UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

UNICAMP

TESE DE DOUTORADO

CARACTERIZAÇÃO DE SINAIS UTILIZANDO-SE TRANSFORMADAS EM ALGORITMOS ADAPTATIVOS BASEADOS NO LMS

Antônio Cláudio Paschoarelli Veiga
Orientador: Prof. Dr. Yuzo Iano (FEEC/DECOM/UNICAMP)

Banca Examinadora:
Prof. Dr. Nelson Delfino D'Ávila Mascarenhas – (UFSCar/São Carlos)
Prof. Dr. Osamu Saotome – (ITA/São José dos Campos)
Prof. Dr. José Antônio Siqueira Dias – (FEEC/DEMIC/UNICAMP)
Prof. Dr. João Baptista Tadanobu Yabu-uti – (FEEC/DECOM/UNICAMP)
Prof. Dr. Edson Moschim – (FEEC/DSIF/UNICAMP)

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação como parte dos requisitos exigidos para a obtenção do título de Doutor em Engenharia Elétrica.

Campinas – SP – Brasil
Julho de 2002

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

V533c

Veiga, Antônio Cláudio Paschoarelli
Caracterização de sinais utilizando-se transformadas em algoritmos adaptativos baseados no LMS / Antônio Cláudio Paschoarelli Veiga. --Campinas, SP: [s.n.], 2002.

Orientador: Yuzo Iano.
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Filtros adaptativos. 2. Algoritmos. 3. Processo estocástico. 4. Processamento de sinais – Técnicas digitais. I. Iano, Yuzo. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

RESUMO

Este trabalho tem como objetivo estudar a inclusão dos processos de decomposição utilizados na geração da Transformada Discreta Seno com Pré-rotação de Eixos, em algoritmos adaptativos no domínio transformado e baseados no (LMS-"*Least Mean Square*"). Nesse sentido, o primeiro passo é inserir o processo de decomposição no problema da filtragem de Wiener, considerando-se que o filtro é FIR. A partir dessa modificação e das médias estatísticas que envolviam esses sinais é obtida a função custo J. A minimização dessa função conduz a um sistema de equações lineares similares as de Wiener-Hopf. Foi empregada a filtragem de Wiener tradicional e a modificada num esquema de predição linear. Simulações são implementadas a fim de se comparar o comportamento da curva relativa ao erro quadrático médio versus números de amostras, considerando-se que as seqüências envolvidas pertençam a um processo AR(1).

Com a incorporação do processo de decomposição na estrutura de filtro adaptativo baseado no algoritmo LMS, se propôs dois esquemas de filtragem adaptativa denominados de D-LMS e DSTr-LMS. Curvas de aprendizagem são usadas na comparação entre as taxas de convergência desses algoritmos e as obtidas pelos NLMS, DST-LMS e o DCT-LMS, quando os mesmos são utilizados na predição linear de processos AR(2) e também de sinais eletromiográficos (EMG). De forma geral, constatou-se um bom desempenho dos algoritmos D-LMS e DSTr-LMS. No caso particular da predição linear de sinais EMG, o DSTr-LMS atingiu uma taxa de convergência tão boa quanto o DST-LMS, porém com um erro de regime permanente inferior.

ABSTRACT

The purpose of this thesis is to study the inclusion of the decomposition process, used in the generation of the Discrete Sine Transform with Pre-rotation Axis, into adaptive algorithms in the transformed domain and based on Least Mean Square (LMS). In this sense, the first step is to insert the decomposition process into Wiener's filtering problem, taking into consideration that the filter is FIR. From this modification and the statistical averages of to the decomposed signals the J cost function is obtained. The minimization of this function leads to a linear equation system similar to Wiener-Hopf's. Wiener's traditional and modified filtering was used in a linear prediction scheme. Simulations are implemented in order to compare the behavior of the curve relative to the mean square error versus sample numbers, considering that the sequences involved belong to an AR(1) process.

With the incorporation of the decomposition process in the adaptive filter structure based on the LMS algorithm, two adaptive filtering schemes called D-LMS and DSTr-LMS were proposed. Learning curves are used to compare the convergence rates between these algorithms and NLMS, DST-LMS and DCT-LMS when they are all used in the linear prediction of AR(2) processes and also eletromiographic signals (EMG). In a general way, D-LMS and DSTr-LMS had a good performance. In the specific case of EMG signals linear prediction, DSTr-LMS reached a convergence rate as good as DST-LMS, but with an inferior steady state error.

AGRADECIMENTOS

Ao meu bom Deus: Pai, Filho e Espírito Santo, à Quem sempre recorri e a sua presença senti!

À minha amada esposa Jânia e filhos Eduardo e Mateus pela paciência, encorajamento e amor dedicados à mim em todos os momentos.

Aos meus amados pais Eduardo e Deyze pelas orações que seguramente alcançaram a atenção do meu Senhor.

À CAPES pela bolsa concedida.

Ao **Prof. Yuzo Iano** pelo estímulo e pela confiança devotados a mim.

Para a UNICAMP o meu reconhecimento pelos benefícios obtidos.

Muito especial à Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia pelo apoio sempre presente.

Agradeço aos professores membros da banca examinadora pela participação.

CONTEÚDO

LISTA DE FIGURAS

4.9 Variação de parâmetros médios relacionados ao filtro D. O sinal desejado é obtidos de um processo AR(1), onde a1 = -0.99. Os valores médios são calculados considerando que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \leq i \leq 401$ e o valor de NF igual a 20.

a)Variação de: (1) $\overline{ww_0}$ *versus i* ; (2) $\overline{ww_1}$ *versus i* ; (3) $\overline{w_1}$ *versus i*

b)Gráficos relativos a variação dos valores médios: (1) $\overline{\hat{\sigma}_{\upsilon}^{2}}$ versus i (2) $\overline{\hat{\sigma}_{e^f}^{2}}$ versus i.

c)Variação de $\overline{er}$ *versus i* .

4.10 Variação de parâmetros médios relacionados ao filtro D. O sinal desejado é obtidos de um processo AR(1), onde a1 = -0.2. Os valores médios são calculados considerando que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \leq i \leq 401$ e o valor de NF igual a 20.

a)Variação de: (1) $\overline{ww_0}$ *versus i* ; (2) $\overline{ww_2}$ *versus i* ; (3) $\overline{w_1}$ *versus i*

b)Gráficos relativos a variação dos valores médios: (1) $\overline{\hat{\sigma}_{\upsilon}^{2}}$ versus i (2) $\overline{\hat{\sigma}_{e^f}^{2}}$ versus i .

c)Variação de $\overline{er}$ *versus i* .

4.11 Variação de parâmetros médios relacionados ao filtro D. O sinal desejado é obtidos de um processo AR(1), onde a1 = -0.2. Os valores médios são calculados considerando que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \leq i \leq 401$ e o valor de NF igual a 20.

a)Variação de: (1) $\overline{ww_1}$ *versus i* ; (2) $\overline{ww_2}$ *versus i* ; (3) $\overline{w_1}$ *versus i*

b)Gráficos relativos a variação dos valores médios: (1) $\overline{\hat{\sigma}_{\upsilon}^{2}}$ versus i (2) $\overline{\hat{\sigma}_{e^f}^{2}}$ versus i.

c)Variação de $\overline{er}$ *versus i* .

4.12 Variação de parâmetros médios relacionados ao filtro D. O sinal desejado é obtidos de um processo AR(1), onde a1 = -0.2. Os valores médios são calculados considerando que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \leq i \leq 401$ e o valor de NF igual a 20.

a)Variação de: (1) $\overline{ww_0}$ *versus i* ; (2) $\overline{ww_1}$ *versus i* ; (3) $\overline{w_1}$ *versus i*

LISTA DE TABELAS

LISTA DE ABREVIATURAS

DTLTI	*Discrete-time Linear Time-invariant System.*
T	Período de amostragem.
Z	*Z Transform.*
ROC	*Region of convergence*.
FIR	*Finite Impulse Response*.
IIR	*Infinite Impulse Response.*
J	*Cost Function or Performance Index Function.*
MSE	*Mean Square Error.*
LS	*Least Squares.*
LMS	*Least Mean Squares*.
DFT	*Discrete Fourier Transform.*
DFT-LMS	*Discrete Fourier Transform Least Mean Square adaptive filter.*
TDAF	*Transform Domain Adaptive Filter.*
DCT	*Discrete Cosine Transform.*
DST	*Discrete Sine Transform.*
WHT	*Walsh-Hadamard Transform*.
DHT	*Discrete Hartley Transform.*
PO2	*Power of Two Transform.*
DCT-LMS	*Discrete Cosine Transform Least Mean Square Algorithm*.
DSTr	*Discrete Sine Transform with Axis Rotation.*
D	Sistema de equações D, referindo-se a decomposição.
GD	*Descent Gradient Algorithm.*
SD	*Steepest Descent Algorithm.*
NLMS	*Normalized Least Mean Square Algorithm.*
DST-LMS	*Discrete Sine Transform Least Mean Square Algorithm*.
D-LMS	Algoritmo LMS com decomposição do sinal de entrada.
DSTr-LMS	Algoritmo LMS associado a Transformada DSTr.

AR(N)	Processos Autoregressivos de N$^{\underline{a}}$ ordem.
EMG	Sinais eletromiográficos.
Dr_i-LMS	Algoritmo D-LMS que exclui a entrada decomposta $r_i(n)$.
$Dr_{i\,j}$-LMS	Algoritmo D-LMS que exclui as entradas decompostas $r_i(n)$ e $r_j(n)$.
$DSTr_i$-LMS	Algoritmo DSTr-LMS que exclui a entrada decomposta $r_{DST,\,i}(n)$.
$DSTr_{i\,j}$-LMS	Algoritmo DSTr-LMS que exclui as entradas decompostas $r_{DST,\,i}(n)$ e $r_{DST,\,j}(n)$.

LISTA DOS PRINCIPAIS SÍMBOLOS

x(n), y(n), d(n), g(n), etc.	Podem representar seqüências determinísticas ou seqüências que representam funções amostra de processos aleatórios estacionários.
{x(n)}, {d(n)}, etc.	Processos aleatórios estacionários.
Z[.]	Operador Transformada Z direta.
Z^{-1}[.]	Operador Transformada Z inversa.
*	Complexo conjugado.
*	Convolução linear discreta.
$\{a_k\}$, $\{b_k\}$, $\{w_i\}$, etc.	Conjunto de coeficientes de um filtro digital.
T	Período de amostragem.
E[.] ou **E**(.)	Operador Média estatística.
J	Denota a função custo, ou função índice de desempenho.
∇	Vetor coluna operador gradiente complexo.
∇_k	O k-ésimo elemento do vetor coluna gradiente complexo.
d(n)	Sinal desejado.
$\hat{d}(n)$	Estimativa do sinal desejado.
y(n)	No esquema de filtragem de Wiener dado pela Fig.(2.2) este símbolo denota também a estimativa do sinal desejado.
$y_o(n)$	Sinal na saída do filtro ótimo (Wiener).
$\hat{d}_o(n)$	É o próprio sinal de saída $y_o(n)$, também denominado de sinal estimado.
$e_o(n)$	Erro ótimo ou erro de estimação relacionado ao filtro ótimo.
σ_d^2	Variância do sinal d(n).
$\sigma_{e_0}^2$	Variância do sinal e_o(n).
$\sigma_{y_0}^2$	Variância do sinal $y_o(n)$.

J_{min}	Mínima variância do erro quadrático médio, ou seja, a variância do sinal $e_o(n)$.
w_{oi}	i-ésimo coeficiente do filtro ótimo.
$r_x(i-k)$	Função autocorrelação do sinal da entrada do filtro de Wiener.
$r_{xd}(i-k)$	Função correlação cruzada entre o sinal desejado e o sinal estimado.
$\underline{\mathbf{w}}$	Denota o vetor coluna constituído pelos coeficientes de um filtro.
w_k	k-ésimo coeficiente de um filtro.
$J(\underline{\mathbf{w}})$	Função custo expressa de forma a enfatizar a sua dependência com o vetor coluna de coeficientes do filtro.
$\underline{\mathbf{x}}(n)$	Vetor coluna constituído das N amostras mais recentes do sinal x(n).
^{T}	Transposta de uma matriz.
$\underline{\mathbf{r}}_{dx}$	Vetor coluna constituído das correlações cruzadas entre a resposta desejada d(n) e os sinais de entrada presentes nas derivações do filtro.
$\underline{\mathbf{R}}_x$	Matriz NxN cujos elementos são as funções de autocorrelação do processo estocástico $\{x(n)\}$.
$\underline{\mathbf{w}}_o$	Vetor coluna constituído pelos coeficientes do filtro de Wiener.
$\underline{\mathbf{R}}_x^{-1}$	Inversa da matriz $\underline{\mathbf{R}}_x$.
$\underline{\mathbf{v}}$	É o vetor coluna que representa a diferença entre os vetores $\underline{\mathbf{w}}$ e $\underline{\mathbf{w}}_o$. Em termos geométricos esse vetor representa um novo sistema de coordenadas, originado por uma translação do sistema de coordenadas $\underline{\mathbf{w}}$.
v_i	i-ésimo elemento do vetor coluna $\underline{\mathbf{v}}$.
$\underline{\mathbf{\Lambda}}$	Matriz diagonal.
$\underline{\mathbf{Q}}$	Matriz ortonormal formada pelos vetores colunas $\underline{\mathbf{q}}_i$, também denominada de matriz de rotação.

$\underline{\mathbf{q}}_i$	Vetores coluna constituídos pelos autovalores da matriz de correlação $\underline{\mathbf{R}}_x$.
$\underline{\mathbf{v}}'$	Representa um novo sistema de coordenadas, originado por uma rotação do sistema de coordenadas $\underline{\mathbf{v}}$.
v_{i}'	i-ésimo elemento do vetor coluna $\underline{\mathbf{v}}'$.
$\underline{\mathbf{r}}(n)$ e $\underline{\mathbf{xr}}(n)$	Vetores originados pelo processo de pré-rotação de $\underline{\mathbf{x}}(n)$.
r_{ni}	i-ésimo elemento do vetor coluna $\underline{\mathbf{r}}(n)$.
$\mathrm{xr}_{\mathrm{ni}}$	i-ésimo elemento do vetor coluna $\underline{\mathbf{xr}}(n)$.
W(z)	Sistema W(z) utiliza as amostras do vetor $\underline{\mathbf{xr}}(n)$ a fim de fornecer o sinal em sua saída y(n).
w_i	i-ésimo coeficiente do sistema W(z).
WW(z)	Sistema WW(z) utiliza as amostras do vetor $\underline{\mathbf{r}}(n)$ a fim de fornecer o sinal em sua saída yy(n).
ww_i	i-ésimo coeficiente do sistema WW(z).
$\mathrm{xr}_{\mathrm{i}}(n)$	i-ésima seqüência formada pelo processo de decomposição associado ao vetor $\underline{\mathbf{xr}}(n)$.
$\mathrm{r}_{\mathrm{i}}(n)$	i-ésima seqüência formada pelo processo de decomposição associado ao vetor $\underline{\mathbf{r}}(n)$.
y(n)	Relacionado a Fig.(3.1) e representa o sinal na saída do sistema W(z).
yy(n)	Relacionado a Fig.(3.1) e representa o sinal na saída do sistema WW(z).
$y_c(n)$	É o sinal completo, ou seja, a soma de y(n) com yy(n).
$\underline{\mathbf{u}}(n)$	Vetor coluna formado pelos elementos dos vetores $\underline{\mathbf{xr}}(n)$ e $\underline{\mathbf{r}}(n)$.
$\underline{\boldsymbol{\theta}}$	Vetor coluna de comprimento 2N formado pelos coeficientes do sistema W(z) e WW(z), ou seja, w_{i} e ww_{i}.

$\underline{\mathbf{p}}$	Vetor coluna constituído pelas funções de correlação cruzada com deslocamentos nulos entre o sinal d(n) e as seqüências $xr_i(n)$.
$\underline{\mathbf{pp}}$	Vetor coluna constituído pelas funções de correlação cruzada com deslocamentos nulos entre o sinal d(n) e as seqüências $r_i(n)$.
$\underline{\mathbf{R}}_{xr}$	Matriz NxN constituída pelas funções de correlação cruzada com deslocamentos nulos entre as entradas decompostas $xr_i(n)$.
$\underline{\mathbf{R}}_{r}$	Matriz NxN constituída pelas funções de correlação cruzada com deslocamentos nulos entre as entradas decompostas $r_i(n)$.
$\underline{\mathbf{R}}_{xr\,r}$	Matriz NxN constituída pelas funções de correlação cruzada com deslocamentos nulos entre as entradas decompostas $xr_i(n)$ e $r_i(n)$.
$\underline{\mathbf{R}}_{r\,xr}$	Matriz NxN constituída pelas funções de correlação cruzada com deslocamentos nulos entre as entradas decompostas $r_i(n)$ e $xr_i(n)$.
$J(\underline{\mathbf{w}},\underline{\mathbf{ww}})$ ou $J(\underline{\theta})$	Função custo do sistema D.
$A_{AR}(z)$	Função de transferência do filtro que gera um processo AR(N).
a_i	Coeficientes do modelo autoregressivo.
$P^f(z)$	Filtro preditor progressivo de um passo a frente (filtro transversal).
$\underline{\mathbf{h}}^f$	Vetor coluna constituído dos coeficientes do preditor progressivo $P^f(z)$.
$e^f(n)$	"*Forward Error Prediction*".
$EP^f(z)$	Filtro erro de predição progressivo.
$\underline{\mathbf{ep}}^f$	Vetor coluna constituído pelos coeficientes do filtro $EP^f(z)$.
$\hat{R}_x(k)$	Estimador dos valores da função autocorrelação.
$h^f_{1_{ap}}$	Valor aproximado de $h^f_{1_{ap}}$.
σ_v^2	Variância do ruído.
$\sigma^2_{e^f}$	Variância do erro de estimação a frente ou progressivo.

$\hat{\sigma}^2_{e^f}$	Valor estimado de $\sigma^2_{e^f}$.
$\hat{\sigma}^2_v$	Valor estimado de σ^2_v
er	Erro relativo entre σ^2_v e $\sigma^2_{e^f}$.
NF	Constante igual a 20.
$\underline{\boldsymbol{H}^f_{1\,ap}}$	Matriz formada por NF por 401 valores de $h^f_{1_{ap}}$.
$\underline{\hat{\sigma}^2_\upsilon}$	Matriz formada por NF por 401 valores de $\hat{\sigma}^2_\upsilon$.
$\underline{\hat{\sigma}^2_{e^f}}$	Matriz formada por NF por 401 valores de $\hat{\sigma}^2_{e^f}$.
$\underline{\mathbf{ER}}$	Matriz formada por NF por 401 valores de er.
$\overline{h}^f_{1\,ap}$	Valor médio estimado de $h^f_{1\,ap}$.
$\overline{\hat{\sigma}^2_\upsilon}$	Valor médio estimado de $\hat{\sigma}^2_\upsilon$.
$\overline{\hat{\sigma}^2_{e^f}}$	Valor médio estimado de $\hat{\sigma}^2_{e^f}$.
$\overline{er}$	Valor médio estimado de er.
$\hat{\boldsymbol{R}}_{xy}(0)$	Estimador do valor da função correlação cruzada entre duas seqüências x(n) e y(n) para deslocamento nulo.
$\underline{\boldsymbol{WW}_0}$	Matriz formada por NF por 401 valores de ww_0.
$\underline{\boldsymbol{WW}_1}$	Matriz formada por NF por 401 valores de ww_1.
$\underline{\boldsymbol{WW}_2}$	Matriz formada por NF por 401 valores de ww_2.
$\underline{\boldsymbol{W}_1}$	Matriz formada por NF por 401 valores de w_1.
$\overline{ww_0}$	Valor médio estimado de ww_0.
$\overline{ww_1}$	Valor médio estimado de ww_1.
$\overline{ww_2}$	Valor médio estimado de ww_2.

