_f'(k)\boldsymbol{Q}_f(k)\left[\begin{array}{cc}\boldsymbol{Q}(k-1) & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{I}_{M\mathbf{x}M}\end{array}\right]\boldsymbol{X}_{N+1}^H(k)\\
&= \boldsymbol{Q}_f'(k)\boldsymbol{Q}_f(k)\left[\begin{array}{cc}\boldsymbol{Q}(k-1) & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{I}_{M\mathbf{x}M}\end{array}\right]\\
&\qquad \times\left[\begin{array}{cc}\boldsymbol{D}_f(k) & \boldsymbol{X}_N^H(k-1)\\ & \boldsymbol{0}^T\\ \multicolumn{2}{c}{\boldsymbol{0}_{(M-1)\times(MN+M)}}\end{array}\right]\\
&= \boldsymbol{Q}_f'(k)\boldsymbol{Q}_f(k)\left[\begin{array}{cc}\boldsymbol{E}_{fq1}(k) & \boldsymbol{0}\\ \boldsymbol{D}_{fq2}(k) & \boldsymbol{U}_N(k-1)\\ \lambda^{1/2}\boldsymbol{x}_0^T & \boldsymbol{0}^T\\ \multicolumn{2}{c}{\boldsymbol{0}_{(M-1)\times(MN+M)}}\end{array}\right]\\
&= \boldsymbol{Q}_f'(k)\left[\begin{array}{cc}\boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{D}_{fq2}(k) & \boldsymbol{U}_N(k-1)\\ \boldsymbol{E}_f(k) & \boldsymbol{0}\end{array}\right].
\end{aligned} \tag{5.8}$$

A ordem das matrizes na EQ. 5.8 cresce com o tempo. Em conseqüência disso, haverá uma seção nula crescente nas mesmas, mas que pode ser removida, resultando em

$$\boldsymbol{U}_{N+1}(k) = \boldsymbol{Q}'_{\theta f}(k) \begin{bmatrix} \boldsymbol{D}_{fq2}(k) & \boldsymbol{U}_N(k-1) \\ \boldsymbol{E}_f(k) & \mathbf{0} \end{bmatrix}. \tag{5.9}$$

Na EQ. 5.9, $\boldsymbol{Q}'_{\theta f}(k)$ é uma matriz de ordem fixa obtida de $\boldsymbol{Q}'_f(k)$. A matriz $\boldsymbol{Q}'_{\theta f}(k)$ de tamanho $(MN+M)\times(MN+M)$ contém as rotações de Givens que anulam $\boldsymbol{D}_{fq2}(k)$ sobre a diagonal de $\boldsymbol{E}_f(k)$ que corresponde à matriz de covariância do erro de predição progressiva de tamanho $M \times M$.

Com base na equação anterior, é possível obter

$$[\boldsymbol{U}_{N+1}(k)]^{-1} = \begin{bmatrix} \mathbf{0} & \boldsymbol{E}_f^{-1}(k) \\ \boldsymbol{U}_N^{-1}(k-1) & -\boldsymbol{U}_N^{-1}(k-1)\boldsymbol{D}_{fq2}(k)\boldsymbol{E}_f^{-1}(k) \end{bmatrix} \boldsymbol{Q}'^{H}_{\theta f}(k). \tag{5.10}$$

A EQ. 5.10 será usada na próxima seção para derivar uma expressão para a atualização do vetor de erros *a posteriori* de predição regressiva.

Ainda da EQ. 5.9, pode-se escrever

$$\begin{bmatrix} \mathbf{0} \\ \boldsymbol{E}_f^0(k+1) \end{bmatrix} = \boldsymbol{Q}'_{\theta f}(k+1) \begin{bmatrix} \boldsymbol{D}_{fq2}(k+1) \\ \boldsymbol{E}_f(k+1) \end{bmatrix}, \tag{5.11}$$

onde $\boldsymbol{E}_f^0(k+1)$ corresponde à matriz de covariância do erro de predição progressiva de ordem zero.

A EQ. 5.11 se justifica já que, para determinar os ângulos das rotações de Givens, $\boldsymbol{Q}'_{\theta f}(k)$, na EQ. 5.9 basta saber que estes devem anular $\boldsymbol{D}_{fq2}(k)$ sobre a diagonal de $\boldsymbol{E}_f(k)$.

Sabendo que $\boldsymbol{Q}_f(k)$ e $\boldsymbol{Q}(k-1)$ podem ser representadas como

$$\boldsymbol{Q}_f(k) = \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}_f(k-1) \end{bmatrix} \tag{5.12}$$

e

$$\boldsymbol{Q}(k-1) = \hat{\boldsymbol{Q}}(k-1) \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}(k-2) \end{bmatrix}, \tag{5.13}$$

respectivamente, e também que

$$\begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}_f(k-1) \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{I}_{M\times M} \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{I}_{M\times M} \end{bmatrix} \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}_f(k-1) \end{bmatrix}, \tag{5.14}$$

pode-se escrever, da EQ. 5.8,

$$
\begin{aligned}
\begin{bmatrix} \mathbf{0} \\ \boldsymbol{D}_{fq2}(k) \\ \boldsymbol{E}_f(k) \end{bmatrix} &= \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}_f(k-1) \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{I}_{M\times M} \end{bmatrix} \\
&\quad \times \begin{bmatrix} 1 & \mathbf{0}^T & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}(k-2) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{I}_{M\times M} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k^H \\ \lambda^{1/2}\boldsymbol{x}_{k-1}^H \\ \vdots \\ \lambda^{k/2}\boldsymbol{x}_0^T \\ \mathbf{0}_{(M-1)\times M} \end{bmatrix} \\
&= \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{I}_{M\times M} \end{bmatrix} \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}_f(k-1) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k^H \\ \lambda^{1/2}\boldsymbol{E}_{fq1}(k-1) \\ \lambda^{1/2}\boldsymbol{D}_{fq2}(k-1) \\ \lambda^{k/2}\boldsymbol{x}_0^H \\ \mathbf{0}_{(M-1)\times M} \end{bmatrix} \\
&= \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{I}_{M\times M} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k^H \\ \mathbf{0} \\ \lambda^{1/2}\boldsymbol{D}_{fq2}(k-1) \\ \lambda^{1/2}\boldsymbol{E}_f(k-1) \end{bmatrix} \\
&= \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) \begin{bmatrix} \boldsymbol{x}_k^H \\ \mathbf{0} \\ \lambda^{1/2}\boldsymbol{D}_{fq2}(k-1) \end{bmatrix} \\ \lambda^{1/2}\boldsymbol{E}_f(k-1) \end{bmatrix} \\
&= \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} \tilde{\boldsymbol{e}}_{fq1}^H(k) \\ \mathbf{0} \\ \lambda^{1/2}\boldsymbol{D}_{fq2}(k) \\ \lambda^{1/2}\boldsymbol{E}_f(k-1) \end{bmatrix}, \qquad (5.15)
\end{aligned}
$$

onde $\tilde{\boldsymbol{e}}_{fq1}^H(k)$ corresponde à primeira linha de $\boldsymbol{E}_{fq1}(k)$ da EQ. 5.8.

Empregando novamente a mesma EQ. 5.8, pode-se obter

$$
\begin{bmatrix} \mathbf{0}^T \\ \boldsymbol{E}_f(k+1) \end{bmatrix} = \overline{\boldsymbol{Q}}_f(k+1) \begin{bmatrix} \widetilde{\boldsymbol{e}}_{fq1}^H(k+1) \\ \lambda^{1/2}\boldsymbol{E}_f(k) \end{bmatrix}, \qquad (5.16)
$$

onde $\overline{\boldsymbol{Q}}_f(k+1)$ é uma matriz de ordem fixa obtida da matriz ortogonal $\boldsymbol{Q}_f(k+1)$, responsável por zerar $\widetilde{\boldsymbol{e}}^H_{fq1}(k+1)$ sobre $\lambda^{1/2}\boldsymbol{E}_f(k)$.

Da EQ. 5.15, pode-se extrair a relação seguinte

$$\begin{bmatrix} \widetilde{\boldsymbol{e}}^H_{fq1}(k+1) \\ \boldsymbol{D}_{fq2}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} \boldsymbol{x}^H_{k+1} \\ \lambda^{1/2}\boldsymbol{D}_{fq2}(k) \end{bmatrix}, \tag{5.17}$$

onde $\tilde{\boldsymbol{e}}^H_{fq1}(k+1)$ é a primeira linha de $\boldsymbol{E}_{fq1}(k+1)$. $\boldsymbol{Q}_\theta(k)$ é uma matriz de ordem fixa obtida de $\boldsymbol{Q}_N(k)$ de tamanho $(N+1)\times(N+1)$.

O processo de estimação conjunta é realizado usando-se a seguinte expressão (DINIZ, 2002; N.KALOUPTSIDIS, 1993):

$$\begin{bmatrix} e_{q1}(k+1) \\ \boldsymbol{d}_{q2}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta(k+1) \begin{bmatrix} d^*(k+1) \\ \lambda^{1/2}\boldsymbol{d}_{q2}(k) \end{bmatrix}. \tag{5.18}$$

O erro *a priori* é dado por

$$\varepsilon(k) = e^*_{q1}(k)/\gamma(k) = e(k)/\gamma^2(k). \tag{5.19}$$

A expressão necessária para a obtenção de $\boldsymbol{Q}_\theta(k+1)$ depende do tipo de vetor de erro a ser atualizado (*a priori* ou *a posteriori*) e será introduzida nas duas seções seguintes.

5.3 O ALGORITMO *FAST* QRD MULTICANAL BASEADO NA ATUALIZAÇÃO DO ERRO DE PREDIÇÃO REGRESSIVA *A PRIORI* (MCFQRD_PRI_B)

Uma abordagem deste algoritmo pode ser encontrada em (RONTOGIANNIS, 1998), onde a sua versão em treliça foi originalmente apresentada.

Da EQ. 2.62, definida no Capítulo 2 para o caso monocanal, pode-se redefinir o vetor de erros de predição regressiva ou *backward a priori*, $\boldsymbol{a}(k)$, para o caso multicanal como

$$\boldsymbol{a}_{N+1}(k+1) = \boldsymbol{U}^{-H}_{N+1}(k)\boldsymbol{x}_{N+1}(k+1). \tag{5.20}$$

Combinando a expressão acima com a EQ. 5.10, obtém-se

$$
\begin{aligned}
\boldsymbol{a}_{N+1}(k+1) &= \lambda^{-1/2}\boldsymbol{Q}'_{\theta f}(k) \\
&\quad \times \begin{bmatrix} \mathbf{0} & \boldsymbol{U}_N^{-H}(k-1) \\ \boldsymbol{E}_f^{-H}(k) & -\boldsymbol{E}_f^{-H}(k)\boldsymbol{D}_{fq2}(k)\boldsymbol{U}_N^{-H}(k-1) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{x}_N(k) \end{bmatrix} \\
&= \lambda^{-1/2}\boldsymbol{Q}'_{\theta f}(k) \\
&\quad \times \begin{bmatrix} \boldsymbol{U}_N^{-H}(k-1)\boldsymbol{x}_N(k) \\ \boldsymbol{E}_f^{-H}(k)\boldsymbol{x}_{k+1} - \boldsymbol{E}_f^{-H}(k)\boldsymbol{D}_{fq2}^{H}(k)\boldsymbol{U}_N^{-H}(k-1)\boldsymbol{x}_N(k) \end{bmatrix} \\
&= \lambda^{-1/2}\boldsymbol{Q}'_{\theta f}(k) \begin{bmatrix} \boldsymbol{a}_N(k) \\ \boldsymbol{r}(k+1) \end{bmatrix}, \qquad (5.21)
\end{aligned}
$$

onde

$$
\begin{aligned}
\boldsymbol{r}(k+1) &= \lambda^{-1/2}\boldsymbol{E}_f^{-H}(k)\boldsymbol{x}_{k+1} - \boldsymbol{E}_f^{-H}(k)\boldsymbol{D}_{fq2}^{H}(k)\boldsymbol{U}_N^{-H}(k-1)\boldsymbol{x}_N(k) \\
&= \lambda^{-1/2}\boldsymbol{E}_f^{-H}(k)\left[\boldsymbol{x}_{k+1} - \left[\boldsymbol{U}_N^{-1}(k-1)\boldsymbol{D}_{fq2}(k)\right]^H \boldsymbol{x}_N(k)\right] \\
&= \lambda^{-1/2}\boldsymbol{E}_f^{-H}(k)\left[\boldsymbol{x}_{k+1} - \boldsymbol{W}_f^{H}(k)\boldsymbol{x}_N(k)\right] \\
&= \lambda^{-1/2}\boldsymbol{E}_f^{-H}(k)\tilde{\boldsymbol{e}}'_f(k+1), \qquad (5.22)
\end{aligned}
$$

com $\tilde{\boldsymbol{e}}'_f(k+1)$ sendo o vetor de erros *a priori* progressivos. A atualização de $\boldsymbol{Q}_\theta(k)$ é feita através da expressão abaixo, similar à sua correspondente unidimensional (mono canal) (REGALIA, 1991).

$$
\begin{bmatrix} 1/\gamma(k+1) \\ \mathbf{0} \end{bmatrix} = \boldsymbol{Q}_\theta(k+1) \begin{bmatrix} 1 \\ -\boldsymbol{a}_N(k+1) \end{bmatrix}. \qquad (5.23)
$$

Para calcular $\boldsymbol{r}(k+1)$ usando a EQ. 5.22 é preciso obter uma matriz inversa. Além da complexidade computacional que advém dessa operação, o uso da matriz inversa pode gerar problemas de instabilidade numérica para matrizes mal condicionadas. Uma alternativa para contornar esse problema é usar a expressão abaixo, obtida da EQ. 5.16 (RONTOGIANNIS, 1998).

$$
\begin{bmatrix} * \\ \mathbf{0} \end{bmatrix} = \overline{\boldsymbol{Q}}_f(k+1) \begin{bmatrix} 1/\gamma(k+1) \\ -\boldsymbol{r}(k+1) \end{bmatrix} \qquad (5.24)
$$

O escalar representado pelo *asterisco* não precisa ser conhecido para o efeito do cálculo de $\boldsymbol{r}(k+1)$ uma vez conhecidos $\overline{\boldsymbol{Q}}_f(k+1)$ e $\gamma(k+1)$. As equações do algoritmo MCFQRD_PRI_B estão na TAB. 5.1.

5.4 O ALGORITMO *FAST* QRD MULTICANAL BASEADO NA ATUALIZAÇÃO DO ERRO DE PREDIÇÃO REGRESSIVA *A POSTERIORI* (MCFQRD_POS_B)

Este algoritmo foi originalmente proposto em (BELLANGER, 1991) e posteriormente abordado em (MEDINA S., 2002). Na abordagem do mesmo a ser feita nesta seção, será introduzida uma nova expressão que otimiza a implementação "em bloco" deste algoritmo.

A definição do vetor de erros de predição regressiva *ou backward a posteriori* dada pela EQ. 2.60, vista no Capítulo 2 para o caso monocanal, pode ser redefinida para o caso multicanal como

$$\boldsymbol{f}_{N+1}(k+1) = \boldsymbol{U}_{N+1}^{-H}(k+1)\boldsymbol{x}_{N+1}(k+1). \tag{5.25}$$

Combinando a expressão acima com a EQ. 5.10, obtém-se

$$\begin{aligned}
\boldsymbol{f}_{N+1}(k+1) &= \boldsymbol{Q}'_{\theta f}(k+1) \\
&\quad \times \begin{bmatrix} \boldsymbol{0} & \boldsymbol{U}_N^{-H}(k) \\ \boldsymbol{E}_f^{-H}(k+1) & -\boldsymbol{E}_f^{-H}(k+1)\boldsymbol{D}_{fq2}(k+1)\boldsymbol{U}_N^{-H}(k) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{x}_N(k) \end{bmatrix} \\
&= \boldsymbol{Q}'_{\theta f}(k+1) \\
&\quad \times \begin{bmatrix} \boldsymbol{U}_N^{-H}(k)\boldsymbol{x}_N(k) \\ \boldsymbol{E}_f^{-H}(k+1)\boldsymbol{x}_{k+1} - \boldsymbol{E}_f^{-H}(k+1)\boldsymbol{D}_{fq2}^H(k+1)\boldsymbol{U}_N^{-H}(k)\boldsymbol{x}_N(k) \end{bmatrix} \\
&= \boldsymbol{Q}'_{\theta f}(k+1) \begin{bmatrix} \boldsymbol{f}_N(k) \\ \boldsymbol{p}(k+1) \end{bmatrix},
\end{aligned} \tag{5.26}$$

onde

$$\begin{aligned}
\boldsymbol{p}(k+1) &= \boldsymbol{E}_f^{-H}(k+1)\boldsymbol{x}_{k+1} - \boldsymbol{E}_f^{-H}(k+1)\boldsymbol{D}_{fq2}^H(k+1)\boldsymbol{U}_N^{-H}(k)\boldsymbol{x}_N(k) \\
&= \boldsymbol{E}_f^{-H}(k+1)\left[\boldsymbol{x}_{k+1} - \left[\boldsymbol{U}_N^{-1}(k)\boldsymbol{D}_{fq2}(k+1)\right]^H \boldsymbol{x}_N(k)\right] \\
&= \boldsymbol{E}_f^{-H}(k+1)\left[\boldsymbol{x}_{k+1} - \boldsymbol{W}_f^H(k+1)\boldsymbol{x}_N(k)\right] \\
&= \boldsymbol{E}_f^{-H}(k+1)\tilde{\boldsymbol{e}}_f(k+1),
\end{aligned} \tag{5.27}$$

com $\tilde{\boldsymbol{e}}_f(k+1)$ sendo o vetor de erros *a posteriori* progressivos. A atualização de $\boldsymbol{Q}_\theta(k)$ é feita através da expressão abaixo, similar à sua correspondente unidimensional (mono canal) (BELLANGER, 1991).

$$\boldsymbol{Q}_\theta(k+1)\begin{bmatrix} 1 \\ \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \gamma(k+1) \\ \boldsymbol{f}_N(k+1) \end{bmatrix} \tag{5.28}$$

Como na seção anterior, é necessária uma expressão para calcular $\boldsymbol{p}(k+1)$ sem usar a EQ. 5.27 que requer uma operação de inversão de matriz. A alternativa é usar a expressão abaixo, obtida da EQ. 5.16, cuja demonstração é apresentada logo em seguida.

$$\overline{\boldsymbol{Q}}_f(k+1)\begin{bmatrix}\gamma(k)\\ \mathbf{0}\end{bmatrix}=\begin{bmatrix}*\\ \boldsymbol{p}(k+1)\end{bmatrix} \tag{5.29}$$

Note que o escalar representado pelo *asterisco* não precisa ser conhecido para o efeito do cálculo de $\boldsymbol{p}(k+1)$, uma vez conhecidos $\overline{\boldsymbol{Q}}_f(k+1)$ e $\gamma(k)$.

Prova: Da EQ. 5.16, vê-se que $\boldsymbol{E}_f(k+1)$ é o fator de Cholesky de $\begin{bmatrix}\tilde{\boldsymbol{e}}_{fq1} & \lambda^{1/2}\boldsymbol{E}_f^H(k)\end{bmatrix}\begin{bmatrix}\tilde{\boldsymbol{e}}_{fq1} & \lambda^{1/2}\boldsymbol{E}_f^H(k)\end{bmatrix}^H$ (GOLUB, 1983). Conseqüentemente, pode-se escrever

$$\begin{aligned}\boldsymbol{E}_f^H(k+1)\boldsymbol{E}_f(k+1) &= \begin{bmatrix}\tilde{\boldsymbol{e}}_{fq1}^H(k+1)\\ \lambda^{1/2}\boldsymbol{E}_f(k)\end{bmatrix}^H\overline{\boldsymbol{Q}}_f^H(k+1)\overline{\boldsymbol{Q}}_f(k+1)\begin{bmatrix}\tilde{\boldsymbol{e}}_{fq1}^H(k+1)\\ \lambda^{1/2}\boldsymbol{E}_f(k)\end{bmatrix}\\ &= \tilde{\boldsymbol{e}}_{fq1}(k+1)\tilde{\boldsymbol{e}}_{fq1}^H(k+1)+\lambda\boldsymbol{E}_f^H(k)\boldsymbol{E}_f(k).\end{aligned} \tag{5.30}$$

A equação acima corresponde à EQ. 5.16 na forma de produto. Pré-multiplicando e pós-multiplicando a EQ. 5.30 por $\boldsymbol{E}_f^{-H}(k+1)\gamma^2(k)$ e $\boldsymbol{E}_f^{-1}(k+1)$, respectivamente, após algumas manipulações algébricas, obtém-se

$$\gamma^2(k)\boldsymbol{I}=\boldsymbol{p}(k+1)\boldsymbol{p}^H(k+1)+\boldsymbol{\Psi}, \tag{5.31}$$

onde $\boldsymbol{\Psi}=\lambda\gamma^2(k)\boldsymbol{E}_f^{-H}(k+1)\boldsymbol{E}_f^H(k)\boldsymbol{E}_f(k)\boldsymbol{E}_f^{-1}(k+1)$.