$\overline{w_1}$	Valor médio estimado de w_1.
$\underline{\mathbf{w}}(n+1)$	Vetor peso cujos valores correspondem ao instante n+1.
$\nabla J(n)$	Vetor gradiente cujos valores correspondem ao instante n.
J(n)	Índice de desempenho variável com o tempo.
$\hat{\underline{\mathbf{R}}}(n)$	Estimativa da função autocorrelação $\underline{\mathbf{R}}$.
$\hat{\underline{\mathbf{p}}}(n)$	Estimativa da correlação cruzada $\underline{\mathbf{p}}$.
$\Vert\ \Vert$	Norma Euclidiana.
δ	Constante positiva próxima de zero.
μ	Constante de convergência de algoritmos adaptativos.
$\mu(n)$	Passo do degrau variável.
N	Número de derivações do filtro Transversal.
$\mathbb{T}$	Representa uma transformada ortogonal de N pontos.
$\underline{\mathbf{x}}_{\mathrm{T}}(n)$	Denota o vetor constituído das amostras de $\underline{\mathbf{x}}(n)$ transformadas por $\mathbb{T}$.
$\underline{\mathbf{w}}_{\mathrm{T}}(n)$	Vetor de pesos relacionado ao algoritmo no domínio da transformada, o DST-LMS ou DCT-LMS.
$\underline{\boldsymbol{\mu}}(n)$	Matriz diagonal de pesos.
$\hat{\sigma}^2_{x_{T,i}}(n)$	Estimativa de $\sigma^2_{x_{T,i}}(n) = \mathbf{E}[x^2_{T,i}(n)]$.
β	Constante utilizada em expressão recursiva para a estimação da variância de um sinal.
$\underline{\mathbf{w}}(n)$	Vetor pesos variável relacionado ao sistema W(z).
$w_i(n)$	i-ésimo coeficiente do vetor $\underline{\mathbf{w}}(n)$.
$\underline{\mathbf{ww}}(n)$	Vetor pesos variável relacionado ao sistema WW(z).
$ww_i(n)$	i-ésimo coeficiente do vetor $\underline{\mathbf{ww}}(n)$.
$\underline{\mathbf{xr}}_{\mathrm{DST}}(n)$	Vetor $\underline{\mathbf{xr}}$(n) transformado pela DST do tipo I.
$\underline{\mathbf{r}}_{\mathrm{DST}}(n)$	Vetor $\underline{\mathbf{r}}$(n) transformado pela DST do tipo I.

$S_N^I(m,n)$	Representa cada um dos elementos da matriz de transformação da DST-I.
$J(n)=\mathbf{E}[\|e(n)\|^2]$ versus n	Curva de aprendizagem.
$\hat{J}(n)$	Representa uma estimação da curva de aprendizagem.
I_1	Intervalo de amostras 1.
I_{1e}	Intervalo estendido de amostras 1.
I_2	Intervalo de amostras 2.
I_{1e}	Intervalo estendido de amostras 2.
I_3	Intervalo de amostras 3.
$\hat{J}(\infty)$	Estimação do erro quadrático médio de regime permanente.
$\hat{J}(I_1)$	Valores médios dos $\hat{J}(.)$ relativos a um intervalo específico.

C A P Í T U L O 1

INTRODUÇÃO

1.1 Conceitos Fundamentais

Como esta tese trata de sinais e sistemas discretos no tempo, é conveniente iniciar este trabalho introduzindo-se alguns conceitos fundamentais. Um sinal discreto no tempo é definido como sendo uma seqüência de números, {x(n)}, onde o índice “n” é um número inteiro que pode variar sobre uma faixa finita ou infinita de valores. Um sistema discreto no tempo é um procedimento numérico que transforma uma seqüência {x(n)} em uma outra seqüência {y(n)}, com características mais desejáveis. Essas características obtidas na seqüência de saída dependem do tipo de aplicação. Alguns exemplos de aplicação onde sistemas discretos se fazem presentes são citados a seguir:

a)Considere uma seqüência de entrada obtida através da amostragem de um sinal de voz captado por um microfone. Considerando ainda que este sinal esteja contaminado por um ruído de fundo ou uma interferência qualquer, é possível projetar um sistema discreto capaz de gerar uma seqüência de saída livre de ruído a partir do sinal de entrada, como por exemplo no cancelamento de eco e de ruído em telefonia ou em sistemas de videoconferência [1,2,3].

b)Pode-se utilizar sistemas discretos capazes de modelar um sinal eletromiográfico e dessa forma possibilitar um armazenamento eficiente dessas informações com o objetivo de controlar de forma mais natural uma prótese eletromecânica. Ainda na área médica, pode-se extrair características de sinais vitais (eletrocardiograma, eletroencefalograma, etc) para a identificação automática de anomalias ou simplesmente para monitoramento do paciente [4,5,6]. Em suma, a utilização de sistemas discretos na área médica é amplamente empregada permitindo-se uma melhor qualidade de vida aos pacientes que necessitam utilizar certos tipos de próteses, diagnósticos mais rápidos e seguros, monitoramento de sinais vitais, etc.

c)Com relação ao processamento digital do sinal de fala, segundo [7], pode-se ter as seguintes classes de aplicações que fazem uso de sistemas discretos:

- Transmissão e armazenamento digital.
- Sistemas de síntese da fala através de modelagem do sistema de produção da fala.
- Sistemas de identificação do locutor.

d)Em processamento de sinais de imagem ou vídeo, os sistemas discretos são utilizados para reduzirem a redundância desses sinais, e assim permitir uma diminuição no espaço de memória necessária para o armazenamento dessas informações ou na redução da taxa de transmissão desses dados. Um exemplo interessante é o uso da compactação de imagens referentes às impressões digitais de criminosos. O espaço utilizado para armazenamento dessas imagens foi reduzido drasticamente utilizando-se algoritmos de compactação, como por exemplo, a Transformada "*Wavelet*". Dessa forma, o tempo de análise e busca de um padrão de uma impressão digital pode ser reduzido e a catalogação e troca dessas informações agilizadas [8,9].

Um sistema linear discreto no tempo e invariável com o deslocamento (DTLTI– "*Discrete-time Linear Time-invariant System*"), cuja descrição é feita pela equação diferença linear a coeficientes constantes dada pela Eq.(1.1), é habitualmente denominado de filtro digital.

$$y(n) = \sum_{k=0}^{p} a_k x(n-k) - \sum_{k=1}^{q} b_k y(n-k) \tag{1.1}$$

onde,

n, k são números inteiros.

x(n) = sinal ou seqüência discreta de entrada do sistema. Caso x(n) tenha sido gerado a partir de uma amostragem de um sinal analógico x(t), então:

$$x(n) = \begin{cases} x(nT), & para \quad t = nT \\ não\ definido, & para \quad t \neq nT \end{cases}$$

onde,

T = período de amostragem.

y(n) = sinal ou seqüência de saída do sistema

p e q são inteiros que determinam a ordem do sistema.

$\{a_k\}$ e $\{b_k\}$ são dois conjuntos de coeficientes do filtro que definem o sistema.

A Eq.(1.1) informa que a seqüência de saída y(n) é obtida através de uma combinação linear de amostras passadas da seqüência de saída y(n-k) para k = 1...q, da amostra presente e de amostras passadas da seqüência de entrada x(n-k) para k = 0...p. Portanto, um filtro digital é um procedimento numérico que transforma uma seqüência de números em uma outra seqüência com características mais desejáveis. Essas características obtidas na seqüência de saída dependem da aplicação.

Uma outra forma de representar um DTLTI é defini-lo no domínio da Transformada Z (Z – "*Z Transform*"). Da mesma forma que a Transformada de Laplace pode ser utilizada na solução de uma equação diferencial linear a coeficientes constantes referente a um sistema analógico, a Transformada Z pode ser utilizada na solução da Eq.(1.1) referente a um sistema discreto. Isto é feito aplicando-se a Transformada Z na equação diferença linear que rege o sistema. A Transformada Z bilateral de uma seqüência discreta x(n) genérica é definida [10-15] por:

$$\begin{aligned} X(z) &= Z[x(n)] \\ X(z) &= \sum_{n=-\infty}^{+\infty} x(n) z^{-n} \end{aligned} \tag{1.2}$$

onde Z[.] denota a Transformada Z do argumento. Logo, a representação do DTLTI no domínio Z é obtida aplicando-se a Transformada Z em ambos os lados da Eq.(1.1),

$$Y(z) = X(z)\sum_{k=0}^{p} a_k z^{-k} - Y(z)\sum_{k=1}^{q} b_k z^{-k} \tag{1.3.a}$$

$$Y(z)\sum_{k=0}^{q} b_k z^{-k} = X(z)\sum_{k=0}^{p} a_k z^{-k} \tag{1.3.b}$$

$$\frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{p} a_k z^{-k}}{1 + \sum_{k=1}^{q} b_k z^{-k}} \tag{1.4.a}$$

ou

$$\frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{p} a_k z^{-k}}{\sum_{k=0}^{q} b_k z^{-k}} \tag{1.4.b}$$

Dessa forma, pode-se definir a função de sistema ou função de transferência desse filtro digital como sendo dada por:

$$H(z) = \frac{Y(z)}{X(z)} \tag{1.5}$$

A saída do filtro no domínio Z pode ser expressa através de H(z) como:

$$\mathrm{Y(z) = H(z).X(z)} \tag{1.6}$$

As Transformadas Z inversas de H(z) e de Y(z) são dadas, respectivamente, por:

$$h(n) = Z^{-1}[H(z)] \tag{1.7}$$

$$y(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k) = \sum_{k=-\infty}^{\infty} h(k)x(n-k) \tag{1.8.a}$$

$$y(n) = x(n) * h(n) = h(n) * x(n) \tag{1.8.b}$$

onde o símbolo “*” indica operador convolução linear. Portanto, a partir da Eqs.(1.8.a) e (1.8.b) observa-se que ao se adotar a seqüência de entrada x(n) como sendo igual a seqüência amostra unitária, a saída y(n) passa a ser chamada de resposta a amostra unitária ou resposta impulsiva h(n).

Na maior parte das aplicações, é desejável projetar sistemas que sejam causais e estáveis. A análise dessas características pode ser feita tanto através de h(n) quanto de H(z). Pode-se determinar se um DTLTI é causal e estável através do estudo da localização dos pólos referentes à função de transferência H(z). Para que um DTLTI seja causal e estável, é necessário que a região de convergência (ROC - "*Region of Convergence*") relativa a esse sistema situe-se na região externa ao maior pólo e inclua o círculo de raio unitário, ou seja, que todos os pólos de H(z) estejam situados na região interna do círculo de raio unitário [14,15].

Observando a Eq.(1.3.b) e considerando-se o caso especial onde q=0, tem-se:

$$Y(z) = X(z)\sum_{k=0}^{p} a_k z^{-k} \tag{1.9}$$

ou no domínio do tempo,

$$y(n) = \sum_{k=0}^{p} a_k x(n-k) \tag{1.10}$$

e o sistema passa a ser denominado de filtro com resposta impulsiva com extensão finita (FIR- "*Finite Impulse Response*"). Ainda com relação à Eq.(1.3.b), caso $q \neq 0$ o sistema passa a ser denominado de filtro com resposta impulsiva com extensão infinita (IIR - "*Infinite Impulse Response*").

1.2 Projetos de Filtros Digitais

Filtros digitais são dispositivos capazes de serem utilizados em uma vasta gama de aplicações. Por exemplo, para reduzir os efeitos de um ruído aditivo ou mesmo na eliminação de uma ou mais componentes espectrais indesejáveis contidas em um sinal. A partir da filtragem ter-se-á na saída desse filtro apenas a parte de interesse do sinal, isto é, o sinal livre de ruído ou das componentes espectrais indesejáveis. Tais situações ocorrem, por exemplo, quando se deseja eliminar a interferência de 60 Hz da linha, presente em um sinal eletrocardiográfico [5].

Filtro linear discreto no tempo é aquele em que a sua saída é uma função linear do sinal em sua entrada, tipicamente especificado através da somatória de convolução linear descrita pela Eq.(1.8). Para essa classe de filtros, têm-se basicamente dois procedimentos teóricos de projetos:

I)O procedimento clássico ou método clássico é aquele onde se propõe projetar um filtro com determinadas características de seletividade em freqüência, tal como o filtro passa baixas, filtro passa altas, filtro passa faixa ou um filtro rejeita faixa. Em situações onde se deseja reduzir o ruído contido em um sinal, necessita-se ter o conhecimento do conteúdo espectral tanto do sinal de interesse quanto do ruído. Nesses casos, a utilização de filtros digitais clássicos ocorre principalmente quando o sinal e o ruído ocupam nitidamente diferentes bandas de freqüências. A seguir são apresentados os métodos clássicos de projetos de filtros digitais [12,13,14,15]:

a)Projetos de filtros digitais IIR utilizando o método da invariância da resposta impulsiva.
b)Projetos de filtros digitais IIR utilizando o método da transformação bilinear.
c)Projetos de filtros digitais FIR utilizando o método das janelas.
d)Projetos de filtros digitais FIR utilizando o método da amostragem em freqüência.

II)Projetos de filtros ótimos são baseados na teoria da otimização e isso torna possível projetar um filtro, de forma a reduzir a componente ruidosa presente no sinal apesar do espectro do ruído e o espectro do sinal partilharem a mesma faixa de freqüências. O problema em questão é encontrar um conjunto de coeficientes que especifique o filtro, de forma que esses coeficientes sejam os melhores com relação a algum "critério pré-estabelecido". No lugar de utilizar-se a expressão "critério pré-estabelecido", costuma-se empregar os seguintes termos: "função objetiva" (J-"*Objective Function*"), "função custo" (J–"*Cost Function*") ou ainda "índice de desempenho" (J-"*Performance Index Function*"). Considere que $\{w_i\}$ seja a resposta impulsiva desse filtro e que y(n) seja a sua saída correspondente ao sinal de entrada x(n). A otimização do filtro é obtida através da minimização de uma função que depende do erro existente entre um sinal denominado de "sinal desejado" d(n) e o sinal de saída y(n), em outras palavras, é fazer uma estimação linear de d(n) a partir de uma outra seqüência, que neste exemplo é x(n). Definem-se dois tipos de filtro [16,17], dependendo do índice de desempenho utilizado:

a)Define-se Filtro de Wiener ao projeto de filtro otimizado através da minimização do valor quadrático médio do erro e(n). Nesse caso, a função custo é dada pelo erro quadrático médio (MSE-"*Mean Square Error*") que é uma função real e independente do tempo, cujos argumentos são os coeficientes $\{w_i\}$ do filtro, os quais podem assumir valores reais ou complexos.

b)Define-se Filtro dos Mínimos Quadrados (LS-"*Least Squares*") ao filtro otimizado através da minimização da soma dos quadrados dos erros. Costuma-se utilizar este critério de otimização quando as estatísticas dos sinais envolvidos não são conhecidas a priori.

1.3 Filtros Digitais Adaptativos: um breve histórico.

Filtros digitais adaptativos são dispositivos muito versáteis, sendo freqüentemente utilizados em uma vasta gama de aplicações que utilizam processamento digital de sinais. O algoritmo (LMS-"*Least-Mean-Square*") ou algoritmo gradiente estocástico [18,19] inventado por Widrow e Hoff em 1960 [20] é um dos algoritmos mais amplamente empregados para filtragem adaptativa devido a sua simplicidade, ou em outros termos, baixa complexidade computacional. Em 1975 esse algoritmo foi generalizado para a forma complexa, ou seja, o algoritmo LMS-Complexo [21], possibilitando-se operar com entradas e coeficientes complexos.

Em certas aplicações, tal como cancelamento de eco em teleconferência, a utilização de filtros adaptativos IIR [22] no domínio do tempo é uma opção normalmente considerada, no entanto, ela esbarra no problema da instabilidade associada aos filtros com resposta impulsiva infinita. Uma outra solução para esse problema seria a escolha de um filtro adaptativo com estrutura FIR. Isso impõe que esse filtro possua um número elevado de coeficientes, ou seja, uma longa resposta impulsiva, também objetivando neutralizar um eco de longa duração. Ocorre que o algoritmo LMS operando com um grande número de coeficientes conduz a um incremento em termos de complexidade computacional. Visando reduzir esse esforço computacional para os casos em que o filtro possua um número elevado de coeficientes, foi sugerida a inserção da Transformada de Fourier Discreta (DFT-"*Discrete Fourier Transform*") no algoritmo LMS complexo, denominando-se DFT-LMS (DFT-LMS - "*Discrete Fourier Transform Least Mean Square Algorithm*") [23] e a utilização da Transformada Rápida de Fourier (FFT-"*Fast Fourier Transform*") em algoritmos LMS com atualização dos pesos implementada bloco a bloco [24,25,26,27] e não amostra por amostra. A esses algoritmos que utilizam a DFT ou a FFT denominam-se de algoritmos adaptativos no domínio da transformada (TDAF-"T*ransform Domain Adaptive Filter*"). Esta denominação se aplica também a um outro grupo que utiliza transformadas ortogonais, tais como a própria DFT entre outras, com a finalidade de melhorar a taxa de convergência.

No algoritmo LMS [20,21] a taxa de convergência decresce à medida que aumenta o espalhamento dos autovalores da matriz de autocorrelação do sinal de entrada [28]. Muitas variações desse algoritmo foram propostas a fim de atingir uma taxa de convergência mais rápida. Em 1981 Narayan [29] foi o primeiro a introduzir a DFT no algoritmo LMS com a finalidade de melhorar a taxa de convergência. A argumentação é que essa melhoria poderia ser atingida através de um "branqueamento" do sinal de entrada antes de passar pelo filtro transversal. O processo utilizado para esse fim é composto por duas partes: a primeira distribui a potência do sinal em diversas faixas através da aplicação da DFT. Na segunda parte ocorre a normalização da potência referente a cada faixa durante o processo de atualização dos pesos do filtro transversal.

Em 1983, Narayan [30] estendeu esse conceito de banco de filtros utilizando tanto a DFT quanto a Transformada Discreta Cosseno (DCT-"*Discrete Cosine Transform*") em algoritmos adaptativos aplicados em processamento de voz. Em 1986 Lee [31] investigou o desempenho de algoritmos adaptativos que utilizavam a DFT, a DCT, a Transformada Discreta Seno (DST-"*Discrete Sine Transform*"), etc, com relação a dois importantes aspectos: taxa de convergência e erro de regime permanente, considerando-se que o fator de convergência utilizado na adaptação dos pesos fosse variável. Em 1989 Marshall [32] apresentou uma série de simulações relativas ao algoritmo LMS associado a diversas transformadas ortogonais, entre elas a Transformada *Walsh-Hadamard* (WHT- "*Walsh-Hadamard Transform*"), a Transformada Discreta Hartley (DHT-"*Discrete Hartley Transform* ") e a Transformada Potência de Dois (PO2-"*Power of Two Transform*"), visando analisar a taxa de convergência para cada caso quando submetidos a diversas classes de sinais de entrada.

Em 1992 Farhang-Boroujeny [33,34] mostrou que a eficiência das transformadas ortogonais em melhorar o desempenho do algoritmo LMS, depende da capacidade dessas transformadas em espalhar a energia do sinal de entrada nas diversas componentes transformadas. Em 1995 Beaufay [35] analisou o desempenho dos algoritmos DFT-LMS e DCT-LMS (DCT-LMS -"*Discrete Cosine Transform Least Mean Square Algorithm*") considerando processos de Markov de primeira ordem. Mais precisamente, ele mostrou que para esses processos com coeficientes de correlação $\rho \in [0,1]$ após a aplicação da DFT e da normalização, o espalhamento dos autovalores tende para $(1+\rho)/(1-\rho)$ à medida que cresce o comprimento do filtro. De forma análoga, aplicando-se a DCT, o espalhamento dos autovalores tende para $(1+\rho)$. Em 1996 Farhang-Boroujeny [36] propôs uma forma eficiente de implementação de diversas transformadas ortogonais aplicadas em filtros adaptativos no domínio transformado, direcionados para melhoria da taxa de convergência. Em 2000 Kim [37] propôs para o algoritmo no domínio transformado DCT-LMS um novo tipo de estimador de potência utilizado no cálculo do passo de degrau variável que atualiza o vetor peso. Também foi obtida uma série de expressões relativas ao desempenho desse algoritmo, baseadas num tratamento estatístico.

1.4 Organização desse Trabalho e Contribuições

Este trabalho é iniciado introduzindo-se no Capítulo 2 o conceito da filtragem de Wiener e expõe-se esse problema considerando-se as seguintes restrições:

-Filtro IIR não causal e causal;

-Filtro FIR causal.

Para esses casos são deduzidas [19,38,39,40] as equações de Wiener-Hopf (solução ótima), isto é, é encontrado o conjunto de coeficientes que minimiza a função custo J. Durante esse processo, surge o conceito do princípio da ortogonalidade e seu corolário, que definem quando o filtro está operando na sua condição ótima. O problema da filtragem de Wiener é representado na forma de somatórias e na forma matricial, considerando-se que o filtro possui uma estrutura FIR. Esse tratamento explicita a natureza quadrática da função custo e torna clara a obtenção, através das derivadas parciais de J, da equação de Wiener-Hopf. Esse capítulo é finalizado abordando as propriedades da função custo J e sua interpretação geométrica [18,19,39,40].

O Capítulo 3 fornece uma introdução da Transformada Seno Discreta com Pré-rotação de Eixos (DSTr –"*Discrete Sine Transform with Axis Rotation*") [41-44]. Inspirando-se nessa transformada, sugere-se uma reformulação do problema da filtragem de Wiener, através da incorporação de certas modificações provenientes dessa transformada. A idéia é utilizar o mesmo critério de desempenho, isto é, utilizar como função custo J o valor quadrático médio (MSE) do erro entre o sinal desejado $d(n)$ e o sinal de entrada $x(n)$. A modificação citada anteriormente diz respeito a separação ou decomposição do sinal x(n) em dois outros, conforme é feito na DSTr. O objetivo é basicamente igual ao problema da filtragem de Wiener, obter um conjunto de coeficientes para o filtro FIR de forma a minimizar a função custo J, o que significa fazer a melhor estimação do sinal d(n) em termos do critério MSE. A partir dessa modificação e da estrutura proposta, é deduzida uma expressão para J expressa em termos de médias estatísticas. A seguir, a minimização da função J expressa tanto na forma de somatórias quanto na forma matricial, conduz a um conjunto de equações lineares denominadas de "equações D" que servem de base para a proposta de dois algoritmos adaptativos a serem vistos no Capítulo 5.

Inicia-se o Capítulo 4 conceituando processos autoregressivos e sua relação com preditores progressivos lineares, cuja otimização resulta nas equações de Wiener-Hopf. É feita essa introdução a fim de se comparar os resultados de simulações, relativas a predição linear de processos AR(1), baseadas nessa teoria e os obtidos pelo "sistema de equações D". Utilizando-se como critério de avaliação a proximidade entre as curvas dadas por: variância estimada do ruído ν(n) gerador do processo AR(1) e a variância do erro de predição progressivo $e^{f}(n)$ relativa a essa função amostra de ruído, ambas em função do número de amostras empregado na estimação dos parâmetros utilizados nos respectivos sistemas de equações, ou seja, o sistema de Wiener e o sistema D. Conclui-se que essas curvas tendem a se sobrepor nos dois sistemas analisados, indicando que o erro de predição produzido pelos dois sistemas são equivalentes.

Inicia-se o Capítulo 5 expondo a solução iterativa das equações de Wiener-Hopf, utilizando o algoritmo gradiente descendente (GD-"*Descent Gradient*") ou da máxima descida (SD-"*Steepest Descent*") [18,19,39,40]. Em seguida, são apresentados os algoritmos adaptativos oriundos do "*Steepest Descent*", no domínio do tempo e no domínio da transformada, a saber:

LMS, NLMS (NLMS-“*Normalized Least Mean Square Algorithm*”), DCT-LMS, DST-LMS (DST-LMS - “*Discrete Sine Transform Least Mean Square Algorithm*”) [18,19,25,39,40]. Após isso, são apresentados dois métodos de filtragem adaptativa baseados no algoritmo LMS e na DSTr. O primeiro esquema de filtragem adaptativa insere o processo de decomposição utilizado na Transformada DSTr, na estrutura do algoritmo LMS normalizado, sendo denominado de algoritmo D-LMS. O segundo esquema é uma variante do primeiro, onde a Transformada DST é aplicada nos sinais decompostos antes de serem utilizados na atualização dos coeficientes do filtro transversal, sendo denominado de algoritmo DSTr-LMS [45,46].

O Capítulo 6 apresenta diversas simulações envolvendo predição linear de processos autoregressivos de 1ª ordem AR(1), onde os valores dos coeficientes obtidos pelo sistema de equações D são comparados com os coeficientes obtidos através do algoritmo D-LMS. Apresenta também, a predição linear de processos AR(2) [47,48] e de sinais eletromiográficos (EMG) [49,50], os resultados são analisados comparando-os com os obtidos pelos algoritmos NLMS, DST-LMS e DCT-LMS, onde se conclui que os algoritmos propostos podem atingir boas taxas de convergência e baixos valores para os erros de regime permanente [45,46,47,48,49,50]. Finalmente, no Capítulo 7 são apresentadas as conclusões gerais, enfatizando as vantagens obtidas pelos algoritmos adaptativos propostos.

C A P Í T U L O   2

FILTRAGEM DE WIENER

2.1 Introdução

Considere-se o modelo empregado na representação do problema da filtragem linear apresentado na Fig.(2.1). Admite-se que o sinal de entrada x(n) e o sinal desejado d(n) sejam seqüências que representam funções amostra de processos aleatórios estacionários, dados respectivamente por {x(n)} e {d(n)}. A partir da entrada x(n), o filtro especificado pela resposta impulsiva w_0, w_1, w_2, ... produz em sua saída a seqüência y(n) que é utilizada como uma estimativa de d(n). Define-se a seqüência erro estimado e(n) como sendo a diferença entre o sinal desejado d(n) e a seqüência de saída do filtro y(n).

Conforme mencionado no Capítulo 1, o procedimento estatístico para o problema de filtragem linear é otimizar o projeto do filtro com relação ao MSE. Considerando que os sinais sejam estacionários no sentido amplo e que as informações estatísticas de d(n) e x(n) coincidam com as informações estatísticas assumidas a priori no processo de otimização [16-19], então a solução ótima obtida é conhecida como Filtro de Wiener.

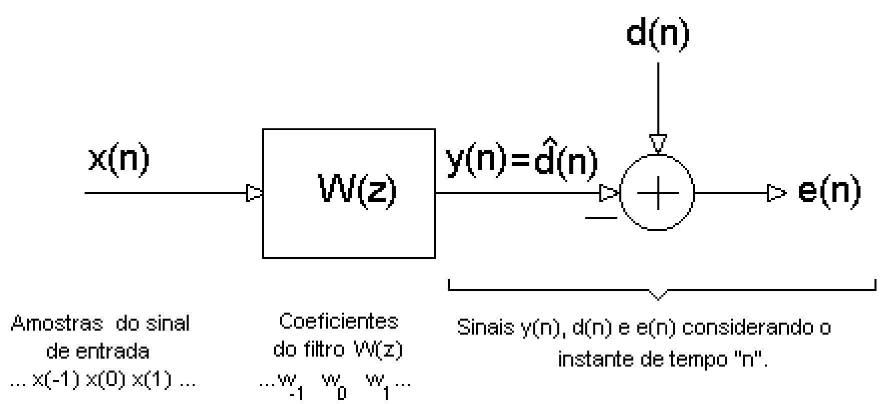


Fig.2.1 Modelo empregado na representação do problema da filtragem linear.

Caso os sinais exibam estatísticas discrepantes com relação às presumidas ou que esses sinais não sejam estacionários, então a solução obtida não é mais a ótima.

Além das seguintes imposições, filtro linear e discreto no tempo, MSE adotado como função custo, outras restrições influenciarão na solução ótima. Pode-se adotar diferentes estruturas para o filtro linear DTLTI considerando-se ou não a existência da causalidade. O filtro pode ser IIR e causal ou não causal, ou ainda, pode-se ter um filtro com a restrição de ser FIR e causal, restrições essas que conduzem a diferentes soluções. Em suma, para cada caso existirá uma solução ótima associada a cada tipo de filtro. A solução obtida considerando-se filtro FIR causal é uma das mais populares devido à simplicidade de se implementar uma solução adaptativa para a filtragem de Wiener utilizando-se essa estrutura. Diferentemente, a utilização de uma estrutura IIR para a solução adaptativa do filtro de Wiener sofrerá problemas de estabilidade existentes no próprio projeto do filtro, além da instabilidade gerada durante o processo de adaptação.

2.2 Desenvolvimento Matemático do Princípio da Ortogonalidade e das Equações de Wiener-Hopf

Nesta seção será apresentado um desenvolvimento matemático que conduzirá ao teorema da ortogonalidade ou princípio da ortogonalidade [19]. Esse teorema é importante no contexto de filtragem linear ótima, pois o mesmo acusa quando o filtro está operando na condição ótima, por conseguinte ele indicará quando os coeficientes do filtro assumirão os seus valores ótimos. Além disso, esse teorema ajudará na obtenção da equação de Wiener-Hopf cuja solução fornece os coeficientes ótimos do filtro, isto é, o filtro de Wiener.

As deduções serão apresentadas considerando-se que o filtro é linear e que as funções amostra $x(n)$ e $d(n)$, presentes na Fig.(2.1), pertencem a processos estocásticos conjuntamente estacionários no sentido amplo, ambas com média nula. Em um primeiro momento, será encontrada a solução ótima considerando-se filtro IIR não causal. Posteriormente, a solução ótima será encontrada sob a restrição do filtro ser IIR, porém causal. Por último, uma solução ótima será obtida considerando-se que o filtro é FIR causal.

2.2.1 Princípio da Ortogonalidade e Equações de Wiener-Hopf Para Filtro IIR Não Causal

Observando-se a Fig.(2.1), pode-se definir a saída através da somatória de convolução como sendo:

$$y(n) = \sum_{k=-\infty}^{+\infty} w_k^* x(n-k) \qquad \text{para todo n inteiro} \tag{2.1}$$

onde o "*" indica complexo conjugado, a fim de abranger a condição de que tanto o sinal de entrada quanto os coeficientes do filtro possam assumir valores complexos. O erro de estimação entre o sinal desejado e o sinal estimado é definido como sendo:

$$e(n) = d(n) - y(n) \tag{2.2.a}$$

$$e(n) = d(n) - \hat{d}(n) \tag{2.2.b}$$

O objetivo da filtragem de Wiener é determinar o conjunto de coeficientes $\{w_k\}$ do filtro que minimiza a função custo J definida como sendo:

$$J = \mathbf{E}[e(n).e^*(n)] \tag{2.3.a}$$

$$J = \mathbf{E}[|e(n)|^2] \tag{2.3.b}$$

onde "**E**" é o operador esperança ou média estatística.

Assumiu-se que tanto os valores do sinal de entrada quanto os valores dos coeficientes do filtro poderiam ser complexos. Assim, expressa-se o k-ésimo coeficiente do filtro na forma cartesiana como sendo:

$$w_k = a_k + jb_k \qquad \text{para todo k inteiro} \tag{2.4}$$

Portanto, J é uma função real e independente do tempo, de multivariáveis complexas que são os coeficientes $\{w_k\}$ e para enfatizar esse fato, essa função poderá também ser representada como J(**w**), onde **w** denota o vetor coluna de coeficientes do filtro. A forma de determinar os valores ótimos dos coeficientes $\{w_k\}$ para o qual a função real de multivariáveis complexas J passa por um valor mínimo, é utilizando-se o operador gradiente ∇. Logo, para minimizar J(**w**) define-se o operador gradiente complexo ∇ como sendo um vetor coluna complexo de infinitos elementos cujo k-ésimo elemento é formado pela derivada parcial de primeira ordem com relação a parte real a_k e a imaginária b_k do coeficiente w_k [19],

$$\nabla = [\ldots \quad \left(\frac{\partial}{\partial a_{k-1}} + j\frac{\partial}{\partial b_{k-1}}\right) \quad \left(\frac{\partial}{\partial a_k} + j\frac{\partial}{\partial b_k}\right) \quad \left(\frac{\partial}{\partial a_{k+1}} + j\frac{\partial}{\partial b_{k+1}}\right) \quad \left(\frac{\partial}{\partial a_{k+2}} + j\frac{\partial}{\partial b_{k+2}}\right) \quad \ldots]^T \tag{2.5.a}$$

onde o k-ésimo elemento desse vetor foi definido como sendo:

$$\nabla_k = \frac{\partial}{\partial a_k} + j\frac{\partial}{\partial b_k} \qquad \text{para todo k inteiro} \tag{2.5.b}$$

A aplicação do operador gradiente ∇ sobre a função custo J(**w**), gera um vetor gradiente multi-dimensional complexo, ∇J:

$$\nabla J = [\ldots \quad \left(\frac{\partial J}{\partial a_{k-1}} + j\frac{\partial J}{\partial b_{k-1}}\right) \quad \left(\frac{\partial J}{\partial a_k} + j\frac{\partial J}{\partial b_k}\right) \quad \left(\frac{\partial J}{\partial a_{k+1}} + j\frac{\partial J}{\partial b_{k+1}}\right) \quad \left(\frac{\partial J}{\partial a_{k+2}} + j\frac{\partial J}{\partial b_{k+2}}\right) \quad \ldots]^T \tag{2.6.a}$$

e cujo k-ésimo elemento desse vetor é dado por:

$$\nabla_k J = \frac{\partial J}{\partial a_k} + j\frac{\partial J}{\partial b_k} \qquad \text{para todo k inteiro} \tag{2.6.b}$$

Conforme [19,39,51], para que a função custo seja minimizada é necessário que cada uma das componentes do vetor ∇J seja igual a zero. Logo, tem-se:

$$\nabla_k J = 0 \tag{2.7}$$

ou utilizando-se as Eqs.(2.1), (2.2.a), (2.3.a) e (2.4) para expressar a Eq.(2.7) em uma outra forma:

$$\nabla_k \mathbf{E}\{[d(n) - \sum_{k=-\infty}^{\infty}(a_k - jb_k)x(n-k)].[d(n) - \sum_{k=-\infty}^{\infty}(a_k - jb_k)x(n-k)]^*\} = 0 \qquad \text{para } \forall \text{ n e k inteiros} \tag{2.8}$$

Substituindo-se a Eq.(2.3.a) na Eq.(2.6.b) tem-se que:

$$\nabla_k J(\underline{\mathbf{w}}) = \mathbf{E}\{\frac{\partial e(n)}{\partial a_k}e^*(n) + \frac{\partial e^*(n)}{\partial a_k}e(n) + j\frac{\partial e(n)}{\partial b_k}e^*(n) + j\frac{\partial e^*(n)}{\partial b_k}e(n)\} \tag{2.9}$$

e calculando-se as derivadas parciais da Eq.(2.9), chega-se a que:

$$\nabla_k J = -2\mathbf{E}[x(n-k)e^*(n)] \qquad \text{para todo k} \tag{2.10}$$

A partir deste momento, define-se e_o como sendo o valor do erro de estimação obtido quando o filtro opera na sua condição ótima, isto é, quando as Eqs.(2.7) ou (2.8) são satisfeitas. Então, tem-se:

$$\nabla_k J = -2\mathbf{E}[x(n-k)e_0^*(n)] = 0 \qquad \text{para todo k} \tag{2.11.a}$$

$$\mathbf{E}[x(n-k)e_0^*(n)] = 0 \qquad \text{para todo k} \tag{2.11.b}$$

A Eq.(2.11.b) é conhecida como “princípio da ortogonalidade” e ela estabelece que quando os coeficientes do filtro assumem os seu valores ótimos o erro entre a saída do filtro e o sinal desejado é mínimo. Para cada valor de “n”, o erro de estimação mínimo $e_o(n)$ é descorrelacionado (ortogonal) com todas as amostras x(n-k) presentes nas derivações do filtro de

Wiener. Pode-se ainda inferir a partir dessa equação que o $e_o(n)$ e x(n-k) devem ser realmente descorrelacionados, pois, caso contrário, o erro de estimação poderia ser reduzido ainda mais até a situação em que não houvesse mais relação entre esses dois sinais.

A Eq.(2.11.b) informa que a média de x(n-k).e(n) é nula quando o filtro opera na condição ótima. Nessa condição, os coeficientes do filtro são denominados de $\{w_{ok}\}$ e a saída do filtro de $y_o(n)$. Agora, examina-se o valor da média quando se considera a substituição de x(n-k), na Eq.(2.11.b), por y(n). Logo, tem-se:

$$\mathbf{E}[y_o(n)e_o^*(n)] = \mathbf{E}[\sum_{k=-\infty}^{+\infty} w_{ok}^* x(n-k)e_o^*]$$

$$\mathbf{E}[y_o(n)e_o^*(n)] = \sum_{k=-\infty}^{+\infty} w_{ok}^* \, \mathbf{E}[x(n-k)e_o^*] \tag{2.12}$$

e pelo princípio da ortogonalidade, $\mathbf{E}[x(n-k)e_o^*] = 0$, então:

$$\mathbf{E}[y_o(n)e_o^*(n)] = 0 \qquad \text{para todo n} \tag{2.13}$$

A Eq.(2.13) é conhecida como "corolário do princípio da ortogonalidade" e é uma conseqüência direta do princípio da ortogonalidade. Ela estabelece que quando o filtro opera na sua condição ótima, o sinal estimado $\hat{d}_o(n)$, que é a própria saída do filtro de Wiener $y_o(n)$, e o erro de estimação $e_o(n)$ são descorrelacionados. A justificativa é a mesma utilizada no princípio da ortogonalidade.