Finalmente, pré-multiplicando e pós- multiplicando a EQ. 5.31 por $\boldsymbol{p}^H(k+1)$ e $\boldsymbol{p}(k+1)$, respectivamente, chega-se a

$$\begin{aligned}\gamma^2(k) &= \boldsymbol{p}^H(k+1)\boldsymbol{p}(k+1)+\frac{\boldsymbol{p}^H(k+1)\boldsymbol{\Psi}\boldsymbol{p}(k+1)}{\boldsymbol{p}^H(k+1)\boldsymbol{p}(k+1)}\\ &= \boldsymbol{p}^H(k+1)\boldsymbol{p}(k+1)+*^2.\end{aligned} \tag{5.32}$$

A expressão na EQ. 5.32 pode ser encarada como um produto de Cholesky. Portanto, existe uma matriz ortogonal $\boldsymbol{Q}$ tal que

$$\begin{bmatrix}\gamma(k)\\ \mathbf{0}\end{bmatrix}=\boldsymbol{Q}\begin{bmatrix}*\\ \boldsymbol{p}(k+1)\end{bmatrix}. \tag{5.33}$$

Agora, recordando o ponto de partida na EQ. 5.16, percebe-se que $\boldsymbol{Q}$ está relacionada com $\overline{\boldsymbol{Q}}_f(k+1)$. Mais especificamente, as rotações de Givens que satisfazem a relação entre a EQ. 5.16 e a EQ. 5.33 são únicas (GOLUB, 1983). Na EQ. 5.16 nota-se que $\overline{\boldsymbol{Q}}_f(k+1)$

é uma seqüencia de rotações de Givens responsável por anular uma linha de uma matriz. Para o caso da EQ. 5.33 em que uma coluna deve ser zerada, deve-se usar $\boldsymbol{Q} = \overline{\boldsymbol{Q}}_f^H(k+1)$. Substituindo-se $\boldsymbol{Q}$ por $\overline{\boldsymbol{Q}}_f^H(k+1)$ na EQ. 5.33 e empregando o fato de que $\overline{\boldsymbol{Q}}_f(k+1)$ é ortogonal pode-se obter a EQ. 5.29 pré–multiplicando o resultado por $\overline{\boldsymbol{Q}}_f(k+1)$. □

O algoritmo MCFQRD_POS_B completo está resumido na TAB. 5.2. A principal diferença da versão deste algoritmo proposta aqui para o apresentado em (BELLANGER, 1991) é a introdução da EQ. 5.29 que permite a execução em bloco do mesmo sem que se tenha que lidar com relações muito complexas ou inversão de matrizes, seja qual for o número de canais. Em (BELLANGER, 1991) o autor recomenda uma execução seqüencial (canal após canal) para os casos em que o número de canais é maior que 2.

5.5 O NOVO ALGORITMO: MCFQD_POS_B: VERSÃO EM TRELIÇA

Nesta seção, é introduzida uma versão em treliça do algoritmo visto na seção anterior. O fato deste algoritmo ser recursivo na ordem favorece a sua implementação em *arrays* sistólicos consumindo menos recursos computacionais.

Em virtude da natureza em blocos do vetor de entrada usado na derivação das equações do algoritmo apresentado na seção anterior, as quantidades $\boldsymbol{D}_{fq2}(k)$, $\boldsymbol{d}_{q2}(k)$ e $\boldsymbol{f}_N(k)$ podem ser particionadas em N blocos. Em particular, a matriz $\boldsymbol{D}_{fq2}(k)$ pode ser escrita da seguinte maneira:

$$\boldsymbol{D}_{fq2}(k) = \begin{bmatrix} \boldsymbol{D}_{fq2}^{(1)}(k) \\ \vdots \\ \boldsymbol{D}_{fq2}^{(N)}(k) \end{bmatrix}, \tag{5.34}$$

onde $\boldsymbol{D}_{fq2}^{(i)}(k)$ tem tamanho $M \times M$. Usando o particionamento citado acima, pode-se reescrever a EQ. 5.11 da seguinte maneira

$$\begin{bmatrix} \boldsymbol{0}_{M(N-i-1)\times M} \\ \boldsymbol{0}_{M(i-1)\times M} \\ \boldsymbol{E}_f^{(i-1)}(k+1) \end{bmatrix} = \boldsymbol{Q}'_{\theta f}{}^{(N-i+1)}(k+1) \begin{bmatrix} \boldsymbol{0}_{M(N-i)\times M} \\ \boldsymbol{D}_{fq2}^{(N-i+1)}(k) \\ \boldsymbol{0}_{M(i-1)\times M} \\ \boldsymbol{E}_f^{(i)}(k+1) \end{bmatrix} \tag{5.35}$$

para $i = N, N-1, \cdots, 1$, pressupondo uma execução regressiva ou *backward*.

Da equação anterior, pode-se constatar que $\boldsymbol{Q}'_{\theta f}(k\ +\ 1)$ da EQ. 5.11 corresponde a

$$\boldsymbol{Q}'_{\theta f}(k\ +\ 1)\ \ =\ \ \boldsymbol{Q}'_{\theta f}{}^{(N)}(k+1)\boldsymbol{Q}'_{\theta f}{}^{(N-1)}(k+1)\cdots\boldsymbol{Q}'_{\theta f}{}^{(1)}(k+1).$$

Observando cuidadosamente a EQ. 5.35, verifica-se que ela também aceita uma execução progressiva ou *forward*, i.e., para $i = 1, 2, \cdots, N$. Esta propriedade, que é a chave para a derivação da versão em treliça do algoritmo apresentado na seção anterior, advém do fato de que cada uma das matrizes $\boldsymbol{Q}'_{\theta f}{}^{(N-i+1)}(k+1)$ na $EQ.$ 5.35 altera apenas uma partição $\boldsymbol{D}_{fq2}^{(N-i+1)}(k)$ $(i = 1, 2, \cdots, N)$ nesta equação, qualquer que seja a ordem de execução do produto no segundo membro da mesma.

A matriz $\boldsymbol{Q}'_{\theta f}(k+1)$ é usada para atualizar o vetor $\boldsymbol{f}_N(k)$, como na EQ. 5.26. Esta mesma equação pode ser reescrita também de forma particionada como

$$\begin{bmatrix} \mathbf{0}_{M(N-i)\times M} \\ \boldsymbol{f}^{(N-i+1)}(k+1) \\ \mathbf{0}_{M(i-1)\times M} \\ \boldsymbol{p}_{i-1}(k+1) \end{bmatrix} = \boldsymbol{Q}'_{\theta f}{}^{(N-i+1)}(k+1) \begin{bmatrix} \mathbf{0}_{M(N-i)\times M} \\ \boldsymbol{f}^{(N-i+2)}(k) \\ \mathbf{0}_{M(i-1)\times M} \\ \boldsymbol{p}_i(k+1) \end{bmatrix}, \tag{5.36}$$

para $i = 1, 2, \cdots, N$.

Com isso, já se tem uma estrutura de execução progressiva para todas as equações do algoritmo apresentado na seção anterior. As equações restantes não carecem de alterações. Os senos e cossenos dos ângulos das rotações em $\boldsymbol{Q}_\theta^{(i)}(k+1)$ são obtidos fazendo

$$\boldsymbol{Q}_\theta^{(i)}(k+1) \begin{bmatrix} \gamma_{i-1}(k+1) \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \gamma_i(k+1) \\ \boldsymbol{f}^{(N-i+2)}(k+1) \end{bmatrix} \tag{5.37}$$

e a estimação conjunta é efetuada de acordo com

$$\begin{bmatrix} \boldsymbol{e}_{q1}^{(i)}(k+1) \\ \boldsymbol{d}_{q2}^{(N-i+1)}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta^{(N-i+1)}(k+1) \begin{bmatrix} \boldsymbol{e}_{q1}^{(i)}(k+1) \\ \lambda^{1/2}\boldsymbol{d}_{q2}^{(N-i+1)}(k) \end{bmatrix} \tag{5.38}$$

ambas para $i = 1, 2, \cdots, N$.

Para adequar as equações das etapas 1–3, 6 e 7 do algoritmo da TAB. 5.2 a essa formulação, basta observar que estas equações também podem ser facilmente particionadas em blocos de tamanho $M \times M$ que serão também executados recursivamente como mostrado

na TAB. 5.3 onde, estão todas as equações do algoritmo proposto. Uma versão em treliça do algoritmo da tabela TAB. 5.1, descrito na Seção 5.3, está disponível em (RONTOGIANNIS, 1998).

Utilizou-se aqui uma notação matricial para simplificar a abordagem matemática. Porém, numa implementação, os produtos envolvendo matrizes ortogonais de rotações de Givens permitem que todas as equações sejam reduzidas a operações de escalares. A título de exemplo, é apresentado na TAB. 5.4 um pseudo–código da implementação da etapa 4 do algoritmo da TAB. 5.3.

Embora o algoritmo proposto aqui (RAMOS, 2004a) apresente as mesmas propriedades no tocante à convergência que o algoritmo de (RONTOGIANNIS, 1998), vale a pena ressaltar que, quanto à complexidade computacional, ele apresenta uma economia apreciável; isto o faz ser uma opção atraente para N e M grandes. Este aspecto será tratado mais detalhadamente no Capítulo 6, onde é avaliado o desempenho das várias versões dos algoritmos multicanais rápidos.

5.6 O ALGORITMO **FQRD** MULTICANAL PARA CANAIS DE ORDENS DIFERENTES

A principal característica dos algoritmos a serem abordados nesta seção prende-se ao fato de que eles podem ser utilizados em aplicações de filtragem adaptativa multicanais com canais de ordens diferentes ou não. Esta funcionalidade é alcançada partindo-se de uma nova abordagem para a construção do vetor de entrada, $\boldsymbol{x}_N$. O caso mais típico para a aplicação deste tipo de algoritmo é a filtragem de Volterra (MATHEWS, 2000).

Esta nova abordagem, além de ser mais geral, proporciona uma redução considerável na complexidade computacional em comparação com os algoritmos multicanais *em bloco* vistos anteriormente.

5.6.1 ASPECTOS GERAIS

Por comodidade, serão tratados aqui alguns aspectos que podem ser abordados conjuntamente e que servem como ponto de partida para a derivação dos dois tipos básicos de algoritmos: o que atualiza o vetor de erros regressivo *a priori* e o que atualiza o vetor de erros regressivo *a posteriori*. Este último, introduzido neste trabalho, será descrito detalhadamente e as equações do primeiro (RONTOGIANNIS, 1998) serão descritos re-

sumidamente.

A seguinte convenção é adotada:

M	Número de canais de entrada;
$N_1, N_2, \cdots, N_M$	Representam os números de *taps* de cada canal de entrada;
$N = \sum_{r=1}^{M} N_r$	

Para simplificar a abordagem, sem perda de generalidade, assume-se que $N_1 \geq N_2 \geq \cdots \geq N_M$.

O ponto de partida para a derivação deste algoritmo é a construção do vetor de entrada de tal maneira que o caso geral de canais de mesma ordem ou não seja atendido. Para tal, é levado em consideração o fato de que cada iteração deste algoritmo será executada em M etapas. Logo, o vetor de entrada sofrerá M atualizações por iteração recebendo assim as amostras mais recentes (instante $k+1$) de cada canal. Para um determinado instante de tempo k obtém-se o vetor $\boldsymbol{x}_N(k)$ a partir do qual se chega ao vetor $\boldsymbol{x}_{N+M}(k+1)$ após M sucessivas atualizações. Para a construção de $\boldsymbol{x}_N$, escolhe-se $N_1 - N_2$ amostras mais recentes do primeiro canal para serem os primeiros elementos de $\boldsymbol{x}_N$, seguidos pelos $N_2 - N_3$ pares de amostras do primeiro e do segundo canais, seguidos por $N_3 - N_4$ trios de amostras dos primeiros três canais e assim por diante até $N_M - N_{M+1}$ conjunto de M amostras de todos os canais. Assume-se que $N_{M+1} = 0$.

O procedimento acima foi aplicado no diagrama da FIG. 5.2 para uma configuração de $M = 3$, $N_1 = 4$, $N_2 = 3$, $N_3 = 2$ e $N_4 = 0$. Observando cuidadosamente este diagrama, pode-se antever a posição que será ocupada pelas novas amostras de cada canal. A posição da amostra mais recente do *i-ésimo* canal, aqui denotada por p_i, é dada de forma compacta pela seguinte expressão (RONTOGIANNIS, 1998),

$$p_i = \sum_{r=1}^{i-1} r(N_r - N_{r+1}) + i \qquad i = 1, 2, \cdots, M. \tag{5.39}$$

Formalmente, pode-se definir os M vetores de entrada, a partir de $\boldsymbol{x}_N(k)$, como:

$$\boldsymbol{x}_{N+1}^H(k+1) = \begin{bmatrix} x_1^*(k+1) & \boldsymbol{x}_N^H(k) \end{bmatrix} \tag{5.40}$$

$$\boldsymbol{x}_{N+i}^H(k+1) = \begin{bmatrix} x_i^*(k+1) & \boldsymbol{x}_{N+i-1}^H(k+1) \end{bmatrix} \boldsymbol{P}_i \tag{5.41}$$

onde $\boldsymbol{P}_i$ é uma matriz de permutação que desloca $x_i(k+1)$ à posição p_i após deslocar os primeiros $p_i - 1$ elementos de $\boldsymbol{x}_{N+i-1}^H(k+1)$ para a esquerda. Concluído o processo acima

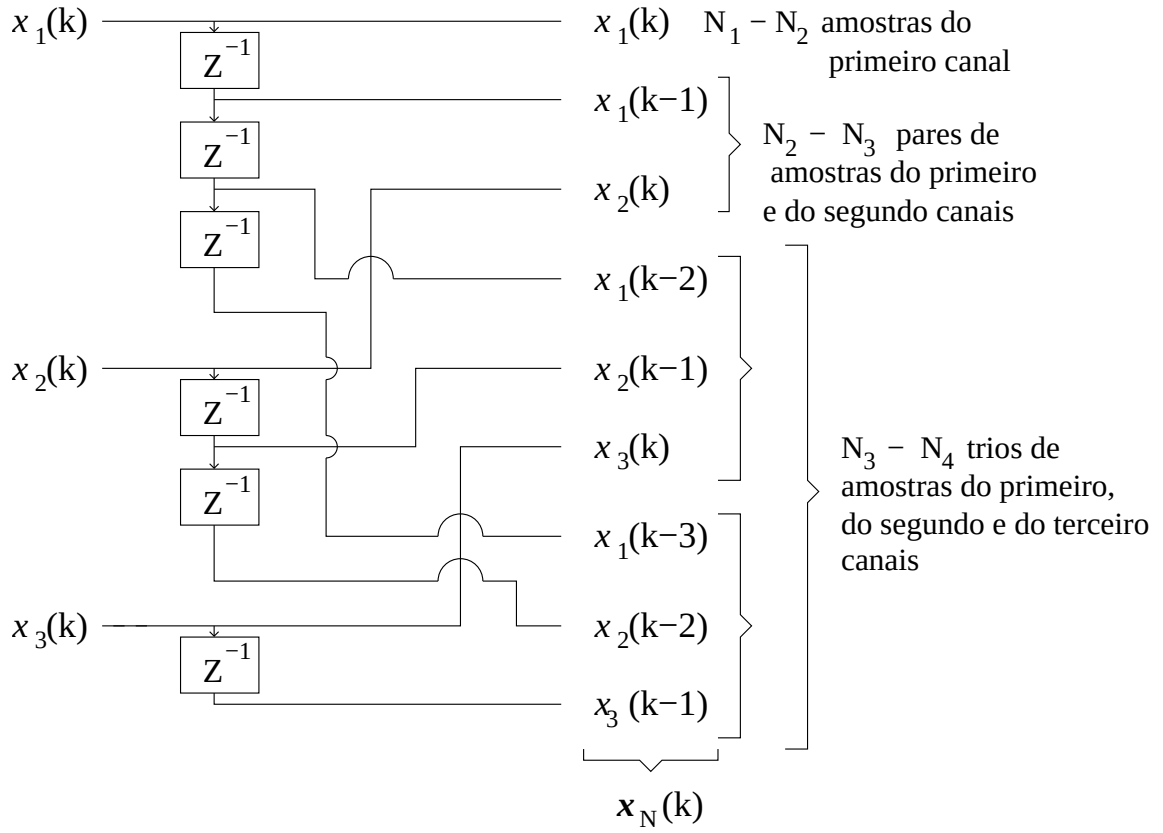

FIG. 5.2: Composição do novo vetor de entrada para $N_1 = 4$, $N_2 = 3$ e $N_3 = 2$.

para os M canais, pode-se observar que

$$\boldsymbol{x}_{N+M}^H(k+1) = \left[\boldsymbol{x}_N^H(k+1) \quad x_1^*(k-N_1+1) \quad \cdots \quad x_M^*(k-N_M+1)\right], \tag{5.42}$$

o que significa, claramente, que os primeiros N elementos de $x_{N+M}^H(k+1)$ formam o vetor de entrada da iteração seguinte.

Para o caso específico do exemplo apresentado na FIG. 5.2, $\boldsymbol{x}_{N+M}^H(k+1) = [x_1(k+1) \quad x_1(k) \quad x_2(k+1) \quad x_1(k-1) \quad x_2(k) \quad x_3(k+1) \quad x_1(k-2) \quad x_2(k-1) \quad x_3(k) \quad x_1(k-3) \quad x_2(k-2) \quad x_3(k-1)]^*$ de onde se obtém $\boldsymbol{x}_N^H(k+1) = [x_1(k+1) \quad x_1(k) \quad x_2(k+1) \quad x_1(k-1) \quad x_2(k) \quad x_3(k+1) \quad x_1(k-2) \quad x_2(k-1) \quad x_3(k)]^*$.

Pelo exposto até aqui, vê-se que, ao contrário do algoritmo em bloco apresentado na seção anterior, este procedimento seqüencial implica em que o processamento dos canais seja feito separadamente, mas de forma cumulativa, isto é, a informação coletada em cada amostra processada de cada canal é usada no processamento da próxima amostra do

próximo canal. Isto leva a uma definição distinta da matriz de informação e, conseqüentemente, do processo de triangularização da mesma. Mais precisamente, tem-se agora M matrizes de informação a cada iteração, definidas da seguinte maneira:

$$\boldsymbol{X}_{N+i}^{H}(k) = \begin{bmatrix} \boldsymbol{x}_{N+i}^{H}(k) \\ \lambda^{1/2}\boldsymbol{x}_{N+i}^{H}(k-1) \\ \vdots \\ \lambda^{k/2}\boldsymbol{x}_{N+i}^{H}(0) \end{bmatrix} \qquad i = 1, 2, \cdots, M. \tag{5.43}$$

Como conseqüência, pode-se concluir que a informação de cada uma dessas matrizes é utilizada na geração da próxima.