Na condição do filtro operando na condição ótima, define-se o valor mínimo de J como J_{min} ou $J_{min}(\underline{\mathbf{w}}_o)$ para enfatizar a dependência com os valores ótimos dos coeficientes do filtro.

$$J_{\mathbf{min}}(\underline{\mathbf{w}}) = J_{\mathbf{min}} = \mathbf{E}[e_o(n)e_o^*(n)] \tag{2.14}$$

Considerando-se que:

$$d(n) = e_o(n) + y_o(n) \tag{2.15}$$

Então o valor quadrático médio da Eq.(2.15) será dado por:

$$\mathbf{E}[d(n).d^*(n)] = \mathbf{E}\{[e_o(n) + y_o(n)][e_o(n) + y_o(n)]^*\}$$

$$\mathbf{E}[d(n).d^*(n)] = \mathbf{E}[e_o(n)e_o^*(n)] + \mathbf{E}[e_o(n)y_o^*(n)] + \mathbf{E}\, y_o(n)e_o^*(n)] + \mathbf{E}[y_o(n)y_o^*(n)]$$

$$\mathbf{E}[|d(n)|^2] = \mathbf{E}[|e_o(n)|^2] + \mathbf{E}[e_o(n)y_o^*(n)] + \mathbf{E}[y_o(n)e_o^*(n)] + \mathbf{E}[|y_o(n)|^2] \tag{2.16}$$

Aplicando-se o corolário do princípio da ortogonalidade na Eq.(2.16), tem-se:

$$\mathbf{E}[|d(n)|^2] = \mathbf{E}[|e_o(n)|^2] + \mathbf{E}[|y_o(n)|^2] \tag{2.17}$$

Como foi assumido que d(n), e(n) e y(n) possuíam médias nulas, então:

$$\sigma_d^2 \;= [|d(n)|^2] \qquad = \text{ variância de d(n)}$$

$$\sigma_{e_o}^2 = \mathbf{E}[|e_o(n)|^2] \qquad = \text{ variância de } e_o(n)$$

$$\sigma_{y_o}^2 = \mathbf{E}[|y_o(n)|^2] \qquad = \text{ variância de } y_o(n)$$

logo, tem-se:

$$\sigma_{\mathbf{d}}^2 = \sigma_{\mathbf{e_o}}^2 + \sigma_{\mathbf{y_o}}^2$$

$$\sigma_{\mathbf{d}}^2 = J_{\mathbf{min}} + \sigma_{\hat{\mathbf{d}}}^2 \tag{2.18.a}$$

$$J_{\mathbf{min}} = \sigma_{\mathbf{d}}^2 - \sigma_{\hat{\mathbf{d}}}^2 \tag{2.18.b}$$

Pode-se ainda, usando-se a Eq.(2.1), reescrever a Eq.(2.11.b) como se segue:

$$\mathbf{E}[x(n-k)e_0^{\,*}(n)] = 0 \qquad \text{para todo k}$$

$$\mathbf{E[}x(n-k)e_o^*\mathbf{(}n\mathbf{)]} = \mathbf{E}\{x(n-k)[d^*(n) - \sum_{i=-\infty}^{+\infty} w_{oi} x^* \mathbf{(}n-i)]\} \qquad \text{para todo k}$$

$$\mathbf{E[}x(n-k)e_o^*\mathbf{(}n\mathbf{)]} = \mathbf{E[}x(n-k)d^*(n)\mathbf{]} - \sum_{i=-\infty}^{+\infty} w_{oi}\, \mathbf{E[}x(n-k)x^*\mathbf{(}n-i)\mathbf{]} \qquad \text{para todo k}$$

$$\sum_{i=-\infty}^{+\infty} w_{oi}\, \mathbf{E[}x(n-k)x^*\mathbf{(}n-i)\mathbf{]} = \mathbf{E[}x(n-k)d^*(n)\mathbf{]} \qquad \text{para todo k} \tag{2.19}$$

As médias presentes na Eq.(2.19) possuem as seguintes interpretações [19,39]:

a) $\mathbf{E[}x(n-k)x^*\mathbf{(}n-i)\mathbf{]}$ é a função autocorrelação do sinal de entrada presente nas derivações do filtro, para um deslocamento igual a (i-k). Essa função pode também ser expressa por:

$$r_x\mathbf{(}i-k\mathbf{)} = \mathbf{E[}x(n-k)x^*\mathbf{(}n-i)\mathbf{]} \tag{2.20}$$

b) $\mathbf{E[}x(n-k)d^*(n)\mathbf{]}$ é a função correlação cruzada entre o sinal de entrada presente nas derivações do filtro e a resposta desejada, para um deslocamento igual a (-k). Essa função pode também ser expressa por:

$$r_{xd}\mathbf{(}-k\mathbf{)} = \mathbf{E[}x(n-k)d^*(n)\mathbf{]} \tag{2.21}$$

Utilizando-se as Eqs.(2.20) e (2.21) na Eq.(2.19), tem-se que:

$$\sum_{i=-\infty}^{+\infty} w_{oi} r_x\mathbf{(}i-k\mathbf{)} = r_{xd}\mathbf{(}-k\mathbf{)} \qquad \text{para todo k} \tag{2.22}$$

A Eq.(2.22) é denominada de equação de Wiener-Hopf para tempo discreto e ela define implicitamente a condição ótima de operação do filtro, através de um sistema infinito de equações lineares onde as infinitas incógnitas são os $\{w_{oi}\}$ [19,39,40].

2.2.2 Equações de Wiener-Hopf Para Filtro IIR Causal

Pode-se ter um outro resultado, considerado o filtro como sendo IIR e com a restrição de ser causal. Nesse caso, a resposta impulsiva é dada por:

$$w_n = 0, \qquad n < 0 \tag{2.23}$$

e a saída do filtro ou o sinal estimado é obtido pela somatória de convolução linear, isto é:

$$y(n) = \hat{d}(n) = w_n * x(n) = \sum_{k=0}^{\infty} w_k^* x(n-k) \qquad \text{para } n \geq 0 \tag{2.24}$$

Para se encontrar os coeficientes ótimos do filtro de Wiener deve-se minimizar a função custo e igualar a zero as componentes do vetor gradiente ∇J. De forma análoga ao caso de filtro IIR não causal, tem-se que a equação de Wiener-Hopf para o filtro de Wiener IIR causal [19, 39,40] é dada por:

$$\sum_{i=0}^{+\infty} w_{oi} r_x(i-k) = r_{xd}(-k) \qquad \text{para } k \geq 0 \tag{2.25}$$

2.2.3 Equações de Wiener-Hopf Para Filtro FIR Causal

Além dos dois casos citados anteriormente, existe ainda uma outra situação e consequentemente um novo resultado será gerado. A presente situação assume que o filtro de Wiener é um filtro FIR causal de extensão N, conforme apresentado na Fig.(2.2). Essa estrutura é freqüentemente utilizada na filtragem adaptativa, principalmente por não possuir problemas com relação à estabilidade devido ao fato do filtro utilizado ser FIR.

A resposta impulsiva do filtro transversal apresentado na Fig.(2.2) é definida através do vetor coluna $\underline{\mathbf{w}}$ de N coeficientes, expressa por:

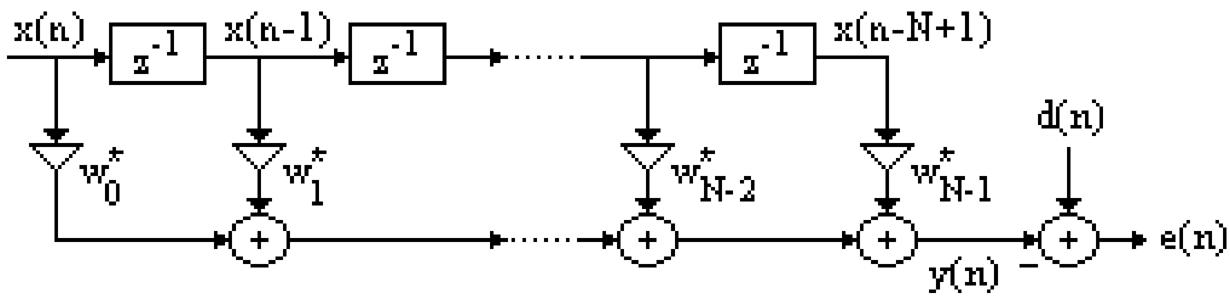


Fig.2.2 Filtro de Wiener FIR, também conhecido como filtro de Wiener transversal.

$$\underline{\mathbf{w}} = [w_0 \ w_1 \ \ w_2 \ \ \dots \ w_{N-1}]^T \tag{2.26}$$

A saída y(n) desse filtro, ou seja, o valor estimado $\hat{d}(n)$ é expresso por:

$$y(n) = \hat{d}(n) = w_n * x(n) = \sum_{k=0}^{N-1} w_k^* x(n-k) \qquad \text{para } n \geq 0 \tag{2.27}$$

Como nesta tese serão tratados apenas sinais com valores reais, considerar-se-á deste ponto em diante que os coeficientes do filtro de Wiener assumirão apenas valores reais. Então a Eq.(2.27) será reescrita sem a utilização do complexo conjugado sobre o vetor **w**, isto é:

$$y(n) = \hat{d}(n) = w_n * x(n) = \sum_{k=0}^{N-1} w_k x(n-k) \qquad \text{para } n \geq 0 \tag{2.28}$$

Ajustando-se a Eq.(2.22) para o caso presente de filtro FIR de N coeficientes, tem-se que a equação de Wiener-Hopf passa a representar um sistema linear de N equações simultâneas a N incógnitas [19,39,40], dadas por:

$$\sum_{i=0}^{N-1} w_{oi} r_x(i-k) = r_{xd}(-k) \qquad \text{para } 0 \leq k \leq N-1 \tag{2.29}$$

onde w_{oi} são os valores ótimos dos coeficientes desse filtro, que podem ser determinados, resolvendo-se o sistema de equações.

2.3 Formulação das Equações de Wiener-Hopf na Forma Matricial considerando-se Filtro FIR

Pode-se representar o problema da filtragem de Wiener na forma matricial. Essa é uma forma útil de enfocar esse problema, devido ao fato de se poder representar as equações da filtragem de Wiener de modo conciso. Inicia-se esse tratamento reescrevendo-se as equações na forma vetorial.

Seja o seguinte vetor coluna:

$$\underline{\mathbf{w}} = [w_0 \ \ w_1 \ \ w_2 \ \ .... \ \ w_{N-1}]^T \tag{2.30}$$

que representa os coeficientes do filtro transversal. Além disso, tem-se que:

$$\underline{\mathbf{x}}(n) = [x(n) \ \ x(n-1) \ \ x(n-2) \ \ .... \ \ x(n-N+1)]^T \tag{2.31}$$

é o vetor coluna constituído das N amostras mais recentes do sinal de entrada. Então, a saída do filtro y(n) será dada por:

$$y(n) = \underline{\mathbf{w}}^T \underline{\mathbf{x}}(n) \tag{2.32}$$

Logo, o erro entre a resposta desejada e o sinal estimado será:

$$e(n) = d(n) - \underline{\mathbf{w}}^T \underline{\mathbf{x}}(n) \tag{2.33}$$

Para sinais reais, define-se a função custo J como sendo:

$$J = \mathbf{E}[e^2(n)] \tag{2.34}$$

Substituindo-se a Eq.(2.33) na Eq.(2.34), tem-se que:

$$J = \mathbf{E}\{[d(n) - \underline{\mathbf{w}}^T \underline{\mathbf{x}}(n)]^2\}$$

$$J = \mathbf{E}[d^2(n) - 2d(n)\underline{\mathbf{w}}^T \underline{\mathbf{x}}(n) + \underline{\mathbf{w}}^T \underline{\mathbf{x}}(n)\underline{\mathbf{x}}^T(n)\underline{\mathbf{w}}]$$

$$J = \mathbf{E}[d^2(n)] - 2\mathbf{E}[\underline{\mathbf{x}}^T(n)\underline{\mathbf{w}}d(n)] + \mathbf{E}[\underline{\mathbf{w}}^T \underline{\mathbf{x}}(n)\underline{\mathbf{x}}^T(n)\underline{\mathbf{w}}]$$

$$J = \mathbf{E}[d^2(n)] - 2\underline{\mathbf{w}}^T \mathbf{E}[\underline{\mathbf{x}}(n)d(n)] + \underline{\mathbf{w}}^T E[\underline{\mathbf{x}}(n)\underline{\mathbf{x}}^T(n)]\underline{\mathbf{w}} \qquad (2.35)$$

Na Eq.(2.35) aparecem algumas médias estatísticas cujas definições e respectivas interpretações são apresentadas a seguir:

a) σ_d^2

Definindo-se:

$$\sigma_d^2 = \mathbf{E}[d^2(n)] \qquad (2.36)$$

onde σ_d^2 representa a variância da resposta desejada.

b) $\underline{\mathbf{r}}_{xd}$ e $\underline{\mathbf{r}}_{dx}$

Definindo-se:

$$\underline{\mathbf{r}}_{xd} = \mathbf{E}[\underline{\mathbf{x}}(n)d(n)] \qquad (2.37.a)$$

então, tem-se:

$$\underline{\mathbf{r}}_{xd} = \mathbf{E}[x(n)d(n) \;\; x(n-1)d(n) \;\; x(n-2)d(n) \;\; \ldots \;\; x(n-N+1)d(n)]^T$$

$$\underline{\mathbf{r}}_{xd} = [r_{xd}(0) \;\; r_{xd}(-1) \;\; r_{xd}(-2) \;\; \ldots \;\; r_{xd}(-N+1)]^T \qquad (2.37.b)$$

pois,

$$r_{xd}(-k) = \mathbf{E}[x(n-k)d(n)] \tag{2.37.c}$$

Uma outra opção é dada pela seguinte definição:

$$\underline{\mathbf{r}}_{dx} = \mathbf{E}[d(n)\underline{\mathbf{x}}(n)] \tag{2.38.a}$$

então, tem-se:

$$\underline{\mathbf{r}}_{dx} = \mathbf{E}[d(n)x(n)\ \ d(n)x(n-1)\ \ d(n)x(n-2)\ \ ...\ \ d(n)x(n-N+1)]^T$$

$$\underline{\mathbf{r}}_{dx} = [r_{dx}(0)\ \ r_{dx}(1)\ \ r_{dx}(2)\ \ ...\ \ r_{dx}(N-1)]^T \tag{2.38.b}$$

pois,

$$r_{dx}(k) = \mathbf{E}[d(n)x(n-k)] \tag{2.38.c}$$

Como,

$$\mathbf{E}[d(n)x(n-k)] = \mathbf{E}[x(n-k)d(n)] \tag{2.39.a}$$

então, tem-se:

$$r_{dx}(k) = r_{xd}(-k) \tag{2.39.b}$$

ou,

$$\underline{\mathbf{r}}_{dx} = \underline{\mathbf{r}}_{xd} \tag{2.39.c}$$

portanto, $\underline{\mathbf{r}}_{xd}$ é o vetor coluna constituído das correlações cruzadas entre os sinais de entrada presentes nas derivações do filtro e a resposta desejada d(n). Da mesma forma, $\underline{\mathbf{r}}_{dx}$ é o vetor coluna constituído das correlações cruzadas entre d(n) e os sinais presentes nas derivações do filtro.

c) $\underline{\mathbf{R}}_{\mathbf{x}}$

Definindo-se:

$$\underline{\mathbf{R}}_{\mathbf{x}} = \mathbf{E}[\underline{\mathbf{x}}(n)\underline{\mathbf{x}}^{T}(n)] \qquad (2.40.a)$$

e

$$r_x(i-k) = \mathbf{E}[x(n-k)x(n-i)] \qquad \text{para } i,k = 0,1,\cdots N-1 \qquad (2.40.b)$$

então, tem-se:

$$\underline{\mathbf{R}}_{\mathbf{x}} = \mathbf{E}\begin{bmatrix} x(0)x(0) & x(0)x(-1) & \dots & x(0)x(-N+1) \\ x(-1)x(0) & x(-1)x(-1) & \cdots & x(-1)x(-N+1) \\ \vdots & \vdots & \ddots & \vdots \\ x(-N+1)x(0) & x(-N+1)x(-1) & \cdots & x(-N+1)x(-N+1) \end{bmatrix}$$

$$\underline{\mathbf{R}}_{\mathbf{x}} = \begin{bmatrix} r_x(0) & r_x(1) & \dots & r_x(N-1) \\ r_x(-1) & r_x(0) & \cdots & r_x(N-2) \\ \vdots & \vdots & \ddots & \vdots \\ r_x(-N+1) & r_x(-N+2) & \cdots & r_x(0) \end{bmatrix} \qquad (2.41)$$

Como x(n) é um processo estocástico estacionário no sentido amplo, então, $r_x(k) = r_x(-k)$. Logo, tem-se:

$$\underline{\mathbf{R}}_x = \begin{bmatrix} r_x(0) & r_x(1) & \dots & r_x(N-1) \\ r_x(1) & r_x(0) & \cdots & r_x(N-2) \\ \vdots & \vdots & \ddots & \vdots \\ r_x(N-1) & r_x(N-2) & \cdots & r_x(0) \end{bmatrix} \tag{2.42}$$

onde $\underline{\mathbf{R}}_x = \mathbf{E}[\underline{\mathbf{x}}(n)\underline{\mathbf{x}}^T(n)]$ é uma matriz de dimensão NxN, cujos elementos são as funções de autocorrelações do processo estocástico **x**(n), com deslocamentos i-k variando de 0 até N-1.

Substituindo-se as Eqs.(2.36), (2.37.a) e (2.41) na Eq.(2.35), obtém-se:

$$J(\underline{\mathbf{w}}) = \sigma_d^2 - 2\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd} + \underline{\mathbf{w}}^T \underline{\mathbf{R}}_x \underline{\mathbf{w}} \tag{2.43}$$

A Eq.(2.43) apresenta de forma clara a dependência da função custo em relação aos coeficientes do filtro transversal.

Deve-se encontrar a condição de operação ótima do filtro, isto é, minimizar a função custo J($\underline{\mathbf{w}}$), Eq.(2.43), com relação aos coeficientes $\underline{\mathbf{w}}$. Portanto, impondo-se que:

$$\frac{\partial J}{\partial \underline{\mathbf{w}}} = 0 \tag{2.44}$$

onde a operação derivada de um escalar por um vetor $\underline{\mathbf{w}}$ é definida da seguinte forma:

$$\frac{\partial(.)}{\partial \underline{\mathbf{w}}} = \begin{bmatrix} \frac{\partial(.)}{\partial w_0} & \frac{\partial(.)}{\partial w_1} & \cdots & \frac{\partial(.)}{\partial w_{N-1}} \end{bmatrix}^T \tag{2.45}$$

Utilizando-se a Eq.(2.45) na Eq.(2.44), tem-se que:

$$\frac{\partial J}{\partial \underline{\mathbf{w}}} = \begin{bmatrix} \frac{\partial J}{\partial w_0} & \frac{\partial J}{\partial w_1} & \cdots & \frac{\partial J}{\partial w_{N-1}} \end{bmatrix}^T = \underline{0} \tag{2.46}$$

Como $\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd}$ e $\underline{\mathbf{w}}^T \underline{\mathbf{R}}_x \underline{\mathbf{w}}$ são escalares e σ_d^2 independe de $\underline{\mathbf{w}}$, então tem-se que:

$$\frac{\partial J}{\partial \underline{\mathbf{w}}} = -2\frac{\partial\left(\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd}\right)}{\partial \underline{\mathbf{w}}} + \frac{\partial\left(\underline{\mathbf{w}}^T \underline{\mathbf{R}}_x \underline{\mathbf{w}}\right)}{\partial \underline{\mathbf{w}}} = 0 \tag{2.47}$$

A Eq.(2.47) possui duas derivadas que serão calculadas a seguir:

a)Para se calcular a derivada

$$\frac{\partial\left(\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd}\right)}{\partial \underline{\mathbf{w}}} \tag{2.48}$$

deve-se utilizar a Eq.(2.45), resultando em:

$$\frac{\partial\left(\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd}\right)}{\partial \underline{\mathbf{w}}} = \left[\frac{\partial\left(\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd}\right)}{\partial w_0} \quad \frac{\partial\left(\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd}\right)}{\partial w_1} \quad \cdots \quad \frac{\partial\left(\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd}\right)}{\partial w_{N-1}}\right]^T \tag{2.49}$$

Além disso,

$$\frac{\partial\left(\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd}\right)}{\partial w_0} = \frac{\partial\left(w_0 r_{xd}(0) + w_1 r_{xd}(-1) + \ldots + w_{N-1} r_{xd}(-N+1)\right)}{\partial w_0}$$

$$\frac{\partial\left(\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd}\right)}{\partial w_0} = r_{xd}(0)$$

logo, basta estender esse procedimento para todos os coeficientes do filtro para se obter que:

$$\frac{\partial\left(\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd}\right)}{\partial \underline{\mathbf{w}}} = \left[r_{xd}(0) \quad r_{xd}(-1) \quad \cdots \quad r_{xd}(-N+1)\right]^T \tag{2.50.a}$$

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\underline{\mathbf{r}}_{\mathrm{xd}}\right)}{\partial\underline{\mathbf{w}}}=\underline{\mathbf{r}}_{\mathrm{xd}} \tag{2.50.b}$$

b) Para se calcular a derivada

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\underline{\mathbf{R}}_{\mathrm{x}}\underline{\mathbf{w}}\right)}{\partial\,\mathbf{w}} \tag{2.51}$$

deve-se utilizar a Eq.(2.40.a) de modo que:

$$\begin{gathered}\underline{\mathbf{w}}^{T}\underline{\mathbf{R}}_{x}\underline{\mathbf{w}}=\underline{\mathbf{w}}^{T}\,\mathbf{E}[\underline{\mathbf{x}}(n)\underline{\mathbf{x}}^{T}(n)]\underline{\mathbf{w}}\\ \underline{\mathbf{w}}^{T}\underline{\mathbf{R}}_{x}\underline{\mathbf{w}}=\mathbf{E}\{[\underline{\mathbf{w}}^{T}\,\underline{\mathbf{x}}(n)][\underline{\mathbf{x}}^{\mathrm{T}}(n)\;\underline{\mathbf{w}}]\}\end{gathered} \tag{2.52}$$

Logo, tem-se:

$$\frac{\partial\left(\underline{\mathbf{w}}^{T}\underline{\mathbf{R}}_{x}\underline{\mathbf{w}}\right)}{\partial\underline{\mathbf{w}}}=\frac{\partial\{\mathbf{E}[(\underline{\mathbf{w}}^{T}\,\underline{\mathbf{x}}(n))(\underline{\mathbf{x}}^{\mathrm{T}}(n)\;\underline{\mathbf{w}})]\}}{\partial\underline{\mathbf{w}}} \tag{2.53.a}$$

e invertendo a ordem entre os operadores esperança e diferenciação, por serem operações lineares, tem-se que:

$$\frac{\partial\left(\underline{\mathbf{w}}^{T}\underline{\mathbf{R}}_{x}\underline{\mathbf{w}}\right)}{\partial\underline{\mathbf{w}}}=\mathbf{E}\left\{\frac{\partial[(\underline{\mathbf{w}}^{T}\,\underline{\mathbf{x}}(n))(\underline{\mathbf{x}}^{\mathrm{T}}(n)\;\underline{\mathbf{w}})]}{\partial\underline{\mathbf{w}}}\right\} \tag{2.53.b}$$

Pode-se aplicar a regra da cadeia da diferenciação na Eq.(2.53.b), observando-se que $\underline{\mathbf{w}}^{T}\,\underline{\mathbf{x}}(n)$ e $\underline{\mathbf{x}}^{\mathrm{T}}(n)\;\underline{\mathbf{w}}$ são escalares. Então, tem-se:

$$\frac{\partial\left(\underline{\mathbf{w}}^T\underline{\mathbf{R}}_x\underline{\mathbf{w}}\right)}{\partial\underline{\mathbf{w}}}=\mathbf{E}\left\{\frac{\partial(\underline{\mathbf{w}}^T\underline{\mathbf{x}}(n))}{\partial\underline{\mathbf{w}}}(\underline{\mathbf{x}}^{\mathbf{T}}(n)\ \underline{\mathbf{w}})+(\underline{\mathbf{w}}^T\underline{\mathbf{x}}(n))\frac{\partial(\underline{\mathbf{x}}^{\mathbf{T}}(n)\ \underline{\mathbf{w}})}{\partial\underline{\mathbf{w}}}\right\}$$

Como $\underline{\mathbf{w}}^T\ \underline{\mathbf{x}}(n)=\underline{\mathbf{x}}^{\mathbf{T}}(n)\ \underline{\mathbf{w}}$, então, tem-se:

$$\frac{\partial\left(\underline{\mathbf{w}}^T\underline{\mathbf{R}}_x\underline{\mathbf{w}}\right)}{\partial\underline{\mathbf{w}}}=2\,\mathbf{E}\left\{\frac{\partial(\underline{\mathbf{w}}^T\underline{\mathbf{x}}(n))}{\partial\underline{\mathbf{w}}}(\underline{\mathbf{x}}^{\mathbf{T}}(n)\ \underline{\mathbf{w}})\right\}$$

Por analogia ao resultado da Eq.(2.50.b), tem-se que:

$$\frac{\partial\left(\underline{\mathbf{w}}^T\ \mathbf{R}_x\ \underline{\mathbf{w}}\right)}{\partial\underline{\mathbf{w}}}=2\,\mathbf{E}\left\{\underline{\mathbf{x}}(n)(\underline{\mathbf{x}}^{\mathbf{T}}(n)\ \underline{\mathbf{w}})\right\}$$

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\underline{\mathbf{R}}_{\mathrm{x}}\underline{\mathbf{w}}\right)}{\partial\underline{\mathbf{w}}}=2\underline{\mathbf{R}}_{\mathrm{x}}\underline{\mathbf{w}} \qquad (2.54)$$

Substituindo-se as Eqs.(2.50.b) e (2.54) na Eq.(2.47), obtém-se:

$$\frac{\partial J}{\partial\underline{\mathbf{w}}}=-2\underline{\mathbf{r}}_{xd}+2\underline{\mathbf{R}}_x\underline{\mathbf{w}}=\underline{\mathbf{0}} \qquad (2.55.a)$$

Como foi imposto que $\frac{\partial J}{\partial\underline{\mathbf{w}}}=\underline{\mathbf{0}}$, então o valor de $\underline{\mathbf{w}}$ é o ótimo, e passa-se a defini-lo como $\underline{\mathbf{w}}_{\mathrm{o}}$. Portanto,

$$\underline{\mathbf{R}}_{\mathrm{x}}\underline{\mathbf{w}}_{\mathrm{o}}=\underline{\mathbf{r}}_{\mathrm{xd}} \qquad (2.55.b)$$

cuja solução é dada por:

$$\underline{\mathbf{w}}_{\mathrm{o}}=\underline{\mathbf{R}}_{\mathrm{x}}^{-1}\underline{\mathbf{r}}_{\mathrm{xd}} \qquad (2.56)$$

Em outras palavras, $\underline{\mathbf{w}}_{\mathrm{o}}$ é o vetor dos coeficientes do filtro ótimo (filtro de Wiener).

2.4 Propriedades da Função Custo MSE e sua Interpretação Geométrica

Na secção anterior, foram calculados os coeficientes ótimos do filtro de Wiener, considerando-se a restrição do filtro ser FIR causal de extensão N. A função custo associada a esse caso pode ser expressa sob duas formas:

- Na forma matricial dada pela Eq.(2.43) e reescrita a seguir:

$$J(\underline{\mathbf{w}}) = \sigma_d^2 - 2\underline{\mathbf{w}}^T \underline{\mathbf{r}}_{xd} + \underline{\mathbf{w}}^T \underline{\mathbf{R}}_x \underline{\mathbf{w}} \tag{2.57}$$

- Ou em termos de somatórias. Para isso, basta substituir as Eqs. (2.30), (2.37.b), (2.37.c), (2.40.a) e (2.40.b) na Eq.(2.57) e expressar o resultado utilizando-se somatórias. Logo, tem-se:

$$J(\underline{w}) = \sigma_d^2 - 2\sum_{k=0}^{N-1} w_k r_{xd}(-k) + \sum_{k=0}^{N-1}\sum_{i=0}^{N-1} w_k w_i r_x(i-k) \tag{2.58}$$

O mínimo erro quadrático médio (MMSE) ou J_{Min} é obtido substituindo-se a Eq.(2.56) na Eq.(2.57). Logo, tem-se:

$$J_{Min} = J(\underline{\mathbf{w}}_o)$$

$$J_{Min} = \sigma_d^2 - 2(\underline{\mathbf{R}}_x^{-1}\underline{\mathbf{r}}_{xd})^T \underline{\mathbf{r}}_{xd} + (\underline{\mathbf{R}}_x^{-1}\underline{\mathbf{r}}_{xd})^T \underline{\mathbf{R}}_x (\underline{\mathbf{R}}_x^{-1}\underline{\mathbf{r}}_{xd})$$

$$J_{Min} = \sigma_d^2 - 2(\underline{\mathbf{R}}_x^{-1}\underline{\mathbf{r}}_{xd})^T \underline{\mathbf{r}}_{xd} + (\underline{\mathbf{R}}_x^{-1}\underline{\mathbf{r}}_{xd})^T \underline{\mathbf{r}}_{xd}$$

$$J_{Min} = \sigma_d^2 - (\underline{\mathbf{R}}_x^{-1}\underline{\mathbf{r}}_{xd})^T \underline{\mathbf{r}}_{xd} \tag{2.59.a}$$

$$J_{Min} = \sigma_d^2 - \underline{\mathbf{r}}_{xd}^T \underline{\mathbf{w}}_o \tag{2.59.b}$$

As incógnitas w_i presentes na Eq.(2.58) são apenas de primeiro e segundo graus. Conclui-se então que a MSE é uma função quadrática e unimodal [18,19], isto é, possui apenas um mínimo (global), não possuindo mínimo local. Isso garante que o problema da filtragem de Wiener com filtro FIR admite apenas uma única solução, $J_{Min} = J(\underline{\mathbf{w}}_o)$, propriedade essa que não se aplica para o caso de utilizar-se o filtro IIR.

As Eqs.(2.57) ou (2.58) por serem formas quadráticas representam uma hiperparábola em um espaço de dimensão N+1 cujos eixos coordenados são J, $w_0 \quad w_1 \quad \cdots \quad w_{N-1}$ e com ponto mínimo localizado em $\underline{\mathbf{w}} = \underline{\mathbf{w}}_o$. A MSE é também conhecida como superfície de desempenho do erro quadrático, devido a sua forma de taça. As Figs.(2.3.a) e (2.3.b) apresentam os gráficos da superfície de erro, considerando-se que as Eqs.(2.58) e (2.59.b) utilizaram os seguintes parâmetros:

a) Para a Fig.(2.3.a) os parâmetros foram:

$$\underline{\mathbf{R}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad ; \qquad \underline{\mathbf{r}}_{xd} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad ; \qquad \sigma_d^2 = 1$$

Portanto, $J(\underline{\mathbf{w}}) = w_0^2 + w_1^2 + 1$; $\underline{\mathbf{w}}_o^T = [w_0 \quad w_1] = [0 \quad 0]$ e $J(\underline{\mathbf{w}}_o) = J_{Min} = 1$

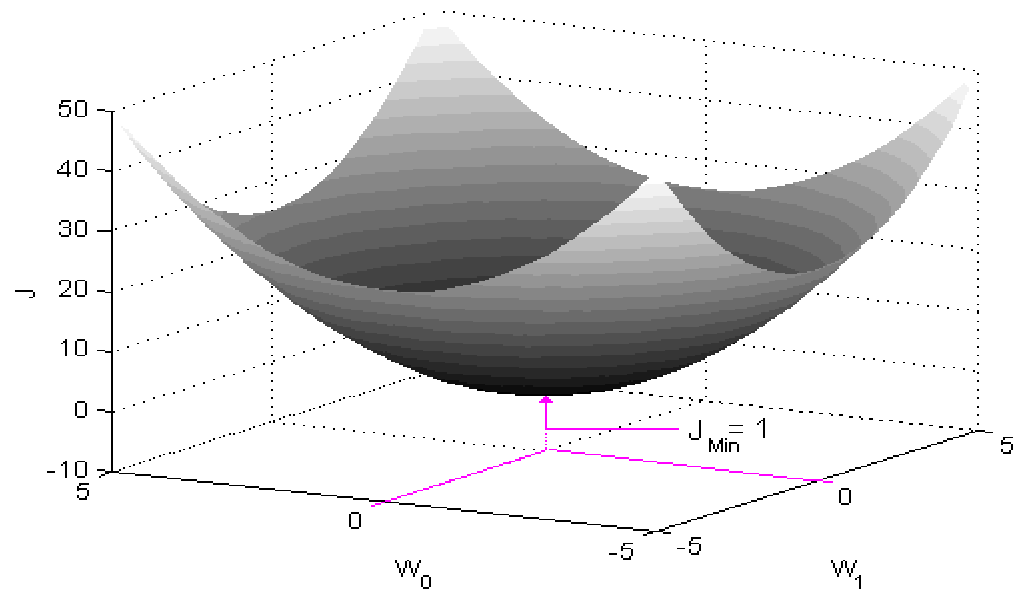
b) Para a Fig.(2.3.b) os parâmetros foram:

$$\underline{\mathbf{R}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad ; \qquad \underline{\mathbf{r}}_{xd} = \begin{bmatrix} 4 \\ 4 \end{bmatrix} \quad ; \qquad \sigma_d^2 = 32 \quad ;$$

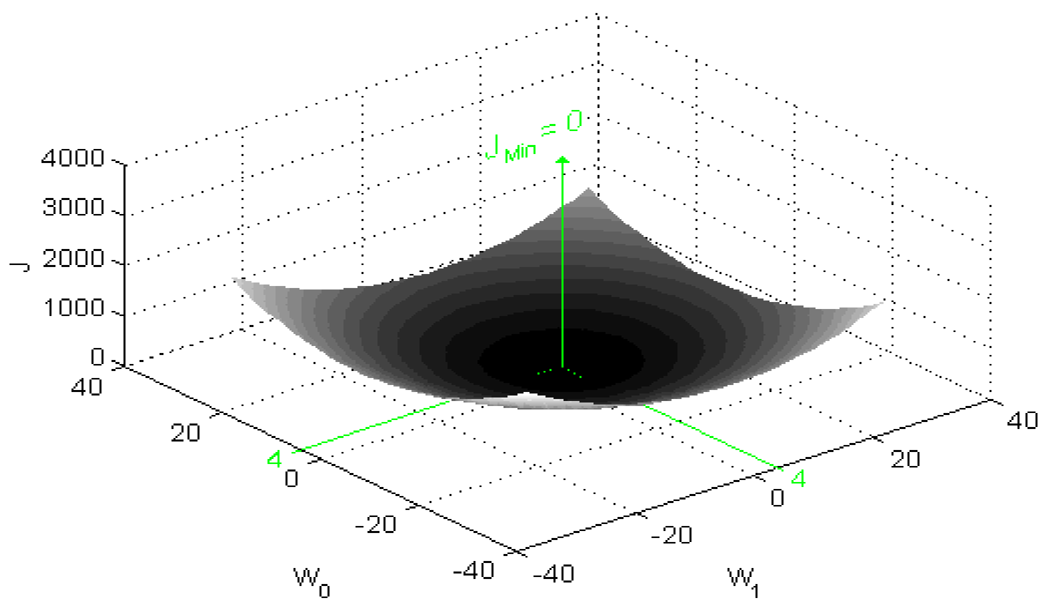
Portanto, $J(\underline{\mathbf{w}}) = w_0^2 + w_1^2 - 8w_0 - 8w_1 + 32$;

$$\underline{\mathbf{w}}_o^T = [w_0 \quad w_1] = [4 \quad 4] \quad \text{e} \quad J(\underline{\mathbf{w}}_o) = J_{Min} = 0$$

Uma equação freqüentemente utilizada e que explicita a função custo $J(\underline{\mathbf{w}})$ como sendo uma forma quadrática é obtida somando-se e subtraindo-se da Eq.(2.57) um fator igual a $\underline{\mathbf{r}}_{xd}^T \underline{\mathbf{w}}_o$



a)



b)

Fig.2.3 Superfície de desempenho do erro quadrático.

a)Superfície de erro considerando-se $\underline{\mathbf{R}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$; $\underline{\mathbf{r}}_{xd} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$; $\sigma_d^2 = 1$, portanto, $J_{Min} = 1$

b)Superfície de erro considerando-se $\underline{\mathbf{R}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$;$\underline{\mathbf{r}}_{xd} = \begin{bmatrix} 4 \\ 4 \end{bmatrix}$; $\sigma_d^2 = 32$, portanto, $J_{Min} = 0$.

conforme apresentado a seguir:

$$J(\underline{\mathbf{w}}) = J_{Min} + \underline{\mathbf{r}}_{xd}^{T}\underline{\mathbf{w}}_{o} - 2\underline{\mathbf{r}}_{xd}^{T}\underline{\mathbf{w}} + \underline{\mathbf{w}}^{\mathbf{T}}\underline{\mathbf{R}}\underline{\mathbf{w}}$$

$$J(\underline{\mathbf{w}}) = J_{Min} + \underline{\mathbf{r}}_{xd}^{T}\underline{\mathbf{w}}_{o} + \underline{\mathbf{w}}^{\mathbf{T}}\underline{\mathbf{R}}\underline{\mathbf{w}} - \underline{\mathbf{w}}^{T}\underline{\mathbf{r}}_{xd} - \underline{\mathbf{w}}^{T}\underline{\mathbf{r}}_{xd}$$

$$J(\underline{\mathbf{w}}) = J_{Min} + \underline{\mathbf{w}}^{\mathbf{T}}\underline{\mathbf{R}}\underline{\mathbf{w}} - \underline{\mathbf{w}}^{T}\underline{\mathbf{R}}\underline{\mathbf{R}}^{-1}\underline{\mathbf{r}}_{xd} - \underline{\mathbf{w}}^{T}\underline{\mathbf{R}}\underline{\mathbf{R}}^{-1}\underline{\mathbf{r}}_{xd} + \underline{\mathbf{w}}_{o}^{T}\underline{\mathbf{R}}\underline{\mathbf{R}}^{-1}\underline{\mathbf{r}}_{xd}$$

$$J(\underline{\mathbf{w}}) = J_{Min} + \underline{\mathbf{w}}^{\mathbf{T}}\underline{\mathbf{R}}\underline{\mathbf{w}} - \underline{\mathbf{w}}^{T}\underline{\mathbf{R}}\underline{\mathbf{w}}_{o} - \underline{\mathbf{w}}^{T}\underline{\mathbf{R}}\underline{\mathbf{w}}_{o} + \underline{\mathbf{w}}_{o}^{T}\underline{\mathbf{R}}\underline{\mathbf{w}}_{o}$$

$$J(\underline{\mathbf{w}}) = J_{Min} + \underline{\mathbf{w}}^{\mathbf{T}}\underline{\mathbf{R}}(\underline{\mathbf{w}} - \underline{\mathbf{w}}_{o}) - (\underline{\mathbf{w}}^{T} - \underline{\mathbf{w}}_{o}^{T})\underline{\mathbf{R}}\underline{\mathbf{w}}_{o}$$

$$J(\underline{\mathbf{w}}) = J_{Min} + \underline{\mathbf{w}}^{\mathbf{T}}\underline{\mathbf{R}}(\underline{\mathbf{w}} - \underline{\mathbf{w}}_{o}) - \underline{\mathbf{w}}_{o}^{T}\underline{\mathbf{R}}(\underline{\mathbf{w}} - \underline{\mathbf{w}}_{o})$$

$$J(\underline{\mathbf{w}}) = J_{Min} + (\underline{\mathbf{w}}^{\mathbf{T}} - \underline{\mathbf{w}}_{o}^{T})\underline{\mathbf{R}}(\underline{\mathbf{w}} - \underline{\mathbf{w}}_{o})$$

$$J(\underline{\mathbf{w}}) = J_{Min} + (\underline{\mathbf{w}} - \underline{\mathbf{w}}_{o})^{T}\underline{\mathbf{R}}(\underline{\mathbf{w}} - \underline{\mathbf{w}}_{o}) \qquad (2.60)$$

onde para sinais reais tem-se que $\underline{\mathbf{R}} = \underline{\mathbf{R}}^{T}$, é uma matriz Toeplitz simétrica real.