Seja $\boldsymbol{U}_{N+i}(k)$ o fator de Cholesky de $\boldsymbol{X}_{N+i}(k)\boldsymbol{X}_{N+i}^{H}(k)$. À semelhança do que foi feito na seção anterior, os dois vetores de interesse $\boldsymbol{a}_{N+i}$ e $\boldsymbol{f}_{N+i}$ podem ser definidos para os dois tipos de algoritmos, baseados na atualização dos erros de predição regressivos a *priori* e a *posteriori*, respectivamente, como

$$\boldsymbol{a}_{N+i}(k+1) = \lambda^{1/2}\boldsymbol{U}_{N+i}^{-H}(k)\boldsymbol{x}_{N+i}(k+1) \tag{5.44}$$

$$\boldsymbol{f}_{N+i}(k+1) = \boldsymbol{U}_{N+i}^{-H}(k+1)\boldsymbol{x}_{N+i}(k+1) \qquad i = 1, 2, \cdots, M. \tag{5.45}$$

Das EQ. 5.42, 5.44 e 5.45, pode-se concluir que

$$\boldsymbol{a}_{N+M}(k+1) = \begin{bmatrix} \boldsymbol{a}_{N}(k+1) \\ \boldsymbol{a}^{(N)}(k+1) \end{bmatrix} \tag{5.46}$$

$$\boldsymbol{f}_{N+M}(k+1) = \begin{bmatrix} \boldsymbol{f}_{N}(k+1) \\ \boldsymbol{f}^{(N)}(k+1) \end{bmatrix} \tag{5.47}$$

sendo que $\boldsymbol{a}^{(N)}(k+1)$ e $\boldsymbol{f}^{(N)}(k+1)$ são os últimos M elementos de $\boldsymbol{a}_{N+M}(k+1)$ e de $\boldsymbol{f}_{N+M}(k+1)$, respectivamente.

As atualizações de $\boldsymbol{a}_{N+i}(k+1)$ e de $\boldsymbol{f}_{N+i}(k+1)$ são realizadas em M etapas a cada iteração, como mostrado abaixo

$$\boldsymbol{a}_{N}(k) \rightarrow \boldsymbol{a}_{N+1}(k+1) \rightarrow \boldsymbol{a}_{N+2}(k+1) \rightarrow \cdots \rightarrow \boldsymbol{a}_{N+M}(k+1)$$

$$\boldsymbol{f}_{N}(k) \rightarrow \boldsymbol{f}_{N+1}(k+1) \rightarrow \boldsymbol{f}_{N+2}(k+1) \rightarrow \cdots \rightarrow \boldsymbol{f}_{N+M}(k+1).$$

As relações descritas até aqui formam a base para a derivação do algoritmo a ser introduzido na próxima sub-seção.

5.6.2 O ALGORITMO BASEADO NA ATUALIZAÇÃO DO VETOR DE ERROS *A POSTERIORI* REGRESSIVO — ESTRUTURA TRANSVERSAL

O algoritmo multicanal que será apresentado nesta seção é, até onde temos conhecimento, uma contribuição original desta dissertação e foi publicado em (RAMOS, 2004b). Basicamente, o ponto de partida para a derivação deste algoritmo é a construção do vetor de entrada com o formato apresentado nas EQ. 5.40 e EQ. 5.41. Usando a mesma abordagem, um algoritmo multicanal rápido, baseado na atualização do vetor de erros *a priori* regressivo, foi proposto em (RONTOGIANNIS, 1998).

O Processo de Triangularização da Matriz de Informação

A EQ. 5.43 sugere que a atualização da matriz de informação de entrada seja realizada em M etapas para cada iteração k.

Etapa I:

$\boldsymbol{X}_{N+1}^{H}(k)$ pode ser definido como

$$\begin{aligned}\boldsymbol{X}_{N+1}^{H}(k) &= \begin{bmatrix} x_1^*(k) & \\ \lambda^{1/2}x_1^*(k-1) & \boldsymbol{X}_N^H(k-1) \\ \vdots & \\ \lambda^{k/2}x_1^*(0) & \boldsymbol{0}^T \end{bmatrix} \\ &= \left[\begin{array}{c|c} \boldsymbol{d}_f^{*(1)}(k) & \boldsymbol{X}_N^H(k-1) \\ & \boldsymbol{0}^T \end{array}\right], \end{aligned} \tag{5.48}$$

onde $\boldsymbol{d}_{f1}^{*(1)}(k) = \begin{bmatrix} x_1(k) & \lambda^{1/2}x_1(k-1) & \cdots & \lambda^{k/2}x_1(0) \end{bmatrix}^H$.

Sendo $\boldsymbol{U}_N(k+1)$ o fator de Cholesky de $\boldsymbol{X}_N(k-1)\boldsymbol{X}_N^H(k-1)$ e $\boldsymbol{Q}_N^{(1)}(k)$ a matriz ortogonal associada a esse processo, pode-se escrever, da EQ. 5.48, que

$$\begin{aligned}&\begin{bmatrix} \boldsymbol{Q}_N^{(1)}(k) & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_{1\times 1} \end{bmatrix} \left[\begin{array}{c|c} \boldsymbol{d}_f^{*(1)}(k) & \boldsymbol{X}_N^H(k-1) \\ & \boldsymbol{0}^T \end{array}\right] \\ &\qquad = \begin{bmatrix} \boldsymbol{e}_{fq1}^{(1)}(k) & \boldsymbol{0} \\ \boldsymbol{d}_{fq2}^{(1)}(k) & \boldsymbol{U}_N(k-1) \\ \lambda^{k/2}x_1^*(0) & \boldsymbol{0}^T \end{bmatrix}. \end{aligned} \tag{5.49}$$

Para completar o processo de triangularização de $\boldsymbol{X}_{N+1}^H(k)$, gerando $\boldsymbol{U}_{N+1}(k)$, deve-se

aplicar duas matrizes de rotações de Givens à EQ. 5.49 como segue

$$\begin{aligned}\begin{bmatrix} \mathbf{0} \\ \boldsymbol{U}_{N+1}(k) \end{bmatrix} &= \boldsymbol{Q}'_f{}^{(1)}(k)\boldsymbol{Q}_f{}^{(1)}(k) \begin{bmatrix} \boldsymbol{e}^{(1)}_{fq1}(k) & \mathbf{0} \\ \boldsymbol{d}^{(1)}_{fq2}(k) & \boldsymbol{U}_N(k-1) \\ \lambda^{k/2}x_1(0) & \mathbf{0}^T \end{bmatrix} \\ &= \boldsymbol{Q}'_f{}^{(1)}(k) \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \boldsymbol{d}^{(1)}_{fq2}(k) & \boldsymbol{U}_N(k-1) \\ e^{(1)}_{fN}(k) & \mathbf{0}^T \end{bmatrix}, \end{aligned} \tag{5.50}$$

sendo $\boldsymbol{Q}_f{}^{(1)}(k)$ a matriz ortogonal responsável por zerar as primeiras $k-N$ linhas de $\boldsymbol{X}^H_{N+1}(k)$ e $\boldsymbol{Q}'_f{}^{(1)}(k)$ completando a triangularização ao zerar $\boldsymbol{d}^{(1)}_{fq2}(k)$, do primeiro ao último elemento, sobre $e^{(1)}_{fN}(k)$. Após remover a seção nula crescente na parte superior da EQ. 5.50 chega-se a

$$\boldsymbol{U}_{N+1}(k) = \boldsymbol{Q}'_{\theta f}{}^{(1)}(k) \begin{bmatrix} \boldsymbol{d}^{(1)}_{fq2}(k) & \boldsymbol{U}_N(k-1) \\ e^{(1)}_{fN}(k) & \mathbf{0}^T \end{bmatrix}. \tag{5.51}$$

Da EQ. 5.51, pode-se obter a relação seguinte que será útil nas próximas etapas para a obtenção de $\boldsymbol{a}_N(k+1)$ e $\boldsymbol{f}_N(k+1)$.

$$[\boldsymbol{U}_{N+1}(k)]^{-1} = \begin{bmatrix} \mathbf{0}^T & \frac{1}{e^{(1)}_{fN}(k)} \\ \boldsymbol{U}_N^{-1}(k-1) & -\frac{1}{e^{(1)}_{fN}(k)}\boldsymbol{U}_N^{-1}(k-1)\boldsymbol{d}^{(1)}_{fq2}(k) \end{bmatrix} \left[\boldsymbol{Q}'_{\theta f}{}^{(1)}(k)\right]^H \tag{5.52}$$

logo,

$$[\boldsymbol{U}_{N+1}(k+1)]^{-1} = \begin{bmatrix} \mathbf{0}^T & \frac{1}{e^{(1)}_{fN}(k+1)} \\ \boldsymbol{U}_N^{-1}(k) & -\frac{1}{e^{(1)}_{fN}(k+1)}\boldsymbol{U}_N^{-1}(k)\boldsymbol{d}^{(1)}_{fq2}(k+1) \end{bmatrix} \left[\boldsymbol{Q}'_{\theta f}{}^{(1)}(k+1)\right]^H \tag{5.53}$$

À semelhança do que foi feito na seção anterior, pode-se, a partir da EQ. 5.50, escrever que

$$\begin{bmatrix} \mathbf{0} \\ e^{(1)}_{f0}(k) \end{bmatrix} = \boldsymbol{Q}'_{\theta f}{}^{(1)}(k) \begin{bmatrix} \boldsymbol{d}^{(1)}_{fq2}(k) \\ e^{(1)}_{fN}(k) \end{bmatrix}. \tag{5.54}$$

Da EQ. 5.54 percebe-se que $\boldsymbol{Q}'_{\theta f}{}^{(1)}(k)$ é uma seqüência de rotações de Givens que anula os elementos de $\boldsymbol{d}^{(1)}_{fq2}(k)$ em relação a $e^{(1)}_{fN}(k)$.

Agora, recordando a EQ. 5.45, pode-se usar a EQ. 5.53 e o vetor de entrada definido na EQ. 5.40 para obter

$$
\begin{aligned}
\boldsymbol{f}_{N+1}(k+1) &= \lambda^{-1/2}\boldsymbol{Q}'_{\theta f}{}^{(1)}(k+1)\begin{bmatrix} \mathbf{0} & \boldsymbol{U}_N^{-H}(k) \\ \frac{1}{e_{fN}^{(1)}(k+1)} & -\frac{1}{e_{fN}^{(1)}(k+1)}\boldsymbol{d}_{fq2}^{H}{}^{(1)}(k+1)\boldsymbol{U}_N^{-H}(k) \end{bmatrix}\begin{bmatrix} x_1(k+1) \\ \boldsymbol{x}_N(k) \end{bmatrix} \\
&= \lambda^{-1/2}\boldsymbol{Q}'_{\theta f}{}^{(1)}(k+1)\begin{bmatrix} \boldsymbol{U}_N^{-H}(k)\boldsymbol{x}_N(k) \\ \frac{x_1(k+1)}{e_{fN}^{(1)}(k+1)} - \frac{1}{e_{fN}^{(1)}(k+1)}\left[\boldsymbol{U}_N^{-1}(k)\boldsymbol{d}_{fq2}^{(1)}(k+1)\right]^H\boldsymbol{x}_N(k) \end{bmatrix} \\
&= \boldsymbol{Q}'_{\theta f}{}^{(1)}(k+1)\begin{bmatrix} \boldsymbol{f}_N(k) \\ p^{(1)}(k+1) \end{bmatrix}, \qquad (5.55)
\end{aligned}
$$

onde

$$
\begin{aligned}
p^{(1)}(k+1) &= \lambda^{-1/2}\left[\frac{x_1(k+1)}{e_{fN}^{(1)}(k+1)} - \frac{1}{e_{fN}^{(1)}(k+1)}\left[\boldsymbol{U}_N^{-1}(k)\boldsymbol{d}_{fq2}^{(1)}(k+1)\right]^H\boldsymbol{x}_N(k)\right] \\
&= \frac{\lambda^{-1/2}}{e_{fN}^{(1)}(k+1)}\left[x_1(k+1) - \left[\boldsymbol{U}_N^{-1}(k)\boldsymbol{d}_{fq2}^{(1)}(k+1)\right]^H\boldsymbol{x}_N(k)\right] \\
&= \frac{\lambda^{-1/2}}{e_{fN}^{(1)}(k+1)}\left[x_1(k+1) - \boldsymbol{w}_{f_N}^H(k+1)\boldsymbol{x}_N(k)\right] \\
&= \frac{e_N{}^{(1)}(k+1)}{\lambda^{1/2}e_{fN}^{(1)}(k+1)} \qquad (5.56)
\end{aligned}
$$

sendo $e_N{}^{(1)}(k+1)$ o erro *a posteriori* da predição progressiva do primeiro canal. A atualização de $\boldsymbol{d}_{fq2}^{(1)}(k)$ é realizada de acordo com

$$
\begin{bmatrix} \tilde{e}_{fq1}^{(1)}(k+1) \\ \boldsymbol{d}_{fq2}^{(1)}(k+1) \end{bmatrix} = \boldsymbol{Q}_{\theta_N}^{(0)}(k)\begin{bmatrix} x_1(k+1) \\ \lambda^{1/2}\boldsymbol{d}_{fq2}^{(1)}(k) \end{bmatrix} \qquad (5.57)
$$

e a matriz $\boldsymbol{Q}_{\theta_{N+1}}(k+1)$, necessária nas próximas etapas, é obtida de

$$
\boldsymbol{Q}_{\theta_{N+1}}^{(1)}(k+1)\begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \gamma_{N+1}^{(1)}(k+1) \\ \boldsymbol{f}_{N+1}(k+1) \end{bmatrix}. \qquad (5.58)
$$

Etapa II:

A matriz de informação de entrada $\boldsymbol{X}_{N+i}^H(k)$ está relacionada com $\boldsymbol{X}_{N+i-1}^H(k)$ da seguinte maneira

$$
\boldsymbol{X}_{N+i}^H(k) = \begin{bmatrix} x_i^*(k) & \\ \lambda^{1/2}x_i^*(k-1) & \boldsymbol{X}_{N+i-1}^H(k) \\ \vdots & \\ \lambda^{k/2}x_i^*(0) & \end{bmatrix}\boldsymbol{P}_i. \qquad (5.59)
$$

Como nos algoritmos abordados anteriormente, aqui também é necessário triangularizar $\boldsymbol{X}^H_{N+i}(k)$ obtendo $\boldsymbol{U}_{N+i}(k)$, que corresponde ao fator de Cholesky de $\boldsymbol{X}^H_{N+i}(k)$. Este processo é demonstrado nos próximos passos. Seja $\boldsymbol{Q}_{\theta_{N+1-i}}(k)$ a matriz ortogonal que permite obter a decomposição QR de $\boldsymbol{X}^H_{N+i-1}(k)$, com $\boldsymbol{U}_{N+i-1}(k)$ sendo o correspondente fator de Cholesky; pode ser escrito, a partir da EQ. 5.59, que

$$\begin{aligned}
\boldsymbol{Q}_f{}^{(i)}(k) \begin{bmatrix} \boldsymbol{Q}_{\theta_{N+1-i}}(k) & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{X}^H_{N+1-i}(k) \\ \mathbf{0}^T \end{bmatrix} &= \boldsymbol{Q}_f{}^{(i)}(k) \\
&\quad \times \begin{bmatrix} \boldsymbol{e}^{(i)}_{fq1_{N+1-i}}(k) & \mathbf{0} \\ \boldsymbol{d}^{(i)}_{fq2}(k-1) & \boldsymbol{U}_{N+i-1}(k) \\ \mathbf{0} & \mathbf{0}^T \end{bmatrix} \boldsymbol{P}_i \\
&= \begin{bmatrix} \boldsymbol{d}^{(i)}_{fq2}(k-1) & \boldsymbol{U}_{N+i-1}(k) \\ e^{(i)}_{f_{N+i-1}}(k) & \mathbf{0}^T \end{bmatrix} \boldsymbol{P}_i.
\end{aligned} \tag{5.60}$$

A EQ. 5.60 resulta da anulação de $\boldsymbol{e}^{(i)}_{fq1_{N+1-i}}(k)$ sobre o primeiro elemento da última linha da matriz, usando-se um fator ortogonal $\boldsymbol{Q}_f{}^{(i)}(k)$ e removendo-se a seção nula resultante.

Como visto anteriormente, $\boldsymbol{P}_i$ é uma matriz de permutação e a sua existência na equação acima impede que a anulação de $\boldsymbol{d}^{(i)}_{fq2}(k)$, para completar a triangularização, seja feita da mesma maneira que nos algoritmos da seção anterior, isto é, zerá-lo diretamente sobre $e^{(i)}_{f_{N+i-1}}(k)$. A aplicação das rotações de Givens que completam a triangularização nesta circunstância está ilustrada na FIG. 5.3. Este processo pode ser descrito como segue. A matriz de permutação, $\boldsymbol{P}_i$, desloca $\boldsymbol{d}^{(i)}_{fq2}(k)$ para a i–ésima posição, como mostrado na parte I da figura. Após isso, são aplicadas $N+i-p_i$ rotações de Givens $\boldsymbol{Q}'_{\theta f}{}^{(i)}$ que anulam os primeiros $N+i-p_i$ elementos de $\boldsymbol{d}^{(i)}_{fq2}(k)$ sobre $e^{(i)}_{f_{N+i-1}}(k)$, de cima a baixo. Com isso, são criados $(N+i)-p_i+1$ elementos não nulos no fim da última linha da matriz resultante, como ilustrado na parte II da figura. Para que a matriz resultante possa ter uma estrutura triangular inferior, precisa-se de outro fator de permutação que desloque a última linha para a posição $N-p_i+1$, após deslocar para baixo as penúltimas $N-p_i$ linhas. Pode-se concluir, pela simples observação, que esta última matriz de permutação corresponde a $\boldsymbol{P}_i^T$. Para garantir que a matriz triangular obtida no processo anterior seja positiva definida, é necessário que $e^{(i)}_{f_{N+i-1}}(k)$ e o restante dos elementos da diagonal de $\boldsymbol{U}_{N+i-1}(k)$ sejam positivos. Com uma inicialização apropriada, os elementos da diagonal

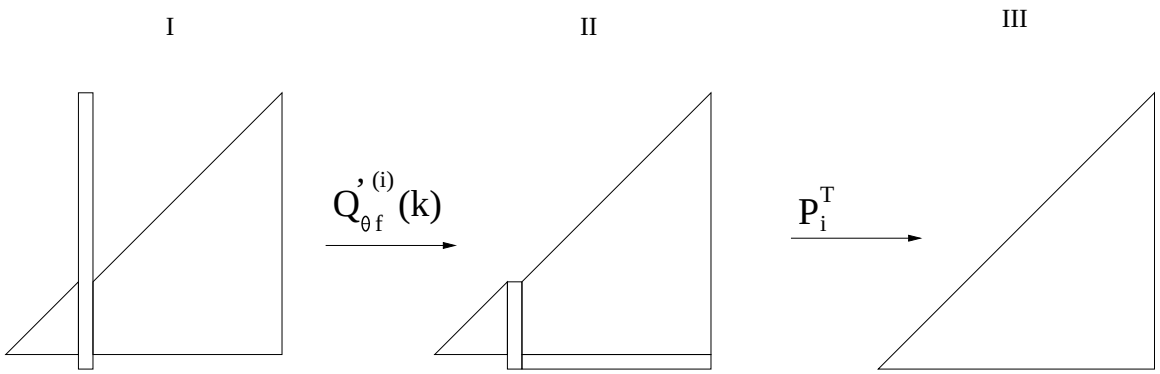

FIG. 5.3: Finalizando a triangularização

principal são atualizados com valores positivos que correspondem aos módulos dos erros dados por $e^{(i)}_{f_{N+i-1}}(k)$. Sendo assim, pode-se finalmente concluir que

$$\boldsymbol{U}_{N+i}(k) = \boldsymbol{P}_i^T \boldsymbol{Q}'^{\,(i)}_{\theta f}(k) \begin{bmatrix} \boldsymbol{d}^{(i)}_{fq2}(k) & \boldsymbol{U}_{N+i-1}(k) \\ \boldsymbol{e}^{(i)}_{f_{N+i-1}}(k) & \boldsymbol{0}^T \end{bmatrix} \boldsymbol{P}_i. \tag{5.61}$$

Da EQ. 5.61, obtém-se as duas relações de interesse seguintes:

$$[\boldsymbol{U}_{N+i}(k)]^{-1} = \boldsymbol{P}_i^T \begin{bmatrix} \boldsymbol{0}^T & \frac{1}{e^{(i)}_{f_{N+i-1}}(k)} \\ \boldsymbol{U}^{-1}_{N+i-1}(k) & -\frac{1}{e^{(i)}_{f_{N+i-1}}(k)} \boldsymbol{U}^{-1}_{N+i-1}(k) \boldsymbol{d}^{(i)}_{fq2}(k) \end{bmatrix} \boldsymbol{Q}'^{H\,(i)}_{\theta f}(k) \boldsymbol{P}_i \tag{5.62}$$

e, conseqüentemente,

$$[\boldsymbol{U}_{N+i}(k+1)]^{-1} = \boldsymbol{P}_i^T \begin{bmatrix} \boldsymbol{0}^T & \frac{1}{e^{(i)}_{f_{N+i-1}}(k+1)} \\ \boldsymbol{U}^{-1}_{N+i-1}(k+1) & -\frac{\boldsymbol{U}^{-1}_{N+i-1}(k+1) \boldsymbol{d}^{(i)}_{fq2}(k+1)}{e^{(i)}_{f_{N+i-1}}(k+1)} \end{bmatrix} \boldsymbol{Q}'^{H\,(i)}_{\theta f}(k+1) \boldsymbol{P}_i \tag{5.63}$$

As relações obtidas acima, levam em consideração o fato de que a matriz $\boldsymbol{P}_i$ é ortogonal para manter a norma de $\boldsymbol{U}_{N+i}$ inalterada.

Assim sendo, a EQ. 5.63 pode ser usada, juntamente com a EQ. 5.45 e a EQ. 5.41, para determinar $\boldsymbol{f}_{N+i}(k+1)$ como segue.

$$
\begin{aligned}
\boldsymbol{f}_{N+i}(k+1) &= \boldsymbol{U}_{N+i}^{-H}(k+1)\boldsymbol{x}_{N+i}(k+1) \\
&= \boldsymbol{P}_i^T \boldsymbol{Q'}_{\theta f}{}^{(i)}(k+1) \begin{bmatrix} \mathbf{0} & \boldsymbol{U}_{N+i-1}^{-H}(k+1) \\ \frac{1}{e_{f_{N+i-1}}^{(i)}(k+1)} & -\frac{\left[\boldsymbol{U}_{N+i-1}^{-1}(k+1)\boldsymbol{d}_{fq2}^{(i)}(k+1)\right]^H}{e_{f_{N+i-1}}^{(i)}(k+1)} \end{bmatrix} \boldsymbol{P}_i \\
&\quad \times \boldsymbol{P}_i^T \begin{bmatrix} x_i(k+1) \\ \boldsymbol{x}_{N+1-i}(k+1) \end{bmatrix} \\
&= \boldsymbol{P}_i^T \boldsymbol{Q'}_{\theta f}{}^{(i)}(k+1) \begin{bmatrix} \boldsymbol{U}_{N+i-1}^{-H}(k+1)\boldsymbol{x}_{N+1-i}(k+1) \\ \frac{x_i(k+1)}{e_{f_{N+i-1}}^{(i)}(k+1)} - \frac{\left[\boldsymbol{U}_{N+i-1}^{-1}(k+1)\boldsymbol{d}_{fq2}^{(i)}(k+1)\right]^H \boldsymbol{x}_{N+1-i}(k+1)}{e_{f_{N+i-1}}^{(i)}(k+1)} \end{bmatrix} \\
&= \boldsymbol{P}_i^T \boldsymbol{Q'}_{\theta f}{}^{(i)}(k+1) \begin{bmatrix} \boldsymbol{f}_{N+i-1}(k+1) \\ p_{N+i-1}^{(i)}(k+1) \end{bmatrix},
\end{aligned}
\tag{5.64}
$$

onde

$$
\begin{aligned}
p_{N+i-1}^{(i)}(k+1) &= \frac{x_i(k+1)}{e_{f_{N+i-1}}^{(i)}(k+1)} - \frac{\left[\boldsymbol{U}_{N+i-1}^{-1}(k+1)\boldsymbol{d}_{fq2}^{(i)}(k+1)\right]^H \boldsymbol{x}_{N+1-i}(k+1)}{e_{f_{N+i-1}}^{(i)}(k+1)} \\
&= \frac{1}{e_{f_{N+i-1}}^{(i)}(k+1)} \left[x_i(k+1) - \boldsymbol{w}_{f_{N+i-1}}^H(k+1)\boldsymbol{x}_{N+1-i}(k+1)\right] \\
&= \frac{e_{N+i-1}^{(i)}(k+1)}{e_{f_{N+i-1}}^{(i)}(k+1)}.
\end{aligned}
\tag{5.65}
$$

O escalar $e_{N+i-1}^{(i)}(k+1)$ é o erro *a posteriori* de predição progressiva do *i–ésimo* canal. A partir da EQ. 5.64, e considerando as definições de $\boldsymbol{P}_i$ e de $\boldsymbol{Q'}_{\theta f}{}^{(i)}(k+1)$, pode-se concluir que os últimos $p_i - 1$ elementos de $\boldsymbol{f}_{N+i}(k+1)$ e de $\boldsymbol{f}_{N+i-1}(k+1)$ são idênticos. Como será visto mais adiante, esta constatação é de suma importância, pois permite uma considerável redução na complexidade computacional do algoritmo.

A atualização de $\boldsymbol{d}_{fq2}^{(i)}(k)$ é realizada de acordo com

$$
\begin{bmatrix} \tilde{e}_{fq1}^{(i)}(k+1) \\ \boldsymbol{d}_{fq2}^{(i)}(k+1) \end{bmatrix} = \boldsymbol{Q}_{\theta_N}^{(i-1)}(k) \begin{bmatrix} x_i^*(k+1) \\ \lambda^{1/2}\boldsymbol{d}_{fq2}^{(i)}(k) \end{bmatrix}.
\tag{5.66}
$$

A matriz de rotações, $\boldsymbol{Q}_{\theta_{N+i}}(k+1)$, necessária para a próxima etapa, é obtida de

$$\boldsymbol{Q}_{\theta_{N+i}}^{(i)}(k+1)\begin{bmatrix}1\\ \mathbf{0}\end{bmatrix}=\begin{bmatrix}\gamma_{N+i}^{(i)}(k+1)\\ \boldsymbol{f}_{N+i}(k+1)\end{bmatrix}. \tag{5.67}$$

O algoritmo completo está descrito na TAB. 5.5 (RAMOS, 2004b).

5.6.3 O ALGORITMO BASEADO NA ATUALIZAÇÃO DO VETOR DE ERROS *A PRIORI* REGRESSIVO

Para evitar repetições desnecessárias, está seção fará uma apresentação bastante sumária na qual serão utilizados alguns resultados obtidos anteriormente. As duas etapas, similares às descritas na sub-seção anterior, serão descritas de forma breve.

Doravante, o vetor de interesse a ser atualizado, e que caracteriza o algoritmo, é $\boldsymbol{a}_{N+i}(k+1)$, introduzido anteriormente.

Etapa I:

Partindo da EQ. 5.44, usando a EQ. 5.52 e o vetor de entrada definido na EQ. 5.40, chega-se a

$$\begin{aligned}
\boldsymbol{a}_{N+1}(k+1) &= \lambda^{-1/2}\boldsymbol{Q}_{\theta f}'^{(1)}(k)\begin{bmatrix}\mathbf{0} & \boldsymbol{U}_N^{-H}(k-1)\\ \frac{1}{e_{fN}^{(1)}(k)} & -\frac{1}{e_{fN}^{(1)}(k)}\boldsymbol{d}_{fq2}^{H\,(1)}(k)\boldsymbol{U}_N^{-H}(k-1)\end{bmatrix}\begin{bmatrix}x_1(k+1)\\ \boldsymbol{x}_N(k)\end{bmatrix}\\
&= \lambda^{-1/2}\boldsymbol{Q}_{\theta f}'^{(1)}(k)\begin{bmatrix}\boldsymbol{U}_N^{-H}(k-1)\boldsymbol{x}_N(k)\\ \frac{x_1(k+1)}{e_{fN}^{(1)}(k)}-\frac{1}{e_{fN}^{(1)}(k)}\left[\boldsymbol{U}_N^{-1}(k-1)\boldsymbol{d}_{fq2}^{(1)}(k)\right]^H\boldsymbol{x}_N(k)\end{bmatrix}\\
&= \boldsymbol{Q}_{\theta f}'^{(1)}(k)\begin{bmatrix}\boldsymbol{a}_N(k)\\ r^{(1)}(k+1)\end{bmatrix},
\end{aligned} \tag{5.68}$$

sendo

$$\begin{aligned}
r^{(1)}(k+1) &= \lambda^{-1/2}\left[\frac{x_1(k+1)}{e_{fN}^{(1)}(k)}-\frac{1}{e_{fN}^{(1)}(k)}\left[\boldsymbol{U}_N^{-1}(k-1)\boldsymbol{d}_{fq2}^{(1)}(k)\right]^H\boldsymbol{x}_N(k)\right]\\
&= \frac{\lambda^{-1/2}}{e_{fN}^{(1)}(k)}\left[x_1(k+1)-\left[\boldsymbol{U}_N^{-1}(k-1)\boldsymbol{d}_{fq2}^{(1)}(k)\right]^H\boldsymbol{x}_N(k)\right]\\
&= \frac{\lambda^{-1/2}}{e_{fN}^{(1)}(k)}\left[x_1(k+1)-\boldsymbol{w}_{f_N}^H(k)\boldsymbol{x}_N(k)\right]\\
&= \frac{e_N'^{(1)}(k+1)}{\lambda^{1/2}e_{fN}^{(1)}(k)}.
\end{aligned} \tag{5.69}$$

O escalar ${e'_N}^{(1)}(k+1)$ é o erro *a priori* da predição progressiva para o primeiro canal. A matrix $\boldsymbol{Q}_{\theta_{N+1}}(k+1)$ é obtida de

$$\begin{bmatrix} 1/\gamma^{(1)}_{N+1} \\ \mathbf{0} \end{bmatrix} = \boldsymbol{Q}^{(1)}_{\theta_{N+1}}(k+1) \begin{bmatrix} 1 \\ -\boldsymbol{a}_{N+1}(k+1) \end{bmatrix}. \tag{5.70}$$

Etapa II:

O vetor $\boldsymbol{a}_{N+i}(k+1)$ é obtido da EQ. 5.63, juntamente com a EQ. 5.45 e a EQ. 5.41, como segue

$$\begin{aligned}
\boldsymbol{a}_{N+i}(k+1) &= \boldsymbol{U}^{-H}_{N+i}(k)\boldsymbol{x}_{N+i}(k+1) \\
&= \boldsymbol{P}_i^T {\boldsymbol{Q}'_{\theta f}}^{(i)}(k) \begin{bmatrix} \mathbf{0} & \boldsymbol{U}^{-H}_{N+i-1}(k) \\ \frac{1}{e^{(i)}_{f_{N+i-1}}(k)} & -\frac{\left[\boldsymbol{U}^{-1}_{N+i-1}(k)\boldsymbol{d}^{(i)}_{fq2}(k)\right]^H}{e^{(i)}_{f_{N+i-1}}(k)} \end{bmatrix} \boldsymbol{P}_i \\
&\times \boldsymbol{P}_i^T \begin{bmatrix} x_i(k+1) \\ \boldsymbol{x}_{N+1-i}(k+1) \end{bmatrix} \\
&= \boldsymbol{P}_i^T {\boldsymbol{Q}'_{\theta f}}^{(i)}(k) \begin{bmatrix} \boldsymbol{U}^{-H}_{N+i-1}(k)\boldsymbol{x}_{N+1-i}(k+1) \\ \frac{x_i(k+1)}{e^{(i)}_{f_{N+i-1}}(k)} - \frac{\left[\boldsymbol{U}^{-1}_{N+i-1}(k)\boldsymbol{d}^{(i)}_{fq2}(k)\right]^H \boldsymbol{x}_{N+1-i}(k+1)}{e^{(i)}_{f_{N+i-1}}(k)} \end{bmatrix} \\
&= \boldsymbol{P}_i^T {\boldsymbol{Q}'_{\theta f}}^{(i)}(k+1) \begin{bmatrix} \boldsymbol{a}_{N+i-1}(k+1) \\ r^{(i)}_{N+i-1}(k+1) \end{bmatrix},
\end{aligned} \tag{5.71}$$

onde

$$\begin{aligned}
r^{(i)}_{N+i-1}(k+1) &= \frac{x_i(k+1)}{e^{(i)}_{f_{N+i-1}}(k)} - \frac{\left[\boldsymbol{U}^{-1}_{N+i-1}(k)\boldsymbol{d}^{(i)}_{fq2}(k)\right]^H \boldsymbol{x}_{N+1-i}(k)}{e^{(i)}_{f_{N+i-1}}(k)} \\
&= \frac{1}{e^{(i)}_{f_{N+i-1}}(k)} \left[x_i(k+1) - \boldsymbol{w}^H_{f_{N+i-1}}(k)\boldsymbol{x}_{N+1-i}(k+1)\right] \\
&= \frac{{e'_{N+i-1}}^{(i)}(k)}{e^{(i)}_{f_{N+i-1}}(k)}.
\end{aligned} \tag{5.72}$$

O escalar $e^{(i)}_{N+i-1}(k)$ é o erro *a priori* de predição progressiva para o *i–ésimo* canal.

De modo semelhante ao que foi apresentado na Seção 5.6.2, pode-se concluir, a partir da EQ. 5.71, que os últimos $p_i - 1$ elementos de $\boldsymbol{a}_{N+i}(k+1)$ e $\boldsymbol{a}_{N+i-1}(k+1)$ são idênticos,

levando, tal como no caso *a posteriori*, a uma redução na complexidade computacional do algoritmo.

De maneira similar, $\boldsymbol{d}_{fq2}^{(i)}(k)$ é atualizado de acordo com

$$\begin{bmatrix} \tilde{e}_{fq1}^{(i)}(k+1) \\ \boldsymbol{d}_{fq2}^{(i)}(k+1) \end{bmatrix} = \boldsymbol{Q}_{\theta_N}^{(i-1)}(k) \begin{bmatrix} x_i(k+1) \\ \lambda^{1/2}\boldsymbol{d}_{fq2}^{(i)}(k) \end{bmatrix} \tag{5.73}$$

e matriz $\boldsymbol{Q}_{\theta_{N+i}}(k+1)$ é obtida de (RONTOGIANNIS, 1998)

$$\begin{bmatrix} 1/\gamma_{N+i}^{(i)} \\ \mathbf{0} \end{bmatrix} = \boldsymbol{Q}_{\theta_{N+i}}^{(i)}(k+1) \begin{bmatrix} 1 \\ -\boldsymbol{a}_{N+i}(k+1) \end{bmatrix}. \tag{5.74}$$

5.7 ESTRUTURA EM TRELIÇA DO ALGORITMO BASEADO NA ATUALIZAÇÃO DO VETOR DE ERROS *A POSTERIORI* REGRESSIVO

Pode-se constatar que o algoritmo da TAB. 5.5 possui uma execução seqüencial composta de M etapas cada qual correspondente a um dos M canais de entrada. Nesta execução seqüencial, a $(i+1)$-*ésima* etapa é executada após a conclusão da $i-$*ésima* etapa. Nesta etapa são obtidas as quantidades $\theta^{(i)}$, $\gamma^{(i)}$ e $\boldsymbol{f}^{(i)}$ que são usadas na etapa seguinte. Esta execução seqüencial, canal após canal, pode ser evitada adotando-se uma forma alternativa para a execução das etapas 2 e 3 do algoritmo da TAB. 5.5. Isso pode ser conseguido usando as seguintes relações para o cálculo de $p_j^{(i)}(k+1)$ e de $|e_{f_j}^{(i)}(k+1)|$:

$$\begin{aligned} |e_{f_j}^{(i)}(k+1)| &= \sqrt{\left(\lambda^{1/2}|e_{f_j}^{(i)}(k)|\right)^2 + \left|e_{fq1_j}^{(i)}(k+1)\right|^2} \\ & \qquad i = 1, 2, \cdots, M \qquad j = p_i, \cdots, L \end{aligned} \tag{5.75}$$

e

$$\begin{aligned} p_j^{(i)}(k+1) &= \frac{\gamma_j^{(i-1)}(k) e_{fq1_j}^{(i)}(k+1)}{|e_{f_j}^{(i)}(k+1)|} \qquad i = 1, 2, \cdots, M \\ & \qquad j = p_i, \cdots, L \end{aligned} \tag{5.76}$$

Com isso consegue-se uma execução ascendente das etapas 2 e 3, semelhante às restantes etapas, o que possibilita a escrita do algoritmo na forma recursiva (estrutura em treliça) apresentada na TAB. 5.6 (RAMOS, 2004d).

Neste capítulo, além de um melhoramento feito na versão em blocos (versão na estrutura transversal), introduziu-se uma versão em treliça do algoritmo QRD-RLS multicanal rápido baseado na atualização de erros *a posteriori* regressivos. A recursividade na ordem deste novo algoritmo é sua principal característica que favorece sua implemetação prática. Foi também introduzida uma versão do algoritmo QRD-RLS multicanal rápido baseado na atualização de erros *a posteriori* regressivos que atende ao caso geral de canais com ordens iguais ou não, o que é particularmente útil em muitos casos onde se lida com filtros multicanais de ordens diferentes, como, por exemplo, na filtragem de Volterra. Para este último caso, foi apresentada também uma versão recursiva que pode ser desejável em algumas situações práticas.

No próximo capítulo, será avaliado o desempenho destes algoritmos quanto à complexidade computacional, à velocidade de convergência e à estabilidade. Serão consideradas duas aplicações distintas a serem descritas.

TAB. 5.1: Equações do Algoritmo MCFQRD_PRI_B.

Inicializações:

$\boldsymbol{f}_N(0) = 0$; $\boldsymbol{D}_{fq2}(0) = 0$;

$\boldsymbol{d}_{q2}(0) = 0$; $\boldsymbol{E}_f(0) = \boldsymbol{I}$;

Todos os *cosenos* $= 1$ e todos os *senos* $= 0$;

Para $k = 1, 2, \cdots$

{

1. Obtendo $\boldsymbol{D}_{fq2}(k+1)$ e $\widetilde{\boldsymbol{e}}^H_{fq1}(k+1)$:

$$\begin{bmatrix} \widetilde{\boldsymbol{e}}^H_{fq1}(k+1) \\ \boldsymbol{D}_{fq2}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} \boldsymbol{x}^H_{k+1} \\ \lambda^{1/2}\boldsymbol{D}_{fq2}(k) \end{bmatrix}$$

2. Obtendo $\boldsymbol{E}_f(k+1)$:

$$\begin{bmatrix} \boldsymbol{0}^T \\ \boldsymbol{E}_f(k+1) \end{bmatrix} = \overline{\boldsymbol{Q}}_f(k+1) \begin{bmatrix} \widetilde{\boldsymbol{e}}^T_{fq1}(k+1) \\ \lambda^{1/2}\boldsymbol{E}_f(k) \end{bmatrix}$$

3. Obtendo $\boldsymbol{r}(k+1)$:

$$\begin{bmatrix} \star \\ \boldsymbol{0} \end{bmatrix} = \overline{\boldsymbol{Q}}_f(k+1) \begin{bmatrix} 1/\gamma(k) \\ -\boldsymbol{r}(k+1) \end{bmatrix}$$

4. Obtendo $\boldsymbol{a}_N(k+1)$:

$$\boldsymbol{a}_{N+1}(k+1) = \boldsymbol{Q}'_{\theta f}(k) \begin{bmatrix} \boldsymbol{a}_N(k) \\ \boldsymbol{r}(k+1) \end{bmatrix}$$

5. Obtendo $\boldsymbol{Q}'_{\theta f}(k+1)$:

$$\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{E}^0_f(k+1) \end{bmatrix} = \boldsymbol{Q}'_{\theta f}(k+1) \begin{bmatrix} \boldsymbol{D}_{fq2}(k+1) \\ \boldsymbol{E}_f(k+1) \end{bmatrix}$$

6. Obtendo $\boldsymbol{Q}_\theta(k+1)$ e $\gamma(k+1)$:

$$\begin{bmatrix} 1/\gamma(k+1) \\ \boldsymbol{0} \end{bmatrix} = \boldsymbol{Q}_\theta(k+1) \begin{bmatrix} 1 \\ -\boldsymbol{a}_N(k+1) \end{bmatrix}$$

7. Estimação Conjunta:

$$\begin{bmatrix} e_{q1}(k+1) \\ \boldsymbol{d}_{q2}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta(k+1) \begin{bmatrix} d^*(k+1) \\ \lambda^{1/2}\boldsymbol{d}_{q2}(k) \end{bmatrix}$$

8. Obtendo o erro *a priori*:

$\varepsilon(k+1) = e^*_{q1}(k+1)/\gamma(k+1)$

}

TAB. 5.2: Equações do Algoritmo MCFQRD_POS_B.