A Eq.(2.60) apresenta de forma clara que para $\underline{\mathbf{w}} = \underline{\mathbf{w}}_{o}$ a MSE assume o seu valor mínimo $J(\underline{\mathbf{w}}) = J_{Min}$.

Definindo-se a seguir o vetor $\underline{\mathbf{v}}$ como sendo a diferença entre o vetor $\underline{\mathbf{w}}$ e a solução de Wiener $\underline{\mathbf{w}}_{o}$, tem-se:

$$\underline{\mathbf{v}} = \underline{\mathbf{w}} - \underline{\mathbf{w}}_{\mathbf{o}} \qquad (2.61.a)$$

$$\underline{\mathbf{v}} = [v_0 \quad v_1 \quad \cdots \quad v_{N-1}]^{T} \qquad (2.61.b)$$

sendo essa diferença um desvio ou erro do vetor $\underline{\mathbf{w}}$ em relação a solução de Wiener $\underline{\mathbf{w}}_{o}$. Portanto, devido a esse desvio tem-se que um erro adicional é gerado em relação ao valor mínimo possível de ser atingido por J. Em termos geométricos esse desvio representa uma translação do sistema de coordenadas $\underline{\mathbf{w}}$ para um novo sistema $\underline{\mathbf{v}}$ cuja origem é $\underline{\mathbf{w}}_{o}$ [18,19,52,53]. Dessa forma, foi imposta uma translação de J para a origem desse novo sistema. Substituindo-se a Eq.(2.61.a) na Eq.(2.60), tem-se a forma quadrática expressa na forma matricial por:

$$J(\underline{\mathbf{w}}) = J_{Min} + \underline{\mathbf{v}}^{T}\underline{\mathbf{R}}\underline{\mathbf{v}} \tag{2.62.a}$$

Tem-se ainda :

$$J(\underline{\mathbf{w}}) = J_{Min} + \sum_{k=0}^{N-1}\sum_{i=0}^{N-1} R(i,k)v_k v_i = J_{Min} + \sum_{k=0}^{N-1}\sum_{i=0}^{N-1} r_x(i-k)v_k v_i \tag{2.62.b}$$

que está na forma de somatória, onde $R(i,k)$ são os elementos da matriz $\underline{\mathbf{R}}$.

Diferentemente do que ocorre na Eq.(2.58), pode-se notar na forma quadrática da Eq.(2.62.b) a ausência de variáveis v_k no primeiro grau, isto é, existem apenas variáveis no segundo grau v_k^2 e produtos cruzados $v_k v_i$ para $k \neq i$. Uma interpretação geométrica para isso, é que através de uma mudança de coordenadas, a translação existente na Eq.(2.58) foi suprimida com a eliminação desses termos de primeiro grau [18,52,53], que não existem mais na Eq.(2.62.b).

No entanto, pode-se simplificar ainda mais a Eq.(2.62.a), observando-se que a matriz de autocorrelação $\underline{\mathbf{R}}$ pode ser decomposta [18,19,52,53] em função de seus autovalores e autovetores, como apresentado a seguir:

$$\underline{\mathbf{R}} = \underline{\mathbf{Q}}\,\underline{\mathbf{\Lambda}}\,\underline{\mathbf{Q}}^{T} = \underline{\mathbf{Q}}\,\underline{\mathbf{\Lambda}}\,\underline{\mathbf{Q}}^{-1} \tag{2.63.a}$$

pois

$$\underline{\mathbf{Q}}^{T} = \underline{\mathbf{Q}}^{-1} \tag{2.63.b}$$

é unitária. De fato:

$$\underline{\mathbf{Q}} = [\underline{\mathbf{q}}_0 \quad \underline{\mathbf{q}}_1 \quad \cdots \quad \underline{\mathbf{q}}_{N-1}] \tag{2.63.c}$$

$$\underline{\mathbf{\Lambda}} = \begin{bmatrix} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_{N-1} \end{bmatrix} \tag{2.63.d}$$

onde $\underline{\mathbf{Q}}$ é a matriz ortonormal cujas colunas $\underline{\mathbf{q}}_i$ são os autovetores da matriz de correlação $\underline{\mathbf{R}}$ e $\underline{\mathbf{\Lambda}}$ é a matriz diagonal formada pelos N diferentes autovalores de $\underline{\mathbf{R}}$. Substituindo-se a Eq.(2.63.a) na Eq.(2.62.a) tem-se que:

$$J(\underline{\mathbf{w}}) = J_{Min} + \underline{\mathbf{v}}^T \underline{\mathbf{Q}} \underline{\Lambda} \underline{\mathbf{Q}}^T \underline{\mathbf{v}} = J_{Min} + (\underline{\mathbf{Q}}^T \underline{\mathbf{v}})^T \underline{\Lambda} \underline{\mathbf{Q}}^T \underline{\mathbf{v}} \tag{2.64}$$

Definindo-se $\underline{\mathbf{v}}^{'}$ como sendo o vetor coluna de pesos cujas componentes são os eixos de um novo sistema de coordenadas, tem-se:

$$\underline{\mathbf{v}}^{'} = \underline{\mathbf{Q}}^{-1} \underline{\mathbf{v}} = \underline{\mathbf{Q}}^T \underline{\mathbf{v}} \tag{2.65.a}$$

$$\text{logo,} \quad \underline{\mathbf{v}} = \underline{\mathbf{Q}} \underline{\mathbf{v}}^{'} \tag{2.65.b}$$

onde as componentes de $\underline{\mathbf{v}}$ e $\underline{\mathbf{v}}^{'}$ são correspondentemente fornecidas por:

$$\underline{\mathbf{v}} = [v_0 \quad v_1 \quad \cdots \quad v_{N-1}]^T \tag{2.65.c}$$

$$\underline{\mathbf{v}}^{'} = [v_0^{'} \quad v_1^{'} \quad \cdots \quad v_{N-1}^{'}]^T \tag{2.65.d}$$

Pela Eq.(2.65.b), $\underline{\mathbf{Q}}$ é a matriz de rotação que transforma os pontos no sistema de coordenadas $\underline{\mathbf{v}}^{'}$ para o sistema $\underline{\mathbf{v}}$. Tomando-se como referência os eixos coordenados dados por $\underline{\mathbf{v}}$, tem-se que as colunas da matriz $\underline{\mathbf{Q}}$ fornecem os eixos coordenados referentes ao sistema $\underline{\mathbf{v}}^{'}$.

Caso a referência adotada seja $\underline{\mathbf{v}}^{'}$, então, são as colunas da matriz $\underline{\mathbf{Q}}^{\mathrm{T}}$ na Eq.(2.65.a) que fornecem os eixos coordenados referentes ao sistema $\underline{\mathbf{v}}$. A Fig.(2.4) ilustra essa afirmação e nela foi adotado $\underline{\mathbf{v}}$ como coordenadas de referência. Tem-se: $\underline{\mathbf{Q}} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$, onde $\underline{\mathbf{Q}}$ é a matriz que fornece os eixos coordenados do novo sistema $\underline{\mathbf{v}}^{'}$, com rotação de –45° em relação a referência $\underline{\mathbf{v}}$. Reescrevendo-se a Eq.(2.64) em termos de $\underline{\mathbf{v}}^{'}$, tem-se que:

$$J(\underline{\mathbf{v}}^{'}) = \mathbf{v}^{'\mathrm{T}}\, \underline{\Lambda}\, \underline{\mathbf{v}}^{'} + J_{Min} \tag{2.66.a}$$

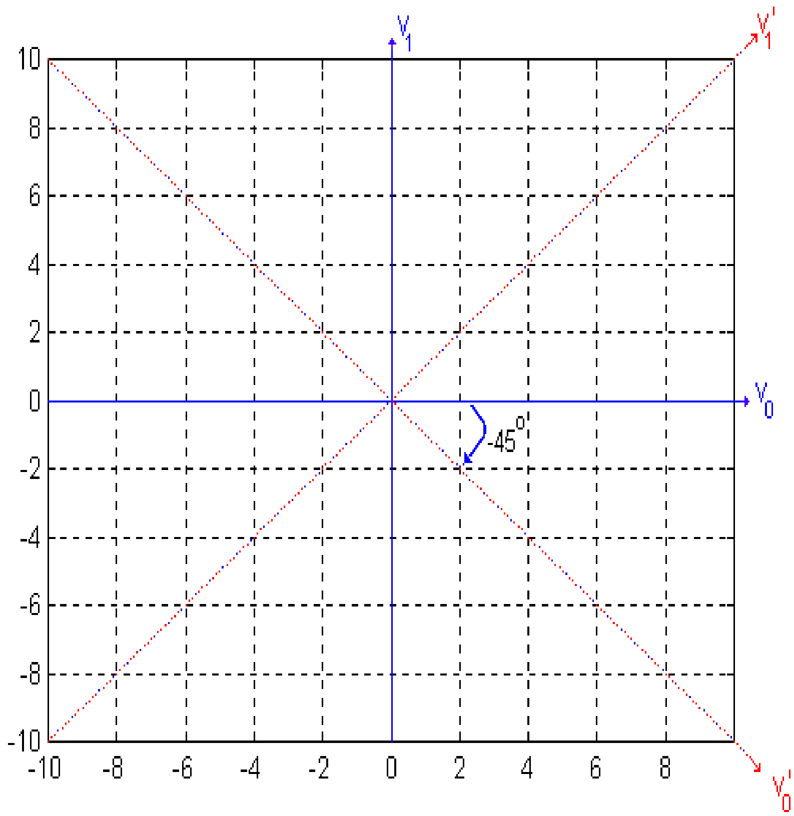


Fig.2.4. Eixos coordenados $\underline{\mathbf{v}}' = [\mathrm{v}_0^{'} \quad \mathrm{v}_1^{'}]$ com uma rotação de –45° em relação ao sistema $\underline{\mathbf{v}} = [\mathrm{v}_0 \quad \mathrm{v}_1]$, devido aos autovetores da matriz $\underline{\mathbf{Q}} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$.

A Eq.(2.66.a) é denominada forma canônica da Eq.(2.60). Essa forma quadrática real nas variáveis $v_0^{'} \quad v_1^{'} \quad \cdots \quad v_{N-1}^{'}$ é um polinômio do tipo:

$$J(\underline{\mathbf{v}}^{'}) = \sum_{k=0}^{N-1} \lambda_k (v_k^{'})^2 + J_{Min} \tag{2.66.b}$$

com coeficientes reais dados pelos autovalores λ_k.

Pode-se observar que a Eq.(2.66.b) difere da Eq.(2.58) por não possuir termos de primeiro grau e nem termos cruzados (mistos de 2º grau). Ela também difere da Eq.(2.62.b) por não possuir termos cruzados. Isto conduz à interpretação de que os eixos coordenados $v_k^{'}$ do vetor $\underline{\mathbf{v}}^{'}$ são os próprios eixos principais da superfície de desempenho do erro. Pela expressão $\underline{\mathbf{v}}^{'} = \underline{\mathbf{Q}}^{T}\underline{\mathbf{v}}$ Eq.(2.65.a), $\underline{\mathbf{Q}}^{T}$ é a matriz de rotação que aplicada ao vetor $\underline{\mathbf{v}}$ gera o vetor rotacionado $\underline{\mathbf{v}}^{'}$, então $\underline{\mathbf{Q}}$ é a matriz que define os eixos principais da superfície de desempenho do erro, como mostra a Fig.(2.4). Como $\underline{\mathbf{Q}}$ é formada pelos autovetores de $\underline{\mathbf{R}}$, então, esses são os que definem os eixos principais da superfície de erro.

A Fig.(2.5) apresenta uma série de curvas que permitem visualizar essas observações de uma outra forma. Na Fig.(2.5.a) existe o gráfico referente a Eq.(2.67.a), onde os seguintes parâmetros [18] foram utilizados para a sua obtenção :

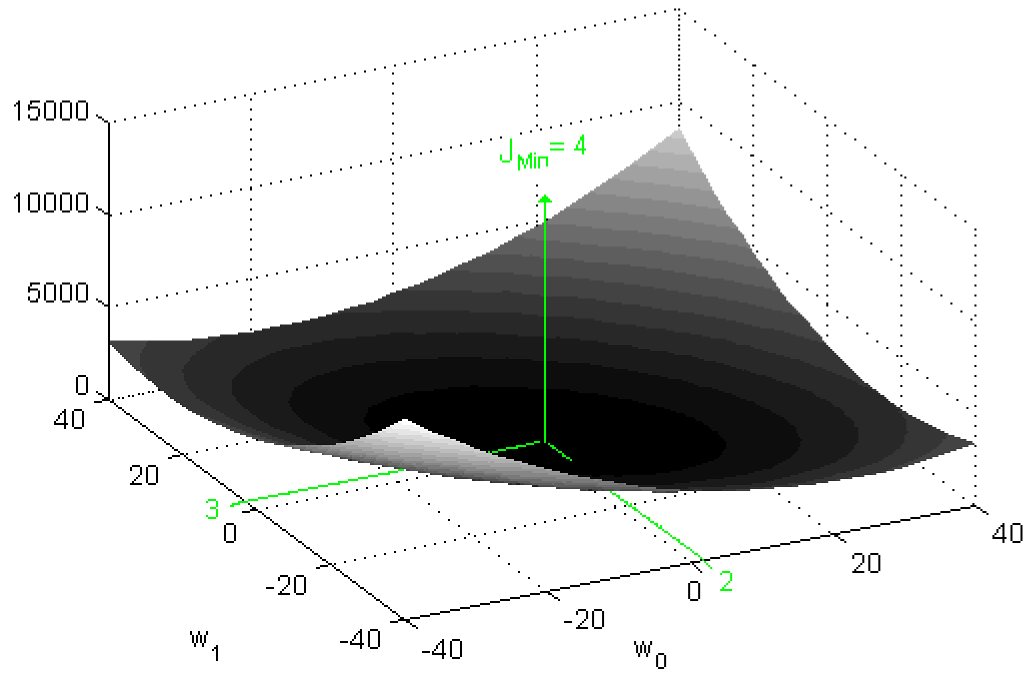
$$\underline{\mathbf{R}} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \quad ; \qquad \underline{\mathbf{r}}_{xd} = \begin{bmatrix} 7 \\ 8 \end{bmatrix} \quad ; \qquad \sigma_d^2 = 42 \quad ;$$

$$J(\underline{\mathbf{w}}) = 2w_0^2 + 2w_1^2 + 2w_0 w_1 - 14w_0 - 16w_1 + \sigma_d^2 \tag{2.67.a}$$

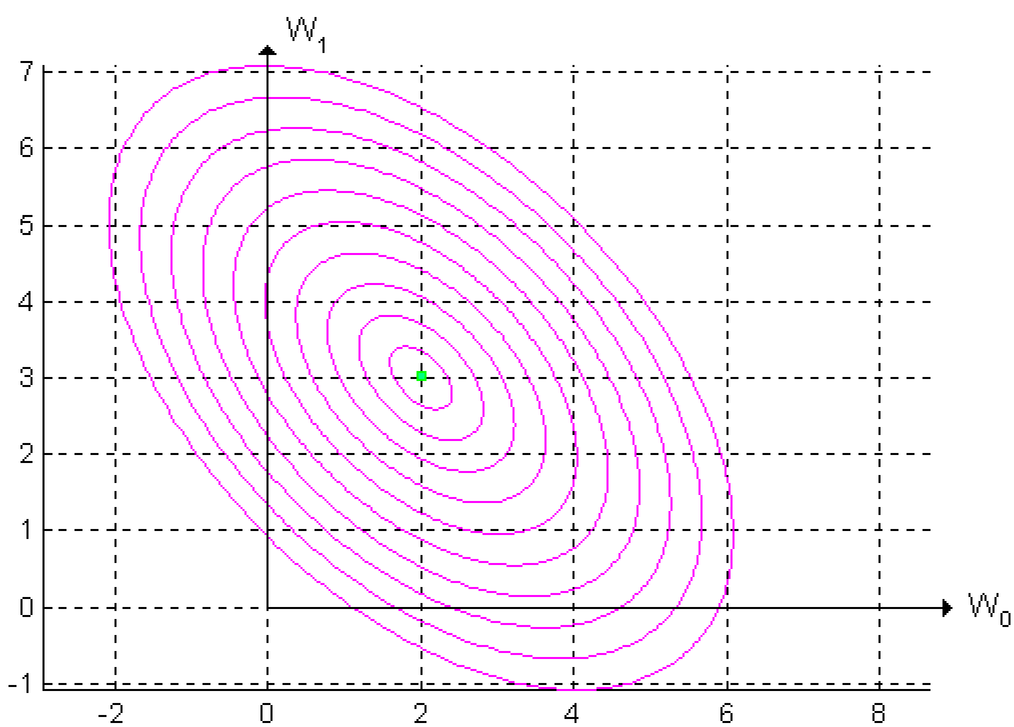
$$\underline{\mathbf{w}}_o^T = \begin{bmatrix} w_0 & w_1 \end{bmatrix} = \begin{bmatrix} 2 & 3 \end{bmatrix} \quad \text{e} \quad J(\underline{\mathbf{w}}_o) = J_{Min} = 4$$

Após a eliminação da translação, obtém-se a Eq.(2.67.b) no sistema de coordenadas $\underline{\mathbf{v}}$.

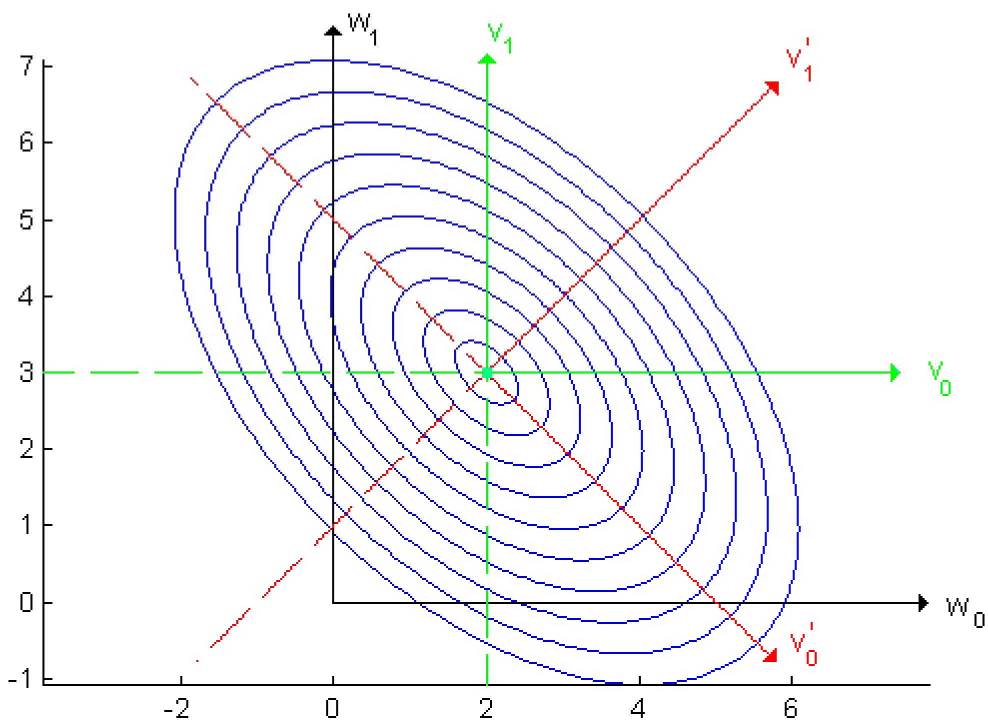
$$J(\underline{v}) = 2v_0^2 + 2v_1^2 + 2v_0 v_1 + J_{\text{Min}} \tag{2.67.b}$$


15000
10000
5000
0
J_{Min}= 4
40
20
3
0
-20
-40
w_1
-40
-20
0
2
20
40
w_0

a)


W_1
7
6
5
4
3
2
1
0
-1
-2
0
2
4
6
8
W_0

b)



c)

Fig.2.5 a) Superfície de desempenho do erro (MSE) referente a Eq.(2.67.a).

b)Curvas de nível do erro quadrático médio (MSE), tomando-se como referência o sistema de coordenadas $\underline{\mathbf{w}}$. Não elimina translação nem rotação.

c) Curvas de nível do erro quadrático médio (MSE) e os diversos sistemas de coordenadas.