Inicializações:

$\boldsymbol{f}_N(0) = 0$; $\boldsymbol{D}_{fq2}(0) = 0$;

$\boldsymbol{d}_{q2}(0) = 0$; $\boldsymbol{E}_f(0) = \boldsymbol{I}$;

Todos os *cosenos* $= 1$ e todos os *senos* $= 0$;

Para $k = 1, 2, \cdots$

{

1. Obtendo $\boldsymbol{D}_{fq2}(k+1)$ e $\widetilde{\boldsymbol{e}}^H_{fq1}(k+1)$:

$$\begin{bmatrix} \widetilde{\boldsymbol{e}}^H_{fq1}(k+1) \\ \boldsymbol{D}_{fq2}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} \boldsymbol{x}^H_{k+1} \\ \lambda^{1/2}\boldsymbol{D}_{fq2}(k) \end{bmatrix}$$

2. Obtendo $\boldsymbol{E}_f(k+1)$:

$$\begin{bmatrix} \mathbf{0}^T \\ \boldsymbol{E}_f(k+1) \end{bmatrix} = \overline{\boldsymbol{Q}}_f(k+1) \begin{bmatrix} \widetilde{\boldsymbol{e}}^H_{fq1}(k+1) \\ \lambda^{1/2}\boldsymbol{E}_f(k) \end{bmatrix}$$

3. Obtendo $\boldsymbol{p}(k+1)$:

$$\begin{bmatrix} \star \\ \boldsymbol{p}(k+1) \end{bmatrix} = \overline{\boldsymbol{Q}}_f(k+1) \begin{bmatrix} \gamma(k) \\ \mathbf{0} \end{bmatrix}$$

4. Obtendo $\boldsymbol{Q}'_{\theta f}(k+1)$:

$$\begin{bmatrix} \mathbf{0} \\ \boldsymbol{E}^0_f(k+1) \end{bmatrix} = \boldsymbol{Q}'_{\theta f}(k+1) \begin{bmatrix} \boldsymbol{D}_{fq2}(k+1) \\ \boldsymbol{E}_f(k+1) \end{bmatrix}$$

5. Obtendo $\boldsymbol{f}_N(k+1)$:

$$\boldsymbol{f}_{N+1}(k+1) = \boldsymbol{Q}'_{\theta f}(k+1) \begin{bmatrix} \boldsymbol{f}_N(k) \\ \boldsymbol{p}(k+1) \end{bmatrix}$$

6. Obtendo $\boldsymbol{Q}_\theta(k+1)$ e $\gamma(k+1)$:

$$\boldsymbol{Q}_\theta(k+1) \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \gamma(k+1) \\ \boldsymbol{f}_N(k+1) \end{bmatrix}$$

7. Estimação Conjunta:

$$\begin{bmatrix} \boldsymbol{e}_{q1}(k+1) \\ \boldsymbol{d}_{q2}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta(k+1) \begin{bmatrix} d^*(k+1) \\ \lambda^{1/2}\boldsymbol{d}_{q2}(k) \end{bmatrix}$$

8. Obtendo o erro *a priori*:

$\varepsilon(k+1) = e^*_{q1}(k+1)/\gamma(k+1)$

}

TAB. 5.3: As Equações do MCFQRD_POS_B Versão em Treliça.

Inicializações:

$\boldsymbol{f}_N(0) = 0$; $\boldsymbol{D}_{fq2}(0) = 0$; $\gamma_0(0) = 1$;

$\boldsymbol{d}_{q2}(0) = 0$; $\boldsymbol{E}_f^i(0) = \delta \boldsymbol{I}$; $\quad \delta =$*número pequeno*;

Todos os *cosenes* $= 1$ e todos os *senos* $= 0$;

Para $k = 1, 2, \cdots$

{

$\widetilde{\boldsymbol{e}}_{fq1}^{(0)\,H}(k+1) = \boldsymbol{x}_{k+1}^H$

A. Obtendo $\boldsymbol{E}_f^{(0)}(k+1)$ e $\overline{\boldsymbol{Q}}_f^{(0)}(k+1)$:

$$\begin{bmatrix} \boldsymbol{0}^T \\ \boldsymbol{E}_f^{(0)}(k+1) \end{bmatrix} = \overline{\boldsymbol{Q}}_f^{(0)}(k+1) \begin{bmatrix} \widetilde{\boldsymbol{e}}_{fq1}^{(0)\,H}(k+1) \\ \lambda^{1/2}\boldsymbol{E}_f^{(0)}(k) \end{bmatrix};$$

B. Obtendo $\boldsymbol{p}_0(k+1)$:

$$\begin{bmatrix} \star \\ \boldsymbol{p}_0(k+1) \end{bmatrix} = \overline{\boldsymbol{Q}}_f^{(0)}(k+1) \begin{bmatrix} \gamma_0(k) \\ \boldsymbol{0} \end{bmatrix};$$

$\boldsymbol{f}^{(N+1)}(k+1) = \boldsymbol{p}_0(k+1)$; $\gamma_0(k+1) = 1$;

$\boldsymbol{e}_{q1}(k+1) = d(k+1)$;

for $i = 1 : N$

{

1. Obtendo $\boldsymbol{D}_{fq2}^{(N-i+1)}(k+1)$ e $\boldsymbol{e}_{fq1}^{(i)}(k+1)$:

$$\begin{bmatrix} \widetilde{\boldsymbol{e}}_{fq1}^{(i)\,H}(k+1) \\ \boldsymbol{D}_{fq2}^{(N-i+1)}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta^{(i)}(k) \begin{bmatrix} \widetilde{\boldsymbol{e}}_{fq1}^{(i-1)\,H}(k+1) \\ \lambda^{1/2}\boldsymbol{D}_{fq2}^{(N-i+1)}(k) \end{bmatrix};$$

2. Obtendo $\boldsymbol{E}_f^{(i)}(k+1)$:

$$\begin{bmatrix} \boldsymbol{0}^T \\ \boldsymbol{E}_f^{(i)}(k+1) \end{bmatrix} = \overline{\boldsymbol{Q}}_f^{(i)}(k+1) \begin{bmatrix} \widetilde{\boldsymbol{e}}_{fq1}^{(i)\,H}(k+1) \\ \lambda^{1/2}\boldsymbol{E}_f^{(i)}(k) \end{bmatrix};$$

3. Obtendo $\boldsymbol{p}_i(k+1)$:

$$\begin{bmatrix} \star \\ \boldsymbol{p}_i(k+1) \end{bmatrix} = \overline{\boldsymbol{Q}}_f^{(i)}(k+1) \begin{bmatrix} \gamma_i(k) \\ \boldsymbol{0} \end{bmatrix};$$

4. Obtendo $\boldsymbol{Q}'_{\theta f}{}^{(N-i+1)}(k+1)$:

$$\begin{bmatrix} \boldsymbol{0}_{M(N-i-1)\times M} \\ \boldsymbol{0}_{M(i-1)\times M} \\ \boldsymbol{E}_f^{(i-1)}(k+1) \end{bmatrix} = \boldsymbol{Q}'_{\theta f}{}^{(N-i+1)}(k+1) \begin{bmatrix} \boldsymbol{0}_{M(N-i)\times M} \\ \boldsymbol{D}_{fq2}^{(N-i+1)}(k) \\ \boldsymbol{0}_{M(i-1)\times M} \\ \boldsymbol{E}_f^{(i)}(k+1) \end{bmatrix};$$

5. Obtendo $\boldsymbol{f}^{(N-i+1)}(k+1)$:

$$\begin{bmatrix} \boldsymbol{0}_{M(N-i)\times M} \\ \boldsymbol{f}^{(N-i+1)}(k+1) \\ \boldsymbol{0}_{M(i-1)\times M} \\ \boldsymbol{p}_{i-1}(k+1) \end{bmatrix} = \boldsymbol{Q}'_{\theta f}{}^{(N-i+1)}(k+1) \begin{bmatrix} \boldsymbol{0}_{M(N-i)\times M} \\ \boldsymbol{f}^{(N-i+2)}(k) \\ \boldsymbol{0}_{M(i-1)\times M} \\ \boldsymbol{p}_i(k+1) \end{bmatrix};$$

6. Obtendo $\boldsymbol{Q}_\theta^{(i)}(k+1)$ e $\gamma_i(k+1)$:

$$\boldsymbol{Q}_\theta^{(i)}(k+1) \begin{bmatrix} \gamma_{i-1}(k+1) \\ \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \gamma_i(k+1) \\ \boldsymbol{f}^{(N-i+2)}(k+1) \end{bmatrix};$$

7. Estimação Conjunta:

$$\begin{bmatrix} e_{q1}^{(i)}(k+1) \\ \boldsymbol{d}_{q2}^{(N-i+1)}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta^{(i)}(k+1) \begin{bmatrix} e_{q1}^{(i-1)}(k+1) \\ \lambda^{1/2}\boldsymbol{d}_{q2}^{(N-i+1)}(k) \end{bmatrix};$$

}

8. Obtendo o erro *a priori*:

$\varepsilon(k+1) = e_{q1}^*(k+1)/\gamma(k+1)$;

}

TAB. 5.4: Pseudo-código da etapa 4: obtendo $\boldsymbol{Q}'_{\theta f}{}^{(N-i+1)}(k+1)$.

$\vdots$
for i = 1:N
{
$\vdots$

3. Obtendo $\boldsymbol{Q}'_{\theta f}{}^{(N-i+1)}(k+1)$:

$\boldsymbol{E}_f^{(i-1)} = \boldsymbol{E}_f^{(i)}$;

$temp\_\boldsymbol{D}_{fq2} = \boldsymbol{D}_{fq2}$;

for $j = 1 : M$

{ for $c = 1 : M$

{ $temp\_\boldsymbol{E}_f^{(i-1)}(M-c+1,c) = \boldsymbol{E}_f^{(i-1)}(M-c+1,c)$;

$\boldsymbol{E}_f^{(i-1)}(M-c+1,c) = \sqrt{\left|\boldsymbol{E}_f^{(i-1)}(M-c+1,c)\right|^2 + |temp\_\boldsymbol{D}_{fq2}(M*(N-j+1)-i+1,c)|^2}$;

$\cos\theta'_f(M*(N-j+1)-i+1,c) = \left|\frac{temp\_\boldsymbol{E}_f^{(i-1)}(M-c+1,c)}{\boldsymbol{E}_f^{(i-1)}(M-c+1,c)}\right|$;

$\sin\theta'_f(M*(N-j+1)-i+1,c) = conj\left(\frac{\cos\theta'_f(M*(N-j+1)-i+1,c)*temp\_\boldsymbol{D}_{fq2}(M*(N-j+1)-i+1,c)}{temp\_\boldsymbol{E}_f^{(i-1)}(M-c+1,c)}\right)$;

$aux = \begin{bmatrix} temp\_\boldsymbol{D}_{fq2}(M*(N-j+1)-i+1,:) \\ \boldsymbol{E}_f^{(i-1)}(M-c+1,:) \end{bmatrix}$;

$aux = row\_rot(aux, \cos\theta'_f(M*(N-j+1)-i+1,c), \sin\theta'_f(M*(N-j+1)-i+1,c))$;

$temp\_\boldsymbol{D}_{fq2}(M*(N-j+1)-i+1,:) = aux(1,:)$;

$\boldsymbol{E}_f^{(i-1)}(M-c+1,:) = aux(2,:)$;

}

}

$\vdots$

}

$\vdots$

Obs: A função $row\_rot()$ efetua as rotações usando os parâmetros $\cos\theta'_f$ e $\sin\theta'_f$ para construir a matriz ortogonal com as rotações de Givens (GOLUB, 1983).

$function\ \ \boldsymbol{A} = row\_rot(\boldsymbol{A}, \cos\theta, \sin\theta)$
$[m,n] = size(\boldsymbol{A})$;
for $i = 1 : n$
{$temp = \boldsymbol{A}(1,i)$;
$\boldsymbol{A}(1,i) = \cos\theta temp - \sin^*\theta \boldsymbol{A}(2,i)$;
$\boldsymbol{A}(2,i) = \sin\theta temp + \cos\theta A(2,i)$;
}

TAB. 5.5: O Algoritmo Multicanal rápido para canais de ordens distintas (*erro a posteriori*)—Estrutura Transversal.

Inicializações:
$\boldsymbol{d}_{fq2}^{(i)} = zeros(N,1)$; $\quad \boldsymbol{f}^{(M)}(0) = \mathbf{0}$; $\quad \boldsymbol{d}_{q2} = \mathbf{0}$;
$\gamma_N^{(0)}(0) = 1$; $\quad e_{f_N}^{(i)}(0) = \mu$; $\quad i = 1, 2, \cdots, M$.
Todos os cossenos = 1; todos os senos = 0;
for $k = 1, 2, \cdots$
{ $\gamma_0^{(1)} = 1$; $\quad e_{q1}^{(0)}(k+1) = d^*(k+1)$;
for $i = 1 : M$,
{ $e_{fq1_0}^{(i)}(k+1) = x_i^*(k+1)$;
for $j = 1 : N$, % Obtendo $e_{fq1}^{(i)}(k+1)$ e $\boldsymbol{d}_{fq2}^{(i)}(k+1)$:
$\{e_{fq1_j}^{(i)}(k+1) = \cos\left[\theta_j^{(i-1)}(k)\right] e_{fq1_{j-1}}^{(i)}(k+1) + \lambda^{1/2}\sin\left[\theta_j^{(i-1)}(k)\right] \boldsymbol{d}_{fq2_{N-j+1}}^{(i)}(k)$;
$\boldsymbol{d}_{fq2_{N-j+1}}^{(i)}(k+1) = \lambda^{1/2}\cos\left[\theta_j^{(i-1)}(k)\right] \boldsymbol{d}_{fq2_{N-j+1}}^{(i)}(k) - \sin^*\left[\theta_j^{(i-1)}(k)\right] e_{fq1_{j-1}}^{(i)}(k+1)$;
}
$|e_{f_N}^{(i)}(k+1)| = \sqrt{\left(\lambda^{1/2}|e_{f_N}^{(i)}(k)|\right)^2 + |e_{fq1_N}^{(i)}(k+1)|^2}$;
for $j = N : -1 : p_i$, % Obtendo $\boldsymbol{Q}'_{\theta f}{}^{(i)}(k+1)$:
$\{e_{f_{j-1}}^{(i)}(k+1) = \sqrt{|e_{f_j}^{(i)}(k+1)|^2 + |\boldsymbol{d}_{fq2_{N-j+1}}^{(i)}(k+1)|^2}$;
$\cos\theta'_{f_j}{}^{(i)}(k+1) = |e_{f_j}^{(i)}(k+1)|/|e_{f_{j-1}}^{(i)}(k+1)|$;
$\sin\theta'_{f_j}{}^{(i)}(k+1) = \left[\cos\theta'_{f_j}{}^{(i)}(k+1)\boldsymbol{d}_{fq2_{N-j+1}}^{(i)}(k+1)/e_{f_j}^{(i)}(k+1)\right]^*$;
}
$p_N^{(i)}(k+1) = \gamma_N^{(i-1)}(k)\left[e_{fq1_N}^{(i)}(k+1)\right]^*/|e_{f_N}^{(i)}(k+1)|$;
for $j = N : -1 : p_i$, % Obtendo $\boldsymbol{f}^{(i)}(k+1)$:
$\{\boldsymbol{f}_{N-j+1}^{(i)}(k+1) = \cos\theta'_{f_j}{}^{(i)}(k+1)\boldsymbol{f}_{N-j+2}^{(i-1)}(k+1) - \left[\sin\theta'_{f_j}{}^{(i)}(k+1)\right]^* p_j^{(i)}(k+1)$;
$p_{j-1}^{(i)}(k+1) = \sin\theta'_{f_j}{}^{(i)}(k+1)\boldsymbol{f}_{N-j+2}^{(i-1)}(k+1) + \cos\theta'_{f_j}{}^{(i)}(k+1)p_j^{(i)}(k+1)$;
}
$\boldsymbol{f}_{N+1-p_i+1}^{(i)}(k+1) = p_{p_i-1}^{(i)}(k+1)$;
for $j = p_i : N$, % Obtendo $\boldsymbol{Q}_\theta^{(i)}(k)$:
$\{\sin\theta_j^{(i)}(k) = -\left[\boldsymbol{f}_{N-j+2}^{(i)}(k+1)\right]^*/\gamma_{j-1}^{(i)}$;

$\cos\theta_j^{(i)}(k) = \sqrt{1 - |\sin\theta_j^{(i)}(k)|^2}$;
$\gamma_j^{(i)}(k) = \cos\theta_j^{(i)}(k)\gamma_{j-1}^{(i)}(k+1)$;
}
} for i
for $j = 1 : N$, % Processo de estimação conjunta:
$\{e_{q1}^{(j)}(k+1) = \cos\theta_j^{(0)}(k+1)\boldsymbol{e}_{q1}^{(j-1)}(k+1) + \lambda^{1/2}\sin\theta_j^{(0)}(k+1)\boldsymbol{d}_{q2}^{(N-j+1)}(k)$;
$\boldsymbol{d}_{q2}^{(N-j+1)}(k+1) = \lambda^{1/2}\cos\theta_j^{(0)}(k+1)\boldsymbol{d}_{q2}^{(N-j+1)}(k) - \left[\sin\theta_j^{(0)}(k+1)\right]^* e_{q1}^{(j-1)}(k+1)$;
}
$\varepsilon(k+1) = \left[e_{q1}^{(N)}(k+1)\right]^*/\gamma_N^{(0)}(k+1)$; % erro *a priori*;
} for k

Obs.: $\theta_j^{(M)}(k) = \theta_j^{(0)}(k+1)$ and $\boldsymbol{f}_{N-j+2}^{(M)}(k) = \boldsymbol{f}_{N-j+2}^{(0)}(k+1)$.
O *asterisco* $(*)$ denota complexo conjugado.

TAB. 5.6: O Algoritmo Multicanal rápido para canais de ordens diferentes (*erro a posteriori*)—Estrutura em Treliça.

Inicializações:

$\boldsymbol{d}_{fq2}^{(i)} = zeros(N,1); \quad \boldsymbol{f}^{(M)}(0) = \mathbf{0}; \quad \boldsymbol{d}_{q2} = \mathbf{0};$

$\gamma_N^{(0)}(0) = 1; \quad e_{f_N}^{(i)}(0) = \mu; \quad i = 1, 2, \cdots, M.$

Todos os cossenos = 1; todos os senos = 0;

for $k = 1, 2, \cdots$

{ $\gamma_0^{(1)} = 1; \quad e_{q1}^{(0)}(k+1) = d^*(k+1);$

$|e_0^{(1)}(k+1)| = \sqrt{\left(\lambda^{1/2}|e_0^{(1)}(k)|\right)^2 + |x_1(k+1)|^2};$

$\boldsymbol{f}_{N+1}^{(1)}(k+1) = [x_1(k+1)]^* / |e_0^{(1)}(k+1)|;$

for $i = 1 : M,$

{ $e_{fq1_0}^{(i)}(k+1) = x_i^*(k+1)$

for $j = 1 : N,$

{ $e_{fq1_j}^{(i)}(k+1) = \cos\left[\theta_j^{(i-1)}(k)\right] e_{fq1_{j-1}}^{(i)}(k+1) + \lambda^{1/2} \sin\left[\theta_j^{(i-1)}(k)\right] \boldsymbol{d}_{fq2_{N-j+1}}^{(i)}(k);$

$\boldsymbol{d}_{fq2_{N-j+1}}^{(i)}(k) = \lambda^{1/2} \cos\left[\theta_j^{(i-1)}(k)\right] \boldsymbol{d}_{fq2_{N-j+1}}^{(i)}(k) - \sin\left[\theta_j^{(i-1)}(k)\right]^* e_{fq1_{j-1}}^{(i)}(k+1);$

if $j \geq p_i - 1,$

$|e_{f_j}^{(i)}(k+1)| = \sqrt{\left(\lambda^{1/2}|e_{f_j}^{(i)}(k)|\right)^2 + |e_{fq1_N}^{(i)}(k+1)|^2};$

$p_j^{(i)}(k+1) = \frac{\gamma_j^{(i-1)}(k)\left[e_{fq1_j}^{(i)}(k+1)\right]^*}{|e_{f_j}^{(i)}(k+1)|};$

if $j = p_i - 1,$

$\boldsymbol{f}_{N+1-j+1}^{(i)}(k+1) = p_j^{(i)}(k+1);$

if $j > p_i - 1,$

$\cos\theta'^{(i)}_{f_j}(k+1) = |e_{f_j}^{(i)}(k+1)| / |e_{f_{j-1}}^{(i)}(k+1)|;$

$\sin\theta'^{(i)}_{f_j}(k+1) = \left[\cos\theta'^{(i)}_{f_j}(k+1)\boldsymbol{d}_{fq2_{N-j+1}}^{(i)}(k+1) / e_{f_j}^{(i)}(k+1)\right]^*;$

$\boldsymbol{f}_{N-j+1}^{(i)}(k+1) = \cos\theta'^{(i)}_{f_j}(k+1)\boldsymbol{f}_{N-j+2}^{(i-1)}(k+1) - \sin\left[\theta'^{(i)}_{f_j}(k+1)\right]^* p_j^{(i)}(k+1);$

$\sin\theta_j^{(i)}(k) = -\left[\boldsymbol{f}_{N-j+2}^{(i)}(k+1)\right]^* / \gamma_{j-1}^{(i)};$

$\cos\theta_j^{(i)}(k) = \sqrt{1 - |\sin\theta_j^{(i)}(k)|^2};$

$\gamma_j^{(i)}(k) = \cos\theta_j^{(i)}(k)\gamma_{j-1}^{(i)}(k+1);$

} do for j

} do for i

for $j = 1 : N$ % Processo de Estimação Conjunta:

{ $e_{q1}^{(j)}(k+1) = \cos\theta_j^{(0)}(k+1)\boldsymbol{e}_{q1}^{(j-1)}(k+1) + \lambda^{1/2} \sin\theta_j^{(0)}(k+1)\boldsymbol{d}_{q2}^{(N-j+1)}(k);$

$\boldsymbol{d}_{q2}^{(N-j+1)}(k+1) = \lambda^{1/2} \cos\theta_j^{(0)}(k+1)\boldsymbol{d}_{q2}^{(N-j+1)}(k) - \sin\left[\theta_j^{(0)}(k+1)\right]^* e_{q1}^{(j-1)}(k+1);$

}

$\varepsilon(k+1) = \left[e_{q1}^{(N)}(k+1)\right]^* / \gamma_N^{(0)}(k+1);$ % erro *a priori*;

} do for k

Obs.: $\theta_j^{(M)}(k) = \theta_j^{(0)}(k+1)$ e $\boldsymbol{f}_{N-j+2}^{(M)}(k) = \boldsymbol{f}_{N-j+2}^{(0)}(k+1)$.

O *asterisco* $(*)$ denota complexo conjugado.

Neste capítulo é avaliado o desempenho dos algoritmos apresentados no Capítulo 5 quanto à complexidade computacional, à velocidade de convergência e à estabilidade. Além da aplicação em *beamforming*, como discutido no Capítulo 4, será avaliado ainda o desempenho do algoritmo multicanal para canais de ordens diferentes numa aplicação de filtragem não linear do tipo Volterra, cujo modelamento pode ser implementado eficientemente usando-se os algoritmos multicanais para canais com ordens distintas, vistos no capítulo anterior.

Este capítulo está dividido como segue. Na Seção 6.1, é avaliada a complexidade computacional dos algoritmos multicanais propostos no Capítulo 5 em comparação com outras versões propostas anteriormente. Na Seção 6.2, o desempenho destes algoritmos multicanais é avaliado nos dois tipos de aplicações supra-citadas. Algumas conclusões são resumidas na Seção 6.3.

6.1 COMPLEXIDADE COMPUTACIONAL DOS ALGORITMOS MULTICANAIS RÁPIDOS

A complexidade computacional dos algoritmos multicanais rápidos abordados neste trabalho está resumida na TAB. 6.1. Observa-se que todos os algoritmos baseados na atualização do erro de predição regressiva ou *backward a posteriori* apresentam uma complexidade computacional menor em relação aos seus correspondentes baseados na atualização do erro de predição regressiva *a priori*.

Os algoritmos multicanais avaliados aqui são derivados utilizando-se duas abordagens distintas para a construção do vetor de entrada: a primeira, que dá origem aos algoritmos multicanais *em bloco*, é a mais intuitiva e foi a originariamente utilizada para derivar algoritmos multicanais rápidos da família RLS baseados na decomposição QR; a segunda, usada em (RONTOGIANNIS, 1998) para derivar algoritmos multicanais baseados na atualização do erro de predição regressiva *a priori*, não é tão intuitiva como a primeira mas mostrou-se mais eficiente por reduzir a complexidade dos algoritmos consideravelmente, como pode ser observado na TAB. 6.1.

Cada algoritmo possui duas formas distintas: uma direta (transversal) e outra recur-

TAB. 6.1: Complexidade Computacional dos Algoritmos Multicanais Rápidos

ALGORITMO	MULTIPLICAÇÕES	DIVISÕES	R. QUADRADAS
Algoritmo da TAB. 5.2	$4M^3N + 7M^2N + 12MN + 5M^2 + 5M$	$2M^2N + MN + 2M$	$M^2N + MN + M$
$\hookrightarrow$ Correspondente *a priori*	$4M^3N + 7M^2N + 13MN + 5M^2 + 6M$	$2M^2N + 2MN + 3M$	$M^2N + MN + M$
Algoritmo da TAB. 5.3	$4M^3N + 17M^2N + 12MN + 5M^2 + 5M$	$2M^2N + 3MN + 2M$	$M^2N + 2MN + M$
$\hookrightarrow$ Correspondente *a priori*†	$4M^3N + 17M^2N + 14MN + 5M^2 + 6M$	$2M^2N + 5MN + 3M$	$M^2N + 2MN + M$
Algoritmo da TAB. 5.5	$14NM + 13M + 5N - 9\sum_{i=1}^{M} p_i$	$3NM + 4M - 3\sum_{i=1}^{M} p_i$	$2NM + 3M - 2\sum_{i=1}^{M} p_i$
$\hookrightarrow$ Correspondente *a priori*†	$15NM + 14M + 5N - 10\sum_{i=1}^{M} p_i$	$4NM + 5M - 4\sum_{i=1}^{M} p_i$	$2NM + 3M - 2\sum_{i=1}^{M} p_i$
Algoritmo da TAB. 5.6	$14NM + 13M + 5N - 9\sum_{i=1}^{M} p_i$	$4NM + 5M - 4\sum_{i=1}^{M} p_i$	$2NM + 3M - 2\sum_{i=1}^{M} p_i$
$\hookrightarrow$ Correspondente *a priori*†	$15NM + 14M + 5N - 10\sum_{i=1}^{M} p_i$	$5NM + 6M - 5\sum_{i=1}^{M} p_i$	$2NM + 3M - 2\sum_{i=1}^{M} p_i$

†(RONTOGIANNIS, 1998)

siva (treliça). Esta última, mais apropriada para implementações práticas usando arrays sistólicos (ASAI, 2000; MOONEN, 1993), acarreta sempre um pequeno aumento na complexidade computacional em relação à sua correspondente forma direta. Na próxima seção, serão apresentados exemplos numéricos que permitem uma melhor visualização deste aspecto.

6.2 RESULTADOS DAS SIMULAÇÕES

Nesta seção, o desempenho dos algoritmos multicanais rápidos propostos é avaliado, tanto no ambiente *beamforming* descrito no Capítulo 4 como numa aplicação usando filtragem de Volterra de segunda ordem, para o caso dos algoritmos multicanais para canais de ordens distintas.

6.2.1 SIMULAÇÕES EM AMBIENTE *BEAMFORMING*

Todos os algoritmos Multicanais propostos neste trabalho foram testados no mesmo ambiente *beamforming* descrito no Capítulo 4. Observa-se que não são conhecidas versões com restrições na forma direta dos algoritmos multicanais rápidos tratados neste trabalho. No entanto, suas excelentes propriedades de velocidade de convergência e estabilidade

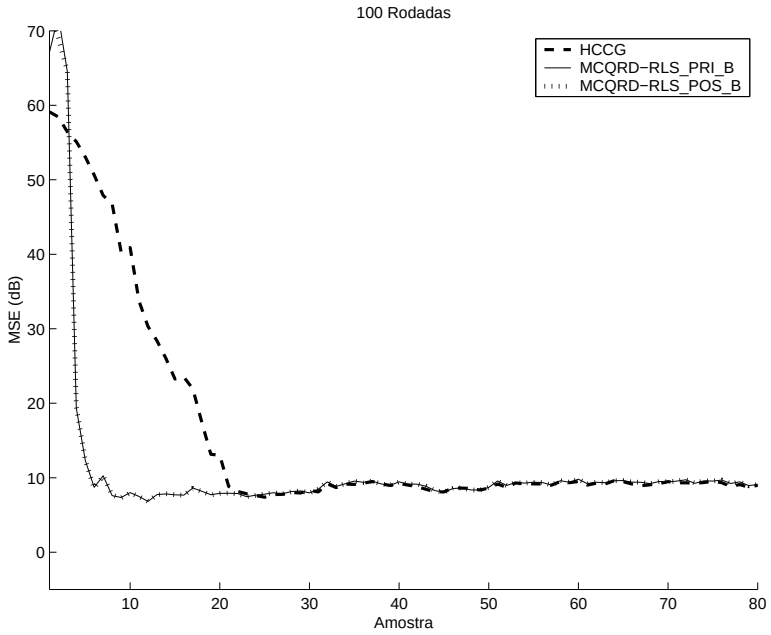

FIG. 6.1: Curva de convergência dos algoritmos multicanais rápidos num ambiente de *beamforming*.

numérica podem ser aproveitadas fazendo-se uso de estruturas como a GSC vista no Capítulo 3.

Nas simulações realizadas neste ambiente, verificou-se que todos estes algoritmos apresentaram desempenho comparável ao GSC-IQRD-RLS (GSC *Inverso* QRD-RLS) e ao HCIQRD-RLS (Householder *Constrained* IQRD-RLS) em termos de velocidade de convergência, como esperado. Isto sugere um *sample support* (a capacidade de conformar o feixe desejado com um número de amostras tão pequeno quanto possível) equivalente. Em aplicações de *beamforming*, em que o ângulo de chegada de interesse está mudando constantemente, o desempenho destes algoritmos faz deles opções atrativas. A única ressalva que pode ser colocada com relação à utilização destes algoritmos é quando se trata de aplicações em que seja necessário ter disponível o vetor de coeficientes da resposta do filtro adaptativo, o que não é possível com os algoritmos da família QR Rápidos.

A curva de aprendizagem destes algoritmos num ambiente *beamforming* pode ser vista na FIG. 6.1, em comparação com o algoritmo CCG. Pode-se observar que os algoritmos multicanais apresentam uma convergência mais rápida em relação ao algoritmo CCG, bem como em relação às versões HCCG e GSC-CG, como aconteceu no **mesmo experimento**

TAB. 6.2: Complexidade Computacional dos Algoritmos Multicanais Rápidos no ambiente *beamforming*: exemplo numérico.

ALGORITMO	MULTIP.	DIVISÕES	RAÍZES QUAD.
Algoritmo da TAB. 5.2	1404	90	48
$\hookrightarrow$ Correspondente *a priori*	1416	102	48
Algoritmo da TAB. 5.3	1765	102	54
$\hookrightarrow$ Correspondente *a priori*†	1782	120	54
Algoritmo da TAB. 5.5	**429**	**69**	48
$\hookrightarrow$ Correspondente *a priori*†	450	90	48
Algoritmo da TAB. 5.6	**429**	**90**	48
$\hookrightarrow$ Correspondente *a priori*†	450	111	48
HCIQRD-RLS	**346**	**13**	**6**
Householder CCG (HCCG)	**249**	**1**	**0**

†(RONTOGIANNIS, 1998)

realizado no Capítulo 4 em relação aos algoritmos GSC-IQRD-RLS e HCIQRD-RLS.

Foi constatado neste experimento que todos os algoritmos multicanais rápidos tratados apresentaram as mesmas curvas de aprendizagem e as mesmas características de estabilidade. Durante o experimento realizado com o mesmo número de amostras (200.000) e 10 experimentos independentes, como feito anteriormente no Capítulo 4, nenhum destes algoritmos apresentou sinais de divergência ou de instabilidade. A complexidade computacional dos mesmos neste experimento está resumida na TAB. 6.2, onde também se encontram as dos algoritmos de melhor desempenho (CG e IQRD-RLS) vistos no Capítulo 4. É importante lembrar que estes algoritmos são os que, dentre os avaliados no Capítulo 4, apresentaram menor complexidade. Não foi possível reduzir a complexidade computacional dos algoritmos multicanais a ponto de superar a relativa simplicidade dos algoritmos CCG (*Constrained Conjugate Gradient*) e as implementações nas estruturas GSC e *Householder Constrained* dos algoritmos CG e IQRD-RLS para este caso particular em que $N = 1$. Porém, vale a pena ressaltar que todos estes algoritmos multicanais mostraram-se excelentes quanto à robustez e à velocidade de convergência e para $N \gg M$, como ilustrado na TAB. 6.3, os algoritmos multicanais se tornam opções bastante atrativas.

Pelas TAB. 6.2 e TAB. 6.3, nota-se a nítida tendência de todos os algoritmos multicanais baseados na atualização do vetor de erros de predição regressiva *a posteriori*

TAB. 6.3: Comp. Comput. dos Algoritmos MCFQRD-RLS: exemplo numérico para $M = 4$ e $N = 50$.

ALGORITMO	MULTIP.	DIV.	R. QUAD.
MCFQRD-RLS em bloco *a priori* (Transv.)	8560	2208	1004
MCFQRD-RLS Seqüencial *a posteriori* (Transv.)	**3012**	**586**	**392**
IQRD-RLS	13001	100	50
Conjugate Gradient (CG)	12951	1	0

apresentarem uma complexidade computacional menor do que a dos baseados no erro a *priori*. Dentre eles, os que destacam como as melhores opções são o da TAB. 5.5, para a forma direta, e o da TAB. 5.6 para a forma recursiva, ambos propostos neste trabalho. Observa-se, ainda, que o algoritmo recursivo da TAB. 5.6 possui uma complexidade computacional (quanto a multiplicações e divisões) um pouco menor do que o correspondente *a priori* na forma direta proposto em (RONTOGIANNIS, 1998).

6.2.2 FILTRAGEM DE VOLTERRA

Nesta seção, serão apresentados de forma resumida alguns conceitos básicos da filtragem de Volterra. Será visto, também, como este problema pode ser abordado numa configuração multicanal em cujo resultante, os canais possuem ordens distintas, o que é propício para a utilização dos algoritmos multicanais das TAB. 5.5 e TAB. 5.6, introduzidos no Capítulo 5.

6.2.2.1 O PROBLEMA DA FILTRAGEM NÃO LINEAR

O princípio da superposição, obedecido por todo e qualquer sistema linear, é definido como:

$$\mathcal{L}\{\alpha x_1(k) + \beta x_2(k)\} = \alpha\mathcal{L}\{x_1(k)\} + \beta\mathcal{L}\{x_2(k)\}. \tag{6.1}$$

Qualquer sistema que não satisfaça este princípio é dito *não linear*. Filtros não lineares encontram aplicação em diversas áreas tais como: sistemas de comunicações, melhoria de voz e imagens corrompidas por ruído, modelamento de distorções introduzidos por canais etc. (MATHEWS, 2000).

Devido às particularidades inerentes aos diversos tipos de sistemas *não lineares*, estes são, normalmente, divididos em classes para facilitar a abordagem e o modelamento

matemático. Em alguns casos, o uso de redes neurais (HAYKIN, 2001) apresenta bons resultados para solucionar problemas que envolvem *não linearidades*. O uso de algoritmos adaptativos, como os dois últimos propostos no Capítulo 5 (algoritmos Multicanais Rápidos com ordens distintas para os diferentes canais), é mais apropriado para a classe de sistemas *não lineares* conhecida como sistemas polinomiais, baseados na expansão da série de Volterra (MATHEWS, 2000; RUGH, 1981; SCHETZEN, 1989).

6.2.2.2 EXPANSÃO DA SÉRIE DE VOLTERRA

O modelo da série de Volterra é muito utilizado na prática por permitir que se utilize, sem maiores restrições, a formulação clássica da filtragem adaptativa aplicada aos sistemas que podem ser modelados por ele. O modelamento de um sistema não linear pela expansão da série de Volterra pode ser expressa através da seguinte relação:

$$\begin{aligned} d(k) \;=\; & w_0 + \sum_{n_1=0}^{L-1} w_1(n_1)x(k-n_1) \\ & + \sum_{n_1=0}^{L-1}\sum_{n_2=0}^{L-1} w_2(n_1,n_2)x(k-n_1)x(k-n_2) \\ & + \sum_{n_1=0}^{L-1}\sum_{n_2=0}^{L-1}\sum_{n_3=0}^{L-1} w_3(n_1,n_2,n_3)x(k-n_1)x(k-n_2)x(k-n_3) + \cdots \\ & + \sum_{n_1=0}^{L-1}\sum_{n_2=0}^{L-1} \cdots \\ & \sum_{n_i=0}^{L-1} w_i(n_1,n_2,\cdots,n_i)x(k-n_1)x(k-n_2)\cdots x(k-n_i) \\ & + \cdots \end{aligned} \tag{6.2}$$

onde $w_i(n_1, n_2, \cdots, n_i)$ para $i = 0, 1, \cdots, L$ são os coeficientes do modelo do filtro *não linear* baseado na série de Volterra e $d(k)$ representa o sinal desejado ou a saída do sistema desconhecido (para um caso de identificação de sistema), sem a presença de ruído aditivo.

Para o caso de um problema de filtragem de Volterra truncado de segunda ordem, usado como exemplo para testar o desempenho dos dois últimos algoritmos propostos neste trabalho (que lidam com o caso de canais de ordens diferentes), a EQ. 6.2 se reduz a

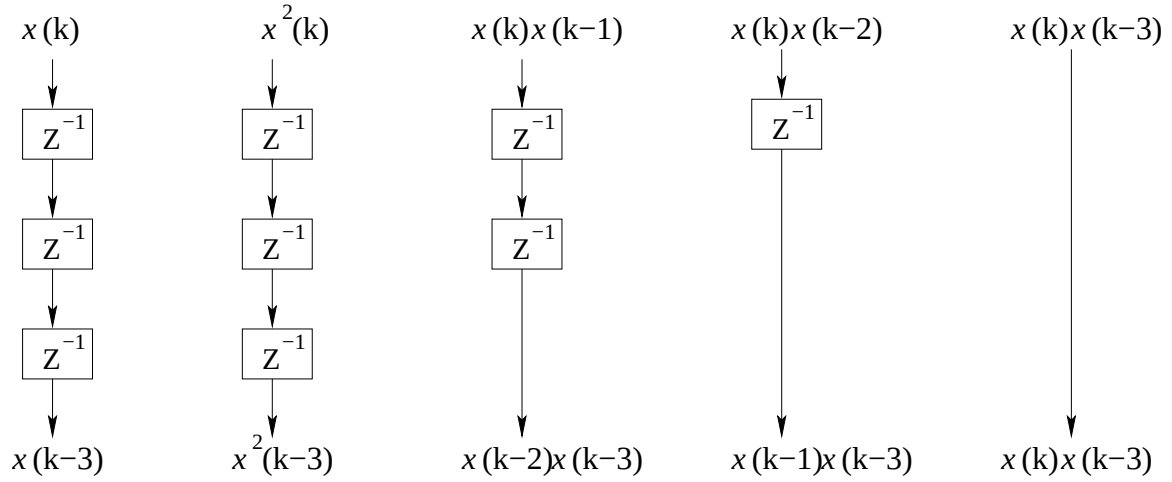

FIG. 6.2: A filtragem de Volterra de segunda ordem como um problema multicanal com $L = 4$.

$$
\begin{aligned}
d(k) &= \sum_{n_1=0}^{L-1} w_1(n_1)x(k-n_1) \\
&\quad + \sum_{n_1=0}^{L-1}\sum_{n_2=0}^{L-1} w_2(n_1,n_2)x(k-n_1)x(k-n_2) + \rho(k),
\end{aligned} \tag{6.3}
$$

Na equação acima, $w_{n_1}(k)$ e $w_{n_1,n_2}(k)$ representam, respectivamente, os coeficientes lineares e não lineares (quadráticos, neste caso) do sistema no instante de tempo k e a parcela $\rho(k)$ representa um ruído aditivo. Na EQ. 6.2, w_0 é uma constante e a sua não inclusão na EQ. 6.3 não afeta o resultado.