Ao final, reescreve-se a Eq.(2.67.b) sob a forma da Eq.(2.64) e aplica-se a Eq.(2.65.a) a esse resultado para se obter no sistema de coordenadas $\underline{\mathbf{v}}^{'}$ a seguinte equação:

$$J(\underline{v}^{'}) = v_0^{'2} + 3v_1^{'2} + 4 \qquad (2.67.c)$$

que é a superfície de desempenho de erro na forma canônica.

Na Fig.(2.5.a) observa-se que devido a presença de termos de 1º grau, existe uma translação entre a origem da superfície de erro e a do sistema de coordenadas $\underline{\mathbf{w}}$. Fazendo-se uma interseção entre a superfície de erro e o plano coordenado $J(\underline{\mathbf{w}})=0$, plano paralelo ao plano $w_0 w_1$ na cota $J(\underline{\mathbf{w}})=0$, então, o resultado é um conjunto vazio. Isso vale para todos valores de $J(\underline{\mathbf{w}})<J_{Min}$. Fazendo-se uma interseção entre a superfície de erro e o plano coordenado $J(\underline{\mathbf{w}})=J_{Min}$, o resultado é um ponto. Finalmente, fazendo-se uma interseção entre a superfície de erro e diversos planos coordenados $J(\underline{\mathbf{w}})>J_{Min}$, os resultados são elipses concêntricas associadas aos respectivos valores constantes de $J(\underline{\mathbf{w}})$, como mostra a Fig.(2.5.b). Observa-se que existe uma translação entre as suas origens e a do sistema de coordenadas $\underline{\mathbf{w}}$, devido à presença de termos de 1º grau. Existe também uma rotação entre os seus eixos principais e $\underline{\mathbf{w}}$, rotação essa cujo ângulo é fornecido pelos autovetores da matriz $\underline{\mathbf{R}}$.

A Eq.(2.67.c) é resultante da eliminação da translação e rotação presentes na Eq.(2.67.a). É interessante notar que a eliminação da translação não alterou os valores dos coeficientes referentes aos termos de 2º grau, puros ou cruzados, e nem a rotação modificou o valor do termo independente J_{Min}. Para efeito comparativo, a Fig.(2.5.c) apresenta as elipses (curvas de nível) em relação a $\underline{\mathbf{v}}'$ e aos antigos sistemas de coordenadas $\underline{\mathbf{w}}$ e $\underline{\mathbf{v}}$.

CAPÍTULO 3

Estrutura do Filtro Considerando-se o Sinal de Entrada Decomposto e Obtenção da Função Custo J

3.1 Introdução

Como visto no capítulo anterior, foi encontrado um conjunto de coeficientes do filtro W(z) que minimizavam o erro entre o sinal desejado d(n) e o sinal de entrada x(n), como mostrado na Fig.(2.1). Para isso, utilizou-se como meta de otimização a minimização da Função Custo J que emprega o erro quadrático médio (MSE) entre o sinal d(n) e o sinal filtrado y(n).

Neste capítulo, pretende-se introduzir modificações na filtragem ótima baseando-se na transformada DSTr [41-44]. A nova estrutura é gerada a partir da decomposição do sinal de entrada x(n), da mesma forma que é feita na citada transformada. O objetivo permanece inalterado, ou seja, encontrar um conjunto de coeficientes que minimize a Função Custo J. A diferença é que em vez de se ter um único sinal de entrada x(n), com a sua decomposição, tem-se a possibilidade de expandir para mais de um o número de sinais e consequentemente também expandir o número de entradas.

Resumindo, a proposta deste capitulo é apresentar uma nova estrutura para a filtragem linear, idealizada a partir da decomposição do sinal de entrada, ou seja, que se baseia em um procedimento similar ao utilizado pela transformada DSTr; partindo-se da decomposição da entrada, obter uma expressão para a Função Custo J a qual é necessária para uma futura utilização em algoritmos adaptativos e por último obter um sistema de equações lineares denominado de sistema de equações “D”, gerado a partir da aplicação do vetor gradiente na Função Custo J, dada pela Eq.(2.7), cuja solução relaciona os coeficientes dos filtros com o processo aleatório de entrada.

A motivação que levou a utilização do procedimento de decomposição, foi a possibilidade de obtenção de uma maior velocidade de convergência associada a um menor erro de regime permanente, em algoritmos adaptativos baseados no LMS, conforme será visto no Capítulo 5. Com esse objetivo procurou-se introduzir esse conceito na base, ou seja, na filtragem de Wiener e posteriormente relacionar as equações obtidas nessa etapa, em particular a da função custo J, com as equações relativas aos algoritmos adaptativos baseados no LMS, que utilizam o processo de decomposição do sinal de entrada. Portanto, a inclusão desse procedimento na filtragem de Wiener e a conseqüente geração do sistema de equações D não proporciona vantagens aparentes sobre o sistema de Wiener-Hopf, no entanto, ele é vantajoso quando aplicado em algoritmos adaptativos baseados no LMS, além de fornecer um desenvolvimento matemático que conecta a teoria da filtragem de Wiener com os algoritmos adaptativos propostos.

3.2 Decomposição de um Sinal Utilizando-se a Transformada DSTr

A idéia fundamental que rege a formulação da Transformada DSTr baseia-se na separação do sinal de entrada em dois outros [41]. A seguir, é explicado esse processo de decomposição para o caso de uma seqüência genérica $g(n)$, não nula no intervalo $-N+1 \leq n \leq 0$, e representada na forma vetorial como um vetor coluna de comprimento N dado por:

$$\underline{\mathbf{g}} = [g(0) \quad g(-1) \quad \cdots \quad g(-N+1)]^T \tag{3.1}$$

Define-se o procedimento de pré-rotação como sendo a diferença entre a seqüência $g(n)$ e a reta $r(n)$, que passa pelos pontos $g(-N+1)$ e $g(0)$ da seqüência $g(n)$. Esse procedimento gera uma seqüência $gr(n)$ definida como sendo:

$$gr(n) = g(n) - r(n) \quad \text{para} \quad -N+1 \leq n \leq 0 \tag{3.2}$$

A representação na forma de um vetor coluna é dada por:

$$\underline{\mathbf{gr}} = [gr(0) \quad gr(-1) \quad \cdots \quad gr(-N+1)]^T \tag{3.3}$$

Além disso, tem-se:

$$r(n) = \frac{g(0) - g(-N+1)}{N-1}(N-1+n) + g(-N+1) \quad \text{para} \quad -N+1 \leq n \leq 0 \tag{3.4}$$

Pode-se também representar um conjunto de valores de $\mathrm{r}(n)$ na forma de um vetor coluna dado por:

$$\underline{\mathbf{r}} = [r(0) \quad r(-1) \quad \cdots \quad r(-N+1)]^T \tag{3.5}$$

Baseado nessa mesma idéia, pode-se modificar a seqüência de entrada $x(n)$ na filtragem de Wiener separando-a em duas outras. Seja a seqüência de entrada $x(n) = x(0), \quad x(1), \quad x(2), \quad \cdots$ e a partir dela define-se, a cada instante de tempo n, o vetor coluna constituído das N amostras mais recentes dessa seqüência:

$$\underline{\mathbf{x}}(n) = [x(n) \;\; x(\text{n-1}) \;\; x(\text{n-2}) \;\; .... \;\; x(\text{n-N}+1)]^T \tag{3.6}$$

De forma análoga, aplica-se o procedimento de pré-rotação no vetor coluna $\underline{\mathbf{x}}(n)$, tendo-se como resultado a criação de dois outros vetores $\underline{\mathbf{r}}(n)$ e $\underline{\mathbf{xr}}(n)$. Seja $\underline{\mathbf{r}}(n)$ o vetor coluna com elementos definidos por:

$$r(n-i) = \frac{x(n) - x(n-N+1)}{N-1}(N-1-i) + x(n-N+1) \quad \text{para} \quad \begin{array}{c} 0 \le \text{n} \\ \text{e} \\ 0 \le i \le N-1 \end{array} \tag{3.7.a}$$

logo, tem-se:

$$\underline{\mathbf{r}}(n) = [r(n) \quad r(n-1) \quad \cdots \quad r(n-N+1)]^T \tag{3.7.b}$$

Seja $\underline{\mathbf{xr}}(n)$ o vetor coluna cujos elementos são definidos por:

$$xr(n-i) = x(n-i) - r(n-i) \quad \text{para} \quad \begin{array}{c} 0 \le \text{n} \\ \text{e} \\ 0 \le i \le N-1 \end{array} \tag{3.8.a}$$

logo, tem-se:

$$\underline{\mathbf{xr}}(n) = [xr(n) \quad xr(n-1) \quad \cdots \quad xr(n-N+1)]^T \tag{3.8.b}$$

A partir deste ponto, observa-se uma importante mudança na maneira de introduzir o sinal de entrada no filtro. Pelas Figs.(2.1) e (2.2) têm-se a introdução das amostras do sinal x(n) serialmente, dessa forma cada derivação do filtro FIR recebe à medida que "n" evolui, exatamente a mesma amostra do sinal x(n) presente na derivação imediatamente anterior a ela. Já na estrutura proposta, como mostra a Fig.(3.1), tem-se N amostras provenientes do vetor **xr**(n) e N amostras do vetor **r**(n), que respectivamente abastecem de forma paralela as entradas de cada um dos sistemas W(z) e WW(z), a medida que "n" evolui. Para maior clareza, admita que o sistema W(z) seja constituído de diversas saídas cujos valores são iguais às respectivas entradas ponderadas por um peso específico, dados por $\{w_i\}$. A soma de todas essas saídas gera o sinal y(n). De forma análoga, tem-se que yy(n) representa a soma de todas as saídas do sistema WW(z).

A Fig.(3.1) apresenta a reformulação do problema de filtragem considerando-se a inserção da $\underline{\mathbf{xr}}(n)$ e $\underline{\mathbf{r}}(n)$ no lugar de $\underline{\mathbf{x}}(n)$ conforme explanado anteriormente. O objetivo nessa proposta de filtragem é minimizar o erro entre o sinal desejado $d(n)$ e a soma das saídas de cada um dos sistemas. Define-se a seqüência de saída completa como sendo a soma dos sinais provenientes de W(z) e WW(z), isto é,

$$yc(n) = y(n) + yy(n) \tag{3.9}$$

A representação dos sistemas W(z) e WW(z) é feita através de dois vetores $\underline{\mathbf{w}}$ e $\underline{\mathbf{ww}}$,

$$\underline{\mathbf{w}} = [w_0 \; w_1 \;\; w_2 \;\; \mathbf{...} \;\; w_{N\text{-}1}]^T \tag{3.10.a}$$

$$\underline{\mathbf{ww}} = [ww_0 \; ww_1 \;\; ww_2 \;\; \mathbf{...} \;\; ww_{N\text{-}1}]^T \tag{3.10.b}$$

Dessa forma, pode-se definir a saída $y(n)$ em função do conjunto de pesos $\{w_i\}$ como:

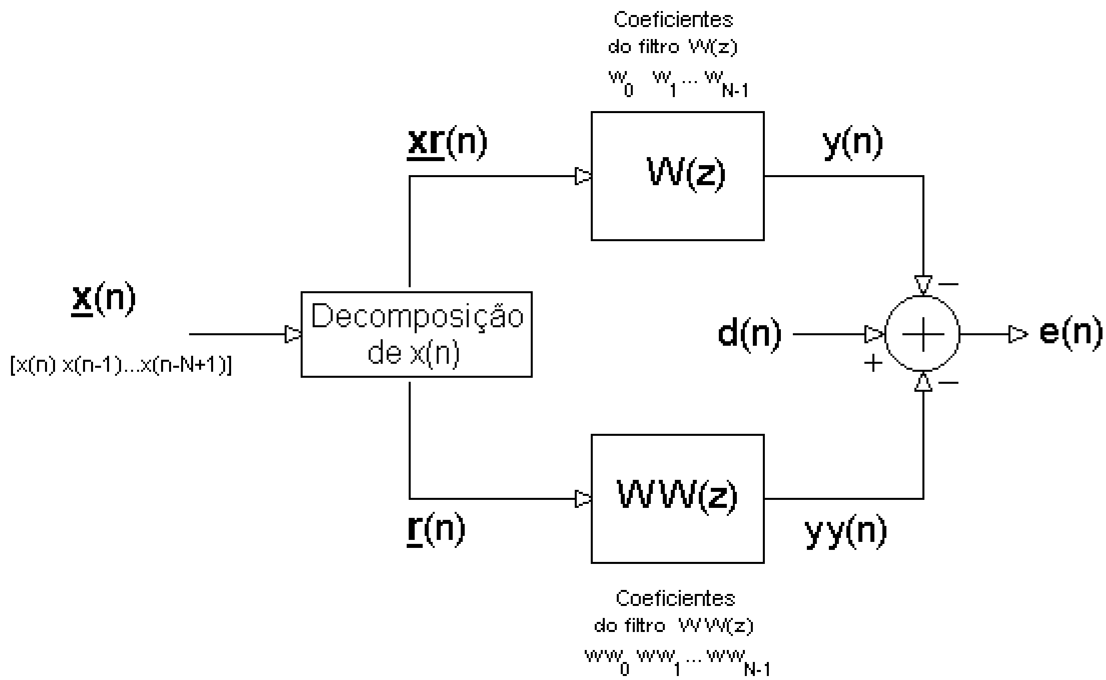


Fig.3.1 Modelo empregado na representação do problema da filtragem linear considerando-se a decomposição do sinal de entrada x(n) em dois outros sinais xr(n) e r(n).

$$y(n) = \sum_{k=0}^{N-1} w_k \, xr(n-k) \qquad \text{para n} \geq 0 \qquad (3.11.a)$$

Na sua forma matricial, tem-se:

$$y(n) = \underline{\mathbf{w}}^T \, \underline{\mathbf{xr}}(n) \qquad (3.11.b)$$

Define-se também a saída $yy(n)$ por:

$$yy(n) = \sum_{k=0}^{N-1} ww_k \, r(n-k) \qquad \text{para n} \geq 0 \qquad (3.11.c)$$

ou por:

$$yy(n) = \underline{\mathbf{ww}}^T \, \underline{\mathbf{r}}(n) \tag{3.11.d}$$

Tem-se que a soma de cada uma das saídas dos sistemas W(z) e WW(z) resulta na seqüência $yc(n)$ utilizada como uma estimativa de $d(n)$. Logo, tem-se:

$$\hat{d}(n) = yc(n) \tag{3.12.a}$$

$$\hat{d}(n) = \sum_{k=0}^{N-1} w_k xr(n-k) + \sum_{k=0}^{N-1} ww_k r(n-k) \qquad \text{para} \quad n \geq 0 \tag{3.12.b}$$

ou por:

$$\hat{d}(n) = \underline{\mathbf{w}}^T \, \underline{\mathbf{xr}}(n) + \underline{\mathbf{ww}}^T \, \underline{\mathbf{r}}(n) \tag{3.12.c}$$

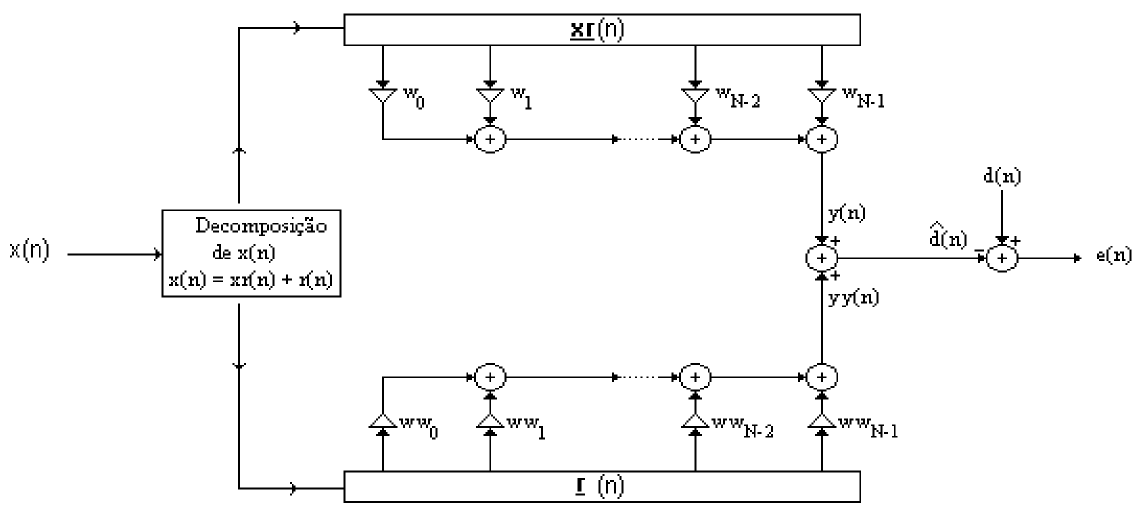


Fig.3.2. Processo de obtenção da estimativa da seqüência desejada d(n).

Pode-se ainda, expressar $\hat{d}(n)$ como sendo:

$$\hat{d}(n) = \underline{\theta}^T \, \underline{\mathbf{u}}(n) \tag{3.13.a}$$

onde

$$\underline{\mathbf{u}}(n) = [xr(n) \quad xr(n-1) \quad \cdots \quad xr(n-N+1) \quad r(n) \quad r(n-1) \quad \cdots \quad r(n-N+1)]^T \tag{3.13.b}$$

$$\underline{\theta} = [w_0 \ \mathrm{w}_1 \ \ldots \ \mathrm{w}_{\text{N-1}} \ ww_0 \ \mathrm{ww}_1 \ \ldots \ \mathrm{ww}_{\text{N-1}}]^T \tag{3.13.c}$$

Logo, o erro de estimação é definido como sendo a diferença entre a resposta desejada e o sinal estimado, isto é:

$$e(n) = d(n) - \hat{d}(n) \tag{3.14}$$

A Fig.(3.2) apresenta o processo de obtenção da estimação do sinal desejado d(n). Inicialmente, ocorre a decomposição de $\underline{\mathbf{x}}(n)$ em $\underline{\mathbf{xr}}(n)$ e $\underline{\mathbf{r}}(n)$, em seguida cada uma dessas seqüências decompostas é filtrada utilizando-se os sistemas W(z) e WW(z). Por último, as seqüências nas saídas desses sistemas são somadas a fim de fornecerem uma estimativa de d(n), por conseguinte uma seqüência de erro associada a essa estimativa.