6.2.2.3 A FILTRAGEM DE VOLTERRA COMO UM PROBLEMA MULTICANAL

O problema expresso pela EQ. 6.3 pode ser facilmente transformado em um problema de filtragem adaptativa multicanal com $M = L + 1$ canais. A amostra mais recente de cada um destes canais pode ser dada por

$$
x_i(k) = \begin{cases} x(k), & i = 1 \\ x(k)x(k-i+2), & i = 2, \cdots, L+1 \end{cases}
$$

e a ordem, ou comprimento, de cada um dos canais, para este problema, pode ser expressa como

$$N_r = \begin{cases} L, & r = 1, 2 \\ L - r + 2, & r = 3, \cdots, L + 1 \end{cases} .$$

A FIG. 6.2 ilustra um caso particular de um sistema de Volterra de segunda ordem com $L = 4$ como um problema multicanal. Aqui pode-se visualizar claramente a diferença de ordem entre os canais. Numa situação como esta (problema multicanal com canais de ordens diferentes), os dois últimos algoritmos propostos na Seção 5.6 do capítulo anterior são particularmente úteis. Para este caso específico, tem-se que $N_1 = 4$, $N_2 = 4$, $N_3 = 3$, $N_4 = 2$ e $N_5 = 1$. Logo, $N = \sum_{r=1}^{M} N_r = 14$, com N_r representando a ordem do r-ésimo canal.

Agora, relembrando a forma de construção do vetor de entrada abordada no capítulo anterior, tem-se: $N_1 - N_2 = 4 - 4 = 0$ amostra mais recente do primeiro canal para serem os primeiros elementos de $\boldsymbol{x}_N$, seguido pelo $N_2 - N_3 = 4 - 3 = 1$ par de amostras do primeiro e do segundo canais, seguidos por $N_3 - N_4 = 3 - 2 = 1$ trio de amostras dos primeiros três canais, seguidos por $N_4 - N_5 = 2 - 1 = 1$ quarteto de amostras dos quatro primeiros canais, seguidos por $N_5 = 1$ quinteto de amostras de todos os cinco canais.

Assim sendo, o vetor resultante fica como na FIG. 6.3. As posições das amostras para o instante $k + 1$ de cada canal neste vetor, como visto no capítulo anterior, são dadas por

$$p_i = \sum_{r=1}^{i-1} r(N_r - N_{r+1}) + i \qquad i = 1, 2, \cdots, M$$

Como no exemplo presente $M = L + 1 = 4 + 1 = 5$, cada p_i será um elemento do vetor $[1 \quad 2 \quad 5 \quad 9 \quad 14]^T$. Vale a pena lembrar que, com a introdução de cada nova amostra no vetor $\boldsymbol{x}_N$, a ordem é aumentada de uma unidade até atingir o valor $N + M$, a cada instante k.

6.2.2.4 SIMULAÇÕES PARA A FILTRAGEM DE VOLTERRA

As duas últimas versões do algoritmo multicanal rápido apresentadas na Seção 5.6 (TAB. 5.5 e TAB. 5.6) que se aplicam a problemas multicanais, com canais de ordens diferentes, foram usadas no problema de filtragem de Volterra de segunda ordem visto como um problema multicanal com canais de ordens diferentes.

Trata-se de uma identificação de sistema com $L = 4$. Portanto, todo o detalhamento feito em 6.2.2.3 se aplica aqui. O vetor de coeficientes ótimo utilizado no experimento é

$$\boldsymbol{x}_{\mathbf{N}}(k) = \begin{bmatrix} x(k) \\ x^2(k) \\ x(k-1) \\ x^2(k-1) \\ x(k)x(k-1) \\ x(k-2) \\ x^2(k-2) \\ x(k-1)x(k-2) \\ x(k)x(k-2) \\ x(k-3) \\ x^2(k-3) \\ x(k-2)x(k-3) \\ x(k-1)x(k-3) \\ x(k)x(k-3) \end{bmatrix}$$

$\mathrm{N}_2 - \mathrm{N}_3 = 1$ par de amostras do primeiro e do segundo canais;

$\mathrm{N}_3 - \mathrm{N}_4 = 1$ trio de amostras dos primeiros tres canais;

$\mathrm{N}_4 - \mathrm{N}_5 = 1$ quarteto de amostras dos quatro primeiros canais;

$\mathrm{N}_5 = 1$ quinteto de amostras de todos os cinco canais.

FIG. 6.3: O vetor de entrada $\boldsymbol{x}_N$.

$\boldsymbol{w}_{ot} = [-0,78; -1,48; -1,39; 0,04; 0,54; 3,72; 1,86; -0,76; -1,62; 0,76; -0,12; 1,41; -1,52; -0,13]$; o fator de esquecimento, λ, foi de $0,98$ e a variância do ruído aditivo, σ_n^2, foi de 10^{-6}.

A FIG. 6.4 mostra a curva de aprendizagem dos algoritmos na aplicação em questão. À semelhança da aplicação anterior, aqui também os algoritmos multicanais rápidos apresentaram a mesma curva de aprendizagem, como esperado. Nesta curva percebe-se nitidamente a maior velocidade de convergência destes algoritmos em relação ao LMS Normalizado (NLMS–do inglês *Normalized Least Mean Square*). O fraco desempenho dos algoritmos LMS e NLMS neste tipo de aplicação, em comparação com algoritmos de convergência rápida da família RLS, já era conhecido, em particular para o caso de uma filtragem de Volterra onde isto ocorre mesmo se o sinal de entrada é ruído branco. Uma abordagem deste tópico está disponível em (MATHEWS, 2000).

A TAB. 6.4 ilustra a complexidade computacional destes algoritmos multicanais aplicados ao problema da filtragem de Volterra. Aqui também, todos os algoritmos baseados

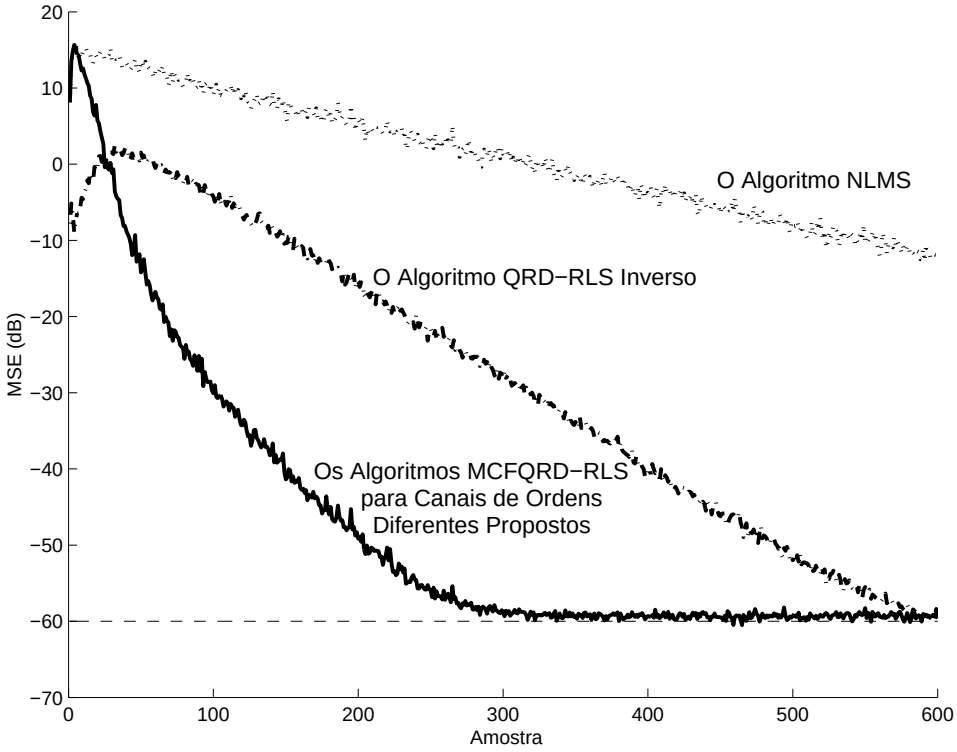

FIG. 6.4: Curva de convergência do problema da filtragem de Volterra de segunda ordem para alguns algoritmos.

na atualização do erro da predição regressiva *a posteriori* apresentaram uma complexidade computacional menor, em termos de divisões e multiplicações, em relação aos baseados na atualização do erro da predição regressiva *a priori*.

6.3 RESUMO

Neste capítulo foram apresentados os resultados da avaliação do desempenho dos algoritmos propostos no capítulo anterior. Pôde-se constatar que os algoritmos multicanais rápidos apresentam um bom desempenho no que diz respeito à velocidade de convergência. Presume-se, desta forma, um bom *sample support* em aplicações de *beamforming*, comparável ao do algoritmo IQRD-RLS, usado dentro das estruturas GSC e *Householder Constrained*, avaliado no Capítulo 4 no mesmo tipo de aplicação.

Quanto à complexidade computacional dos algoritmos multicanais propostos neste trabalho, em comparação com versões baseadas na atualização do vetor de erro de predição

TAB. 6.4: Complexidade Computacional dos Algoritmos Multicanais Rápidos na Filtragem de Volterra: exemplo numérico.

ALGORITMO	MULTIPLICAÇÕES	DIVISÕES	RAÍZES QUAD.
Algoritmo da TAB. 5.5	**836**	**137**	93
$\hookrightarrow$ Correspondente *a priori*†	880	181	93
Algoritmo da TAB. 5.6	**836**	**181**	93
$\hookrightarrow$ Correspondente *a priori*†	880	225	93
CIQRD-RLS	1121	28	14
Conjugate Gradient	1107	1	0

†(RONTOGIANNIS, 1998)

a priori progressiva propostas anteriormente em (RONTOGIANNIS, 1998), viu-se que as novas versões aqui propostas apresentam uma redução computacional considerável em termos de multiplicações e divisões.

7.1 CONCLUSÕES

Nesta dissertação, após uma breve introdução e revisão de conceitos e algoritmos básicos, com e sem restrições lineares, foi feita uma avaliação do desempenho de vários algoritmos adaptativos rápidos com restrições bem como suas correspondentes versões sem restrições, usando as estruturas GSC e Householder Constrained, numa aplicação LCMV. Pôde ser constatado que a grande maioria dos algoritmos de convergência rápida disponíveis na literatura apresentaram problemas de estabilidade neste tipo de aplicação. Algoritmos como o RLS e o *Quasi* Newton, por exemplo, apresentaram desempenho insatisfatório tanto nas versões com restrições como nas sem restrições usadas nas estruturas GSC e Householder Constrained. Dos algoritmos avaliados, o único que demonstrou estabilidade na sua versão com restrições lineares foi o do Gradiente Conjugado. O algoritmo QRD-RLS Inverso mostrou-se estável quando usado na sua versão direta dentro das estruturas supra mencionadas sendo que o mesmo não acontece com a versão com restrições (forma direta) deste algoritmo proposta em (CHERN, 2002).

A seguir, foram buscadas novas soluções com o estudo e proposta de novos algoritmos multicanais rápidos usando decomposição QR que pudessem ser utilizados no problema de *Beamforming* dentro de estruturas como a GSC ou a Householder Constrained. Foram estudados os algoritmos multicanais *a priori* e os correspondentes *a posteriori* foram obtidos.

Com base nos experimentos realizados, constatou-se que os algoritmos multicanais rápidos possuem uma velocidade de convergência comparável a do QRD-RLS Inverso que foi o mais rápido de todos na aplicação de *beamforming* realizada no Capítulo 4 (considerando-se apenas aqueles que fornecem uma solução LCMV geral e não apenas MVDR). Contudo, foi observado que os algoritmos MCFQRD-RLS investigados apresentam uma complexidade computacional maior que a dos algoritmos monocanais de convergência rápida avaliados. Verificou-se, ainda, que as três novas versões do algoritmo MCFQRD-RLS (duas na estrutura em treliça e uma na transversal) propostas neste trabalho apresentam uma redução significativa na complexidade computacional em relação

às suas versões correspondentes baseadas no erro *a priori*.

Vale a pena ressaltar o fato de que, para esta classe de algoritmos, as versões transversais apresentaram sempre uma complexidade computacional menor que as correspondentes versões em treliça. No entanto, estas últimas apresentam algumas vantagens desejáveis: maior simplicidade na implementação usando *arrays* sistólicos; e o fato de, para um problema de uma determinada ordem, esta estrutura (em treliça) fornecer, na mesma implementação, as soluções do mesmo problema para as ordens inferiores.

7.2 TRABALHOS FUTUROS

Algoritmos incorporando restrições na solução que sejam suficientemente rápidos e estáveis em aplicações de *beamformer* adaptativo, a um custo computacional atraente, continuam sendo um desafio a ser pesquisado. Com base nisto, propomos os seguintes tópicos para pesquisas futuras:

- Investigar e, se possível, solucionar o problema de instabilidade para os algoritmos que demonstraram instabilidade em aplicação de *beamforming* (ressalta-se, especialmente, o algoritmo *Quasi* Newton cuja versão real é estável);
- Por terem exibido boas propriedades como convergência rápida, estabilidade e custo computacional aceitável, seria interessante uma investigação sobre possíveis versões dos algoritmos multicanais rápidos cujas soluções incorporassem restrições lineares;
- investigar a possibilidade de implementações adaptativas dos filtros de Wiener em múltiplos estágios (GOLDSTEIN, 1998; HONIG, 2002; DE CAMPOS, 2003) incorporando restrições lineares;
- Apesar do avanço tecnológico verificado nos últimos anos, que tem permitido a construção de processadores cada vez mais rápidos e capazes de operar com uma precisão numérica relativamente grande, seria interessante verificar o comportamento destes algoritmos em ambientes de precisão finita. Alguns trabalhos neste sentido foram desenvolvidos em (KIM, 1991; SIQUEIRA, 1994) para os algoritmos baseados no RLS e mais recentemente em (APOLINÁRIO JR., 2003) para versões monocanais dos algoritmos FQRD-RLS.

Esta dissertação apresentou um panorama bastante amplo sobre os algoritmos de filtragem adaptativa de convergência rápida. Em particular, os que fazem uso da decomposição QR foram abordados com razoável profundidade e novas versões da família dos algoritmos multicanais rápidos, que fazem uso da decomposição QR, foram introduzidas. Todas as derivações foram feitas para o caso geral de ambiente complexo e os algoritmos são apresentados em pseudo–código fáceis de implementar. Isso tudo contribui para que este trabalho venha a ser uma referência útil para os interessados nesta matéria.

ALEXANDER, S. T. e GHIRNIKAR, A. L. **A method for recursive least squares filtering based upon an inverse QR decomposition**. *In: IEEE Transactions on Signal Processing*, 41:20–30, January 1993.

APOLINÁRIO JR., J. A., e DINIZ, P. S. R. **A new fast QR algorithm based on *a priori* errors**. *In: IEEE Signal Processing Letters*, 4:307–309, November 1997.

APOLINÁRIO JR., J. A. ***Novos algoritmos de filtragem adaptativa: LMS com reutilização de dados e RLS rápidos baseados na decompositção QR***. Tese de Doutorado–COPPE/Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brasil, 1998.

APOLINÁRIO JR, J. A., DE CAMPOS, M. L. R. e BERNAL, C. P. **The constrained conjugate-gradient algorithm**. *In: Signal Processing Letters*, 7:351–354, 2000b.

APOLINÁRIO JR., J. A., SIQUEIRA, M. G. e DINIZ, P. S. R. **On fast QR algorithms based on backward prediction errors: New results and comparisions**. *In: Proc. First Balkan Conf. on Signal Process., Commun., Circuits, and Systems*, June 2000a.

APOLINÁRIO JR., J. A., SIQUEIRA, M. G. e DINIZ, P. S. R. **Fast QR Algorithms Based on Backward Prediction Errors: A New Implementation and Its Finit Precision Performance**. *Birkhäuser, Systems, and Signal Processing*, 22(4):335–349, July/August 2003.

ASAI, T. e MATSUMOTO, T. **A systolic array RLS processor**. *In: IEEE 51st Vehicular Technology Conference Proceedings, VTC 2000-Spring Tokyo*, 3:2247–2251, May 2000.

BARTON, P. **Digital beamforming for radar**. *In: IEE Poc. Pt. F*, 127:266–277, August 1980.

BELLANGER, M. G. **The FLS-QR algorithm for adaptive filtering: The case of multichannel signals**. *In: Signal Processing*, 22:115–126, 1991.

CHANG, P. S. e WILLSON JR., A. N. **Adaptive filtering using modified conjugate gradient**. *In: Proceedings 38th Midwest Symposium on Circuits and Systems*, págs. 243–246, August 1995.

CHANG, P. S. e WILLSON JR., A. N. **Analysis of conjugate gradient algorithms for adaptive filtering**. *In: IEEE Transactions on Signal Processing*, 48:409–418, 2000.

CHERN, S.-J. e CHANG, C.-Y. **Adaptive linearly constrained inverse QRD-RLS beamforming algorithm for moving jammers suppression**. *In: IEEE Transactions on Antennas and Propagation*, 50(8):1138–1150, August 2002.

CIOFFI, J. M. **The fast adaptive ROTOR's RLS algorithm**. *In: IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-38:631–653, April 1990.

CURTIS, T. E. **Digital beamforming for sonar systems**. *In: IEE Poc. Pt. F*, 127: 257–265, August 1980.

DE CAMPOS, M. L. R. e ANTONIOU, A. **A new quasi-Newton adaptive filtering algorithm**. *In: IEEE Transactions on Circuits and Systems — Part II*, 44:924–934, November 1997.

DE CAMPOS, M. L. R., WERNER, S. e APOLINÁRIO JR., J. A. **Constrained adaptation algorithms employing Housholder transformation**. *In: IEEE Transactions on Signal Processing*, 50(9):2187–2195, 2002.

DE CAMPOS, M. L. R., WERNER, S. e APOLINÁRIO JR., J. A. **On an efficient implementation of the multistage wiener filter through householder reflections for DS-CDMA interference suppression**. *IEEE, Globecom*, págs. 2350–2354, 2003.

DE CAMPOS, M. L. R., WERNER, S., APOLINÁRIO JR., J. A. e LAAKSO, T. I. **Constrained quasi-Newton algorithm for CDMA mobile communications**. *International Telecommunications Symposium*, págs. 371–376, August 1998.

DINIZ, P. S. R. ***Adaptive Filtering: Algorithms and Practical Implementations***. 2nd Edition, Kluwer Academic Publishers, Boston, 2002.

FROST III, O. L. **An algorithm for linearly constrained adaptive array processing**. *In: Proceedings of IEEE*, 60:926–935, August 1972.

GALDINO, J. F., PINTO, E. L. e DE ALENCAR, M. S. **Desempenho do algoritmo LMS na identificação de canais variantes no tempo e seu emprego em esquemas de recepção MLSE-PSP**. *In: Revista da Sociedade Brasileira de Telecomunicações - SBT*, 18:273–284, 2003.

GHIRNIKAR, A. L. e ALEXANDER, S. T. **Performance and implementation of the inverse QR adaptive filter**. *In: ICASSP-92.,IEEE International Conference on Acoustics, Speech, and Signal Processing*, 4:29–32, March 1992.

GOLDSTEIN, J. S., REED, I. S. e SCHARF, L. L. **A multistage representation of the Wiener filter based on orthogonal projections**. *In: IEEE Transactions on Information Theory*, 44(7):2943–2959, November 1998.

GOLUB, G. H. e LOAN, C. F. V. ***Matrix Computations***. Baltimore: The Johns Hopkins University Press, 1983.

GRIFFITHS, L. J. e JIM, C. W. **An alternative approach to linearly constrained adaptive beamforming**. *In: IEEE Transactions on Antennas and Propagation*, AP-30:27–34, January 1982.

HAYKIN, S. ***Adaptive Filter Theory***. New Jersey: Prentice-Hall, Englewood-Cliffs, 1996.

HAYKIN, S. ***Redes Neurais–Princípios e Prática***. Segunda Edição, Bookman, Porto Alegre, 2001.

HIEMSTRA, J. D. ***Robust Implementations of the Multistage Wiener Filter***. Ph.D Thesis, Faculty of the Virginia Polytechnic Institute and State University, April 2003.

HONIG, M. L. e GOLDSTEIN, J. S. **A multistage representation of the Wiener filter based on orthogonal projections**. *In: IEEE Transactions on Communications*, 50(6):986–994, June 2002.

KIM, K.-H. e POWERS, E. J. **Analysis of initialization and numerical instability of fast RLS algorithms**. *In: ICASSP-91., International Conference on Acoustics, Speech, and Signal Processing*, 4:1861–1864, April 1991.

LING, F. **Givens rotation based least squares lattice and related algorithms**. *In: IEEE Transactions Signal Processing*, 39:1541–1551, July 1991.

LITVA, J. e LO, T. K. ***Digital Beamforming in Digital Communications***. Artech House Publishers, Boston/London, 1996.

MATHEWS, V. J. e SICURANZA, G. L. ***Polynomial Signal Processing***. Wiley–Intercience: John Wiley and Sons, 2000.

MEDINA S., C. A., APOLINÁRIO JR., J. A. e SIQUEIRA, M. G. **A unified framework for multichannel fast QRD-LS adaptive filters based on backward prediction errors**. *MWSCAS'O2, Tulsa–USA*, 3, August 2002.

MIRANDA, M. D. e GERKEN, M. . **An hybrid QR-lattice least squares algorithm using *a priori* errors**. *In:* Proceedings of the 38 Midwest Symposium on Circuits and Systems, Rio de Janeiro, RJ, Brasil:983–986, August 1995.

MOONEN, M. **Systolic MVDR beamforming with inverse updating**. *In: IEE Proceedings Radar and Signal Processing*, 140:175–178, June 1993.

MOONEN, M. e PROUDLER, I. K. **Mvdr beamforming and generalized sidelobe cancellation based on inverse updating with residual extraction**. *In: IEEE Transactions on Circuits and Systems.*, 47(2):352–358, April 2000.

N.KALOUPTSIDIS e THEODORIDIS, S. ***Adaptive Systems Identification and Signal Processing Algorithms***. Englewood Cliffs, NJ: Prentice Hall, 1993.

RAMOS, A. L. L. e APOLINÁRIO JR, J. A. **A lattice version of the multichannel FQRD algorithm based on *a posteriori* backward errors**. *In: Lecture Notes in Computer Science, ICT'2004*, 3124:488–497, August 2004a.

RAMOS, A. L. L. e APOLINÁRIO JR, J. A. **A new multiple order multichannel fast QRD algorithm and its application to non-linear system identification**. *XXI Simpósio Brasileiro de Telecomunicações, SBT 2004*, September 2004b.

RAMOS, A. L. L. e APOLINÁRIO JR, J. A. **On numerical robustness of constrained RLS–like algorithms**. *XXI Simpósio Brasileiro de Telecomunicações, SBT 2004*, September 2004c.

RAMOS, A. L. L., APOLINÁRIO JR, J. A. e SIQUEIRA, M. G. **A new order recursive multiple order multichannel fast QRD algorithm**. *Thirty-Eighth Annual Asilomar Conference on Signals, Systems, and Computers*, pág. a aparecer, November 2004d.

REGALIA, P. A. e BELLANGER, M. **On the duality between fast QR methods and lattice methods in least squares adaptive filtering**. *In: IEEE Transactions on Signal Processing*, SP–39(4):879–891, April 1991.

RESENDE, L. S., ROMANO, J. M. T. e BELLANGER, M. G. **A fast least-squares algorithm for linearly constrained adaptive filtering**. *In: IEEE Transactions on Signal Processing*, 44:1168–1174, May 1996.

RONTOGIANNIS, A. A. e THEODORIDIS, S. **New fast inverse QR least squares adaptive algorithms**. *In:* Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, 4:1412–1415, 1995.

RONTOGIANNIS, A. A. e THEODORIDIS, S. **Multichannel fast QRD-LS adaptive filtering: New technique and algorithms**. *In: IEEE Transactions on Signal Processing*, 46:2862–2876, November 1998.

RUGH, W. J. ***The Volterra-Wiener Approach***. John Hopkins University Press, Baltimore, Maryland, 1981.

SCHETZEN, M. ***The Volterra and Wiener Theories of Nonlinear Systems***. Reprint Edition, R. E. Krieger Publishing Company, Malabar, Florida, 1989.

SIQUEIRA, M. G. e DINIZ, P. S. R. **Finite precision analysis of the conventional QR decomposition RLS algorithm**. *In: Proceedings of the 37th Midwest Symposium on Circuits and Systems*, 2:990–993, August 1994.

STUTZMAN, W. L. e THIELE, G. A. ***Antenna Theory and Design***. 2nd Edition, Wiley, New York, NY, 1998.

TANG, C.-F. T. ***Adaptive array systems Using QR-Based RLS and CRLS Techniques with systolic array Architectures***. Ph.D Thesis–University of Maryland, 1991.

VAIDYANATHAN, P. P. ***Multirate Systems and Filter Banks***. Allan V. Oppenheim: Prentice Hall, 1993.

VAN VENN, B. D. e BUCKLEY, K. M. **Beamforming: A versatile approach to spatial filtering**. *In: IEEE ASSP Magazine*, págs. 4–24, April 1988.

9 APÊNDICES

9.1 APÊNDICE 1: TÓPICOS SOBRE A IMPLEMENTAÇÃO DOS ALGORITMOS QUE FAZEM USO DA DECOMPOSIÇÃO QR

Neste apêndice são detalhados alguns aspectos sobre a implementação das equações matriciais dos algoritmos que usam a decomposição QR. As considerações feitas aqui serão baseadas no algoritmo QRD-RLS convencional, mas incorporam todos os aspectos relevantes para a implementação de toda esta família de algoritmos.

Relembrando a primeira equação na TAB. 2.3 tem-se

$$\begin{bmatrix} \mathbf{0}^T \\ \boldsymbol{U}(k) \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} \boldsymbol{x}^H(k) \\ \lambda^{1/2}\boldsymbol{U}(k-1) \end{bmatrix}. \tag{9.1}$$

Como visto no Capítulo 2, $\boldsymbol{Q}_\theta(k) = \prod_{i=N}^{1} \boldsymbol{Q}_{\theta_i}(k)$. Cada um dos fatores $\boldsymbol{Q}_{\theta_i}(k)$ é responsável por zerar um elemento do vetor $\boldsymbol{x}(k)$ sobre a diagonal de $\boldsymbol{U}(k-1)$ que, de acordo com a notação usada neste trabalho, pode ter uma estrutura triangular inferior ou superior, como na ilustração da FIG. 9.1.

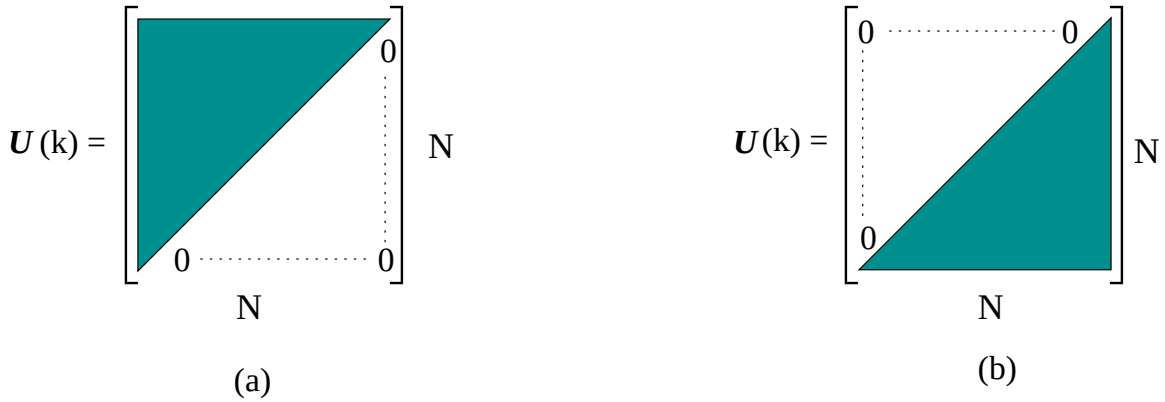

FIG. 9.1: As diferentes triangularizações de $\boldsymbol{U}(k)$: (a) SUPERIOR e (b) INFERIOR.

A estrutura das matrizes $\boldsymbol{Q}_{\theta_i}(k)$, $i = N, N-1, \cdots, 1$, para cada um dos tipos de triangularização, é dada por

$$SUPERIOR: \quad \boldsymbol{Q}_{\theta_i}(k) = \begin{bmatrix} \cos\theta_i(k) & \mathbf{0}^T & -\sin^*\theta_i(k) & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{I}_{i-1} & \mathbf{0} & \mathbf{0}\cdots\mathbf{0} \\ \sin\theta_i(k) & \mathbf{0}^T & \cos\theta_i(k) & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0}\cdots\mathbf{0} & \mathbf{0} & \boldsymbol{I}_{N-i} \end{bmatrix} \tag{9.2}$$

e

$$INFERIOR: \quad \boldsymbol{Q}_{\theta_i}(k) = \begin{bmatrix} \cos\theta_i(k) & \mathbf{0}^T & -\sin^*\theta_i(k) & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{I}_{N-i} & \mathbf{0} & \mathbf{0}\cdots\mathbf{0} \\ \sin\theta_i(k) & \mathbf{0}^T & \cos\theta_i(k) & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0}\cdots\mathbf{0} & \mathbf{0} & \boldsymbol{I}_{i-1} \end{bmatrix}. \tag{9.3}$$

Considerando como exemplo um caso em que $N = 3$, a matriz $\boldsymbol{Q}_\theta(k)$, para o caso de triangularização superior, é dada por

$$\boldsymbol{Q}_\theta(k) = \begin{bmatrix} c\theta_3 c\theta_2 c\theta_1 & -c\theta_3 c\theta_2 s^*\theta_1 & -c\theta_3 s^*\theta_2 & -s^*\theta_3 \\ s\theta_1 & c\theta_1 & 0 & 0 \\ s\theta_2 c\theta_1 & -s\theta_2 s^*\theta_1 & c\theta_2 & 0 \\ s\theta_3 c\theta_2 c\theta_1 & -s\theta_3 c\theta_2 s^*\theta_1 & -s\theta_3 s^*\theta_2 & c\theta_3 \end{bmatrix} \tag{9.4}$$

e, para triangularização inferior, por

$$\boldsymbol{Q}_\theta(k) = \begin{bmatrix} c\theta_3 c\theta_2 c\theta_1 & -s^*\theta_3 & -c\theta_3 s^*\theta_2 & -c\theta_3 c\theta_2 s^*\theta_1 \\ s\theta_3 c\theta_2 c\theta_1 & c\theta_3 & -s\theta_3 s^*\theta_2 & -s\theta_3 c\theta_2 s^*\theta_1 \\ s\theta_2 c\theta_1 & 0 & c\theta_2 & -s\theta_2 s^*\theta_1 \\ s\theta_1 & 0 & 0 & c\theta_3 \end{bmatrix}, \tag{9.5}$$

o que justifica o particionamento de $\boldsymbol{Q}_\theta(k)$ como na EQ. 2.39 no Capítulo 2.

Implementação

O pseudo-código a seguir, usando uma notação similar à usada no Matlab©, ilustra uma implementação do algoritmo QRD-RLS. Neste exemplo, considerou-se o caso em a matriz $\boldsymbol{U}(k)$ é triangular inferior, como na FIG. 9.1 (b). A implementação das equações dos outros algoritmos da família RLS que usam a decomposição QR segue o mesmo padrão ilustrado neste exemplo.Um pseudo–código semelhante para de uma das equações da TAB. 5.3 está exemplificada na TAB. 5.4. Os algoritmos das TAB. 5.5 e TAB. 5.6 já estão num formato (pseudo-código) fácil de implementar.

TAB. 9.1: Pseudo-código do algoritmo QRD-RLS.

QRD-RLS

% Inicializações:

$$\boldsymbol{U}(0) = \begin{bmatrix} 0 & \cdots & 0 & 1 \\ \vdots & & 1 & 0 \\ \vdots & \iddots & \vdots & \vdots \\ 1 & 0 & \cdots & 0 \end{bmatrix}_{N\times N}; \qquad \boldsymbol{d}_{q2} = \mathbf{1}_{N\times 1};$$

Para $k > 0$,

{

Obtendo $\boldsymbol{Q}_\theta(k)$ e atualizando $\boldsymbol{U}(k)$:

$\boldsymbol{x}' = \boldsymbol{x}^H(k)$; % O vetor de entrada no instante k

$\gamma(k) = 1$;

% Implementando a EQ. 2.36:

for $i = 1 : N$

{% I - Cálculo das rotações:

$$\cos\theta_i = \frac{\lambda^{1/2}\left|\boldsymbol{U}_{N-i+1,i}\right|}{\sqrt{\left|\boldsymbol{x}'_i\right|^2 + \lambda^{1/2}\left|\boldsymbol{U}_{N-i+1,i}\right|^2}};$$

$$\sin\theta_i = \left[\frac{\boldsymbol{x}'_i\cos\theta_i}{\lambda^{1/2}\boldsymbol{U}_{N-i+1,i}}\right]^*;$$

$\gamma(k) = \gamma(k)\cos\theta_i$;

% II - Efetuando a multiplicação matricial:

for j=1:N

{$aux = \boldsymbol{x}'_j$;

$\boldsymbol{x}'_j = \cos\theta_i aux - \lambda^{1/2}\sin^*\theta_i \boldsymbol{U}_{N-i+1,j}$;

$\boldsymbol{U}_{N-i+1,j} = aux\sin\theta_i + \lambda^{1/2}\cos\theta_i \boldsymbol{U}_{N-i+1,j}$;

}

}

% Implementando a EQ. 2.38:

$e_{q1} = d^*(k)$;

for $i = 1 : N$

{$aux = e_{q1}$;

$e_{q1} = \cos\theta_i aux - \lambda^{1/2}\sin^*\theta_i \boldsymbol{d}_{q2_{N-i+1}}$;

$\boldsymbol{d}_{q2_{N-i+1}} = \sin\theta_i aux + \lambda^{1/2}\cos\theta_i \boldsymbol{d}_{q2_{N-i+1}}$;

}

$\varepsilon(k) = e^*_{q1}/\gamma(k)$; % Erro *a priori*

}

Obs: O *asterisco* $(*)$ denota complexo conjugado.

Ao longo deste trabalho, usou-se letras minúsculas e maiúsculas em negrito para denotar, respectivamente, vetores e matrizes. O complexo conjugado de $\boldsymbol{v}^T$, ou o seu hermitiano, é indicado como

$$\boldsymbol{v}^H = (\boldsymbol{v}^T)^* \tag{9.6}$$

O *gradiente* de ξ em relação a $\boldsymbol{w}$ é definido como o vetor obtido da diferenciação da função escalar ξ em relação ao vetor $\boldsymbol{w}$. Considerando inicialmente o caso real, o gradiente é dado pelo vetor coluna seguinte.

$$\boldsymbol{\nabla_w}\xi = \begin{bmatrix} \frac{\partial \xi}{\partial w_0} \\ \frac{\partial \xi}{\partial w_1} \\ \vdots \\ \frac{\partial \xi}{\partial w_{N-1}} \end{bmatrix} \tag{9.7}$$

Da definição acima, é possível provar-se que

$$\boldsymbol{\nabla_w}(\boldsymbol{b}^T\boldsymbol{w}) = \boldsymbol{\nabla_w}(\boldsymbol{w}^T\boldsymbol{b}) = \boldsymbol{b} \tag{9.8}$$

e que

$$\boldsymbol{\nabla_w}(\boldsymbol{w}^T\boldsymbol{R}\boldsymbol{w}) = \boldsymbol{R}^T\boldsymbol{w} + \boldsymbol{R}\boldsymbol{w} \tag{9.9}$$

o qual, quando $\boldsymbol{R}$ é simétrico, conduz a

$$\boldsymbol{\nabla_w}(\boldsymbol{w}^T\boldsymbol{R}\boldsymbol{w}) = 2\boldsymbol{R}\boldsymbol{w} \tag{9.10}$$

Assumindo agora o caso complexo, o vetor $\boldsymbol{w}$ passa a ser composto por duas partes, uma real e outra imaginária, tal que $\boldsymbol{w} = \boldsymbol{a} + j\boldsymbol{b}$. Assim, o gradiente para o caso complexo é definido como segue.

$$\boldsymbol{\nabla_w}\xi = \begin{bmatrix} \frac{\partial \xi}{\partial a_0} + j\frac{\partial \xi}{\partial b_0} \\ \frac{\partial \xi}{\partial a_1} + j\frac{\partial \xi}{\partial b_1} \\ \vdots \\ \frac{\partial \xi}{\partial a_{N-1}} + j\frac{\partial \xi}{\partial b_{N-1}} \end{bmatrix} \tag{9.11}$$

o que corresponde à soma de dois gradientes reais.

$$\boldsymbol{\nabla_w}\xi = \boldsymbol{\nabla_a}\xi + j\boldsymbol{\nabla_b}\xi \tag{9.12}$$

ou, de maneira equivalente, a $2\frac{\partial \xi}{\partial \boldsymbol{w}^*}$ onde $\frac{\partial}{\partial \boldsymbol{w}}$ é a *derivada* em relação a $\boldsymbol{w}$ e é dada por

$$\frac{\partial}{\partial \boldsymbol{w}} = \frac{1}{2}\begin{bmatrix} \frac{\partial}{\partial a_0} - j\frac{\partial}{\partial b_0} \\ \frac{\partial}{\partial a_1} - j\frac{\partial}{\partial b_1} \\ \vdots \\ \frac{\partial}{\partial a_{N-1}} - j\frac{\partial}{\partial b_{N-1}} \end{bmatrix} \tag{9.13}$$

e

$$\frac{\partial}{\partial \boldsymbol{w}^*} = \frac{1}{2}\begin{bmatrix} \frac{\partial}{\partial a_0} + j\frac{\partial}{\partial b_0} \\ \frac{\partial}{\partial a_1} + j\frac{\partial}{\partial b_1} \\ \vdots \\ \frac{\partial}{\partial a_{N-1}} + j\frac{\partial}{\partial b_{N-1}} \end{bmatrix} \tag{9.14}$$

que corresponde ao *conjugado* da *derivada* em realção à $\boldsymbol{w}$.

A seguir é apresentado um exemplo do uso do gradiente complexo calculando $\boldsymbol{\nabla_w} E[e(k)e^*(k)]$ dado pela EQ. 2.1.

$$\boldsymbol{\nabla_w} E[e(k)e^*(k)] = E\{e^*(k)[\boldsymbol{\nabla_w} e(k)] + e(k)[\boldsymbol{\nabla_w} e^*(k)]\} \tag{9.15}$$

Calculando cada gradiente ...

$$\begin{aligned} \boldsymbol{\nabla_w} e(k) =& \boldsymbol{\nabla_a}[d(k) - \boldsymbol{w}^H\boldsymbol{x}(k)] + j\boldsymbol{\nabla_b}[d(k) - \boldsymbol{w}^H\boldsymbol{x}(k)] \\ =& -\boldsymbol{x}(k) - \boldsymbol{x}(k) \\ =& -2\boldsymbol{x}(k) \end{aligned} \tag{9.16}$$

e

$$\begin{aligned} \boldsymbol{\nabla_w} e^*(k) =& \boldsymbol{\nabla_a}[d^*(k) - \boldsymbol{w}^T\boldsymbol{x}^*(k)] + j\boldsymbol{\nabla_b}[d^*(k) - \boldsymbol{w}^T\boldsymbol{x}^*(k)] \\ =& -\boldsymbol{x}^*(k) + \boldsymbol{x}^*(k) \\ =& \boldsymbol{0} \end{aligned} \tag{9.17}$$

tal que o resultado final é

$$\begin{aligned}\boldsymbol{\nabla_w} E[e(k)e^*(k)] =& E\{e^*(k)[-2\boldsymbol{x}(k)] + e(k)[\mathbf{0}]\} \\ =& -2E[e^*(k)\boldsymbol{x}(k)] \\ =& -2E[\boldsymbol{x}(k)[d(k) - \boldsymbol{w}^H\boldsymbol{x}(k)]^*\} \\ =& -2E[\boldsymbol{x}(k)d(k)^*] + 2E[\boldsymbol{x}(k)\boldsymbol{x}^H(k)]\boldsymbol{w} \\ =& -2\boldsymbol{p} + 2\boldsymbol{R}\boldsymbol{w}. \end{aligned} \tag{9.18}$$