Diante do que foi exposto, existe uma dada interpretação para a transformação do problema da filtragem de Wiener quando se utiliza a decomposição do sinal x(n). Traçando-se um paralelo direto com sistemas de múltiplas entradas, pode-se dizer que com a decomposição de $\underline{\mathbf{x}}(n)$ gerou-se dois vetores $\underline{\mathbf{xr}}(n)$ e $\underline{\mathbf{r}}(n)$, sendo a dimensão de cada um deles Nx1. Cada um desses N elementos representa amostras de N diferentes seqüências no tempo "n" e dessa forma foram gerados 2*N sinais a partir do vetor $\underline{\mathbf{x}}$(n). À medida que "n" evolui continuamente, vai ocorrendo a decomposição de $\underline{\mathbf{x}}$(n) e, por conseguinte, cada um dos 2*N sinais gerados são processados pelos sistemas W(z) e WW(z) .

A Fig.(3.3) apresenta o conjunto de sinais gerados a partir da decomposição, o sinal desejado d(n) e o erro de estimação e(n). Dessa nova interpretação do problema, cuja representação em diagramas de blocos consta na citada figura, são redefinidas as variáveis do sistema sob essa nova ótica. O vetor coluna constituído das N amostras mais recentes da seqüência de entrada é o mesmo, dado por $\underline{\mathbf{x}}(n)$, porém a forma de conceituá-lo é modificada no aspecto de que esse vetor é formado pelas amostras tomadas, a cada instante "n", de N seqüências de entradas, dadas por:

$$\underline{\mathbf{x}}(n) = [x_0(n) \;\; x_1(n) \;\; x_2(n) \;\; .... \;\; x_{N-1}(n)]^T \tag{3.15.a}$$

e

$$x_i(n) = x(n-i) \quad para \quad \begin{matrix} 0 \le n \\ e \\ 0 \le i \le N-1 \end{matrix} \tag{3.15.b}$$

representa a amostra da seqüência x_i no instante "n".

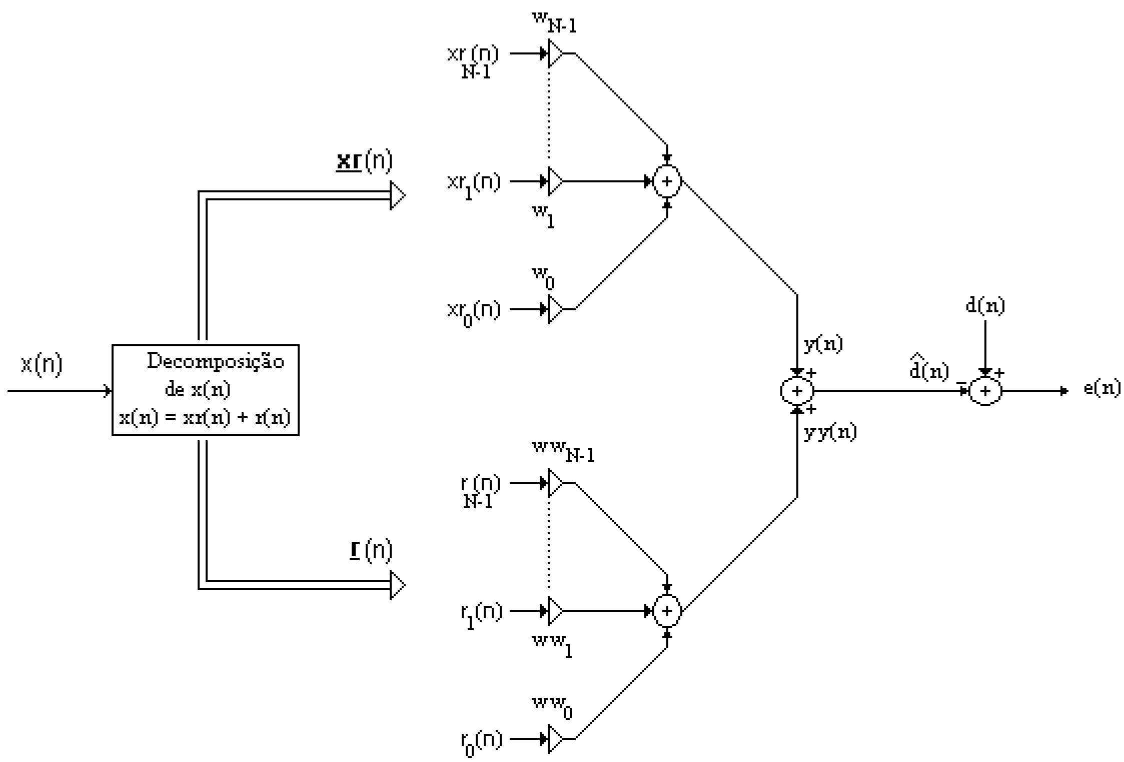


Fig.3.3. Transformação de um sistema com entrada simples para um sistema de múltiplas entradas e o processo de obtenção da estimativa da seqüência desejada d(n).

De forma análoga, aplica-se o procedimento de pré-rotação no vetor coluna $\underline{\mathbf{x}}(n)$, tendo-se como resultado a criação de dois outros vetores $\underline{\mathbf{r}}(n)$ e $\underline{\mathbf{xr}}(n)$. Seja $\underline{\mathbf{r}}(n)$ o vetor coluna com elementos definidos por:

$$r_i(n) = \frac{x_0(n) - x_{(N-1)}(n)}{N-1}(N-1-i) + x_{(N-1)}(n) \quad \text{para} \quad \begin{matrix} 0 \leq \text{n} \\ \text{e} \\ 0 \leq i \leq N-1 \end{matrix} \tag{3.16.a}$$

que representa a amostra da seqüência r_i no instante "n". Logo, tem-se:

$$\underline{\mathbf{r}}(n) = [r_0(n) \;\; r_1(n) \;\; r_2(n) \;\; .... \;\; r_{(N-1)}(n)]^T \tag{3.16.b}$$

Observa-se que os valores das amostras presentes nas Eqs. (3.16.a) e (3.16.b) permanecem os mesmos em relação aos valores presentes nas Eqs.(3.7.a) e (3.7.b). Apenas a notação foi modificada em função da interpretação dada ao problema, isto é, considera-se o sistema com múltiplas entradas. Portanto, pela Fig.(3.3) e utilizando-se a Eq.(3.16.b), pode-se representar a saída yy(n) como sendo:

$$yy(n) = \underline{\mathbf{ww}}^{\mathrm{T}} \underline{\mathbf{r}}(n) \tag{3.17}$$

Seja $\underline{\mathbf{xr}}(n)$ o vetor coluna com elementos definidos por:

$$xr_i(n) = x_i(n) - r_i(n) \, \text{para} \quad \begin{matrix} 0 \leq \text{n} \\ \text{e} \\ 0 \leq \text{i} \leq N-1 \end{matrix} \tag{3.18.a}$$

que representa uma amostra da seqüência xr_i no instante "n", logo,

$$\underline{\mathbf{x\,r}}(n) = [xr_0(n) \;\; xr_1(n) \;\; xr_2(n) \;\; .... \;\; xr_{(N-1)}(n)]^T \tag{3.18.b}$$

Novamente, observa-se que os valores das amostras presentes nas Eqs. (3.18.a) e (3.18.b) permanecem os mesmos em relação aos valores presentes nas Eqs. (3.8.a) e (3.8.b), apenas a notação foi modificada como explicado anteriormente. Portanto, pela Fig.(3.3) e utilizando-se a Eq.(3.18.b), pode-se representar a saída y(n) como sendo:

$$y(n) = \underline{\mathbf{w}}^{T}\,\underline{\mathbf{xr}}(n) \tag{3.19}$$

A Fig.(3.3) apresentou de forma explícita a geração das 2*N seqüências a serem processadas pelos sistemas W(z) e WW(z) de múltiplas entradas, a somatória das saídas dos dois sistemas e a formação da estimativa de d(n). Em suma, o sinal estimado $\hat{d}(n)$ é uma combinação linear das 2*N amostras obtidas pelo processo de decomposição.

3.3 Determinação da Função Custo J Baseando-se nas Esperanças dos Sinais Decompostos

Usando-se um procedimento análogo ao da seção 2.3, o objetivo é encontrar uma solução para o problema representado na Fig.(3.3). Admite-se que as múltiplas entradas $xr_i(n)$ presentes no sistema da referida figura e o sinal desejado d(n) sejam estacionários no sentido amplo e que essas entradas possuam algum nível de correlação com o sinal desejado. Partindo-se da Função Custo J definida pela Eq.(2.34), deseja-se encontrar um conjunto de coeficientes para os sistemas W(z) e WW(z) de forma a minimizar o erro de estimação e(n). Para isso, torna-se necessário relacionar a Função Custo J em termos dos coeficientes do filtro, isto é, dos pesos referentes a cada uma das entradas. Posteriormente a aplicação do operador gradiente na Função Custo J e a imposição de que cada componente desse vetor seja nula, gera um conjunto de equações lineares cuja solução minimiza o erro de estimação.

Seja a Função Custo J ou J(**w**), dada pela Eq.(2.34) reescrita a seguir como:

$$J = \mathbf{E}[e^2(n)] = \mathbf{E}\{[d(n) - \hat{d}(n)]^2\} \tag{3.20}$$

onde e(n) representa o erro de estimação entre o sinal desejado e o sinal estimado. Utilizando-se as Eqs.(3.9), (3.12.a), na Eq.(3.14), chega-se em:

$$e(n) = d(n) - y(n) - yy(n) \tag{3.21.a}$$

$$e(n) = d(n) - \sum_{i=0}^{N-1} [w_i xr_i(n) + ww_i r_i(n)]$$

$$e(n) = d(n) - \sum_{i=0}^{N-1} w_i xr_i(n) \ - \sum_{i=0}^{N-1} ww_i r_i(n) \qquad \text{para} \quad n \geq 0 \tag{3.21.b}$$

Reescrevendo-se a Eq. (3.21.b) em termos das Eqs. (3.17) e (3.19), tem-se:

$$e(n) = d(n) - \underline{\mathbf{w}}^T \, \underline{\mathbf{xr}}(n) - \underline{\mathbf{ww}}^T \, \underline{\mathbf{r}}(n) \tag{3.21.c}$$

Substituindo-se a Eq.(3.21.a) na Eq.(3.20),

$$J = \mathbf{E}[d^2(n) - 2d(n)y(n) - 2d(n)yy(n) + y^2(n) + 2y(n)yy(n) + yy^2(n)]$$

$$J = \mathbf{E}[d^2(n)] - 2\,\mathbf{E}[d(n)y(n)] - 2\,\mathbf{E}[d(n)yy(n)] + \mathbf{E}[y^2(n)] + 2\,\mathbf{E}[y(n)yy(n)] + \mathbf{E}[yy^2(n)] \tag{3.22}$$

Na Eq.(3.22) aparecem algumas médias estatísticas cujas definições e respectivas interpretações são apresentadas a seguir:

a) $\mathbf{E}[d^2(n)] = \sigma_d^2$ é o valor quadrático médio do sinal desejado. (3.23)

b) $\mathbf{E}[d(n)y(n)]$ pode ser expandida através da utilização da Eq.(3.19). Logo, tem-se:

$$\mathbf{E}[d(n)y(n)] = \mathbf{E}[y(n)d(n)] = \mathbf{E}[\underline{\mathbf{w}}^T \, \underline{\mathbf{xr}}(n)d(n)] = \underline{\mathbf{w}}^T \, \mathbf{E}[\underline{\mathbf{xr}}(n)d(n)]$$

$$\mathbf{E}[d(n)y(n)] = \underline{\mathbf{w}}^T \, \mathbf{E}[d(n)\underline{\mathbf{xr}}(n)] = \underline{\mathbf{w}}^T \, \underline{\mathbf{p}} \tag{3.24}$$

onde,

$$\underline{\mathbf{p}} = \mathbf{E}[d(n)xr_0(n) \quad d(n)xr_1(n) \quad \cdots \quad d(n)xr_{N-1}(n)]^T \tag{3.25.a}$$

$$\underline{\mathbf{p}} = [p_0 \quad p_1 \quad \cdots \quad p_{N-1}]^T \tag{3.25.b}$$

representa a correlação entre o sinal desejado $d(n)$ e as entradas $xr_i(n)$.

c) $\mathbf{E}[d(n)yy(n)]$ pode ser expandida através da utilização da Eq. (3.17). Logo, tem-se:

$$\mathbf{E}[d(n)yy(n)] = \mathbf{E}[yy(n)d(n)] = \mathbf{E}[\underline{\mathbf{ww}}^T\,\underline{\mathbf{r}}(n)d(n)] = \underline{\mathbf{ww}}^T\,\mathbf{E}[\underline{\mathbf{r}}(n)d(n)]$$

$$\mathbf{E}[d(n)yy(n)] = \underline{\mathbf{ww}}^T\,\mathbf{E}[d(n)\underline{\mathbf{r}}(n)] = \underline{\mathbf{ww}}^T\,\underline{\mathbf{pp}} \tag{3.26}$$

onde,

$$\underline{\mathbf{pp}} = \mathbf{E}[d(n)r_0(n) \quad d(n)r_1(n) \quad \cdots \quad d(n)r_{N-1}(n)]^T \tag{3.27.a}$$

$$\underline{\mathbf{pp}} = [pp_0 \quad pp_1 \quad \cdots \quad pp_{N-1}]^T \tag{3.27.b}$$

representa a correlação entre o sinal desejado $d(n)$ e as diversas entradas $r_i(n)$.

d) $\mathbf{E}[y^2(n)]$ é o valor quadrático médio de y(n)

$$\mathbf{E}[y^2(n)] = \mathbf{E}[\underline{\mathbf{w}}^T\,\underline{\mathbf{xr}}(n)\underline{\mathbf{xr}}^T(n)\;\underline{\mathbf{w}}] = \underline{\mathbf{w}}^T\,\mathbf{E}[\underline{\mathbf{xr}}(n)\underline{\mathbf{xr}}^T(n)]\;\underline{\mathbf{w}} \tag{3.28.a}$$

$$\mathbf{E}[y^2(n)] = \underline{\mathbf{w}}^T\,\underline{\mathbf{R}}_{xr}\,\underline{\mathbf{w}} \tag{3.28.b}$$

onde,

$$\underline{\mathbf{R}}_{xr} = \mathbf{E}\begin{bmatrix} xr_0(n)xr_0(n) & xr_0(n)xr_1(n) & \cdots & xr_0(n)xr_{N-1}(n) \\ xr_1(n)xr_0(n) & xr_1(n)xr_1(n) & \cdots & xr_1(n)xr_{N-1}(n) \\ \vdots & \vdots & \ddots & \vdots \\ xr_{N-1}(n)xr_0(n) & xr_{N-1}(n)xr_1(n) & \cdots & xr_{N-1}(n)xr_{N-1}(n) \end{bmatrix} \tag{3.29}$$

$$\underline{\mathbf{R}}_{xr} = \begin{bmatrix} R_{xr_0 xr_0} & R_{xr_0 xr_1} & \cdots & R_{xr_0 xr_{N-1}} \\ R_{xr_1 xr_0} & R_{xr_1 xr_1} & \cdots & R_{xr_1 xr_{N-1}} \\ \vdots & \cdots & \ddots & \vdots \\ R_{xr_{N-1} xr_0} & R_{xr_{N-1} xr_1} & \cdots & R_{xr_{N-1} xr_{N-1}} \end{bmatrix} \tag{3.30}$$

representa a correlação cruzada entre as entradas $xr_i(n)$.

e) $\mathbf{E}[yy^2(n)]$ é o valor quadrático médio de yy(n)

$$\mathbf{E}[yy^2(n)] = \mathbf{E}[\underline{\mathbf{ww}}^T\, \underline{\mathbf{r}}(n)\underline{\mathbf{r}}^T(n)\underline{\mathbf{ww}}] = \underline{\mathbf{ww}}^T\, \mathbf{E}[\underline{\mathbf{r}}(n)\underline{\mathbf{r}}^T(n)]\underline{\mathbf{ww}} \tag{3.31.a}$$

$$\mathbf{E}[y^2(n)] = \underline{\mathbf{ww}}^T\, \underline{\mathbf{R}}_r\, \underline{\mathbf{ww}} \tag{3.31.b}$$

onde,

$$\underline{\mathbf{R}}_r = \mathbf{E}\begin{bmatrix} r_0(n)r_0(n) & r_0(n)r_1(n) & \cdots & r_0(n)r_{N-1}(n) \\ r_1(n)r_0(n) & r_1(n)r_1(n) & \cdots & r_1(n)r_{N-1}(n) \\ \vdots & \vdots & \ddots & \vdots \\ r_{N-1}(n)r_0(n) & r_{N-1}(n)r_1(n) & \cdots & r_{N-1}(n)r_{N-1}(n) \end{bmatrix} \tag{3.32}$$

$$\underline{\mathbf{R}}_r = \begin{bmatrix} R_{r_0 r_0} & R_{r_0 r_1} & \cdots & R_{r_0 r_{N-1}} \\ R_{r_1 r_0} & R_{r_1 r_1} & \cdots & R_{r_1 r_{N-1}} \\ \vdots & \cdots & \ddots & \vdots \\ R_{r_{N-1} r_0} & R_{r_{N-1} r_1} & \cdots & R_{r_{N-1} r_{N-1}} \end{bmatrix} \tag{3.33}$$

representa a correlação cruzada entre as entradas $r_i(n)$.

f) $\mathbf{E}[y(n)yy(n)]$ é a correlação entre as saídas y(n) e yy(n).

$$\mathbf{E}[y(n)yy(n)] = \mathbf{E}[\underline{\mathbf{w}}^T\, \underline{\mathbf{xr}}(n)\underline{\mathbf{r}}^T(n)\underline{\mathbf{ww}}] = \underline{\mathbf{w}}^T\, \mathbf{E}[\ \underline{\mathbf{xr}}(n)\underline{\mathbf{r}}^T(n)]\underline{\mathbf{ww}} \tag{3.34.a}$$

$$\mathbf{E}[y(n)yy(n)] = \underline{\mathbf{w}}^{\mathrm{T}}\,\underline{\mathbf{R}}_{\mathrm{xr\ r}}\,\underline{\mathbf{ww}} \tag{3.34.b}$$

onde,

$$\mathbf{E}[\underline{\mathbf{xr}}(n)\underline{\mathbf{r}}^{T}(n)] = \mathbf{E}\begin{bmatrix} xr_0(n)r_0(n) & xr_0(n)r_1(n) & \cdots & xr_0(n)r_{N-1}(n) \\ xr_1(n)r_0(n) & xr_1(n)r_1(n) & \cdots & xr_1(n)r_{N-1}(n) \\ \vdots & \cdots & \ddots & \vdots \\ xr_{N-1}(n)r_0(n) & xr_{N-1}(n)r_1(n) & \cdots & xr_{N-1}(n)r_{N-1}(n) \end{bmatrix} \tag{3.35}$$

$$\underline{\mathbf{R}}_{xr\ r} = \begin{bmatrix} R_{xr_0\ r_0} & R_{xr_0\ r_1} & \cdots & R_{xr_0\ r_{N-1}} \\ R_{xr_1\ r_0} & R_{xr_1\ r_1} & \cdots & R_{xr_1\ r_{N-1}} \\ \vdots & \cdots & \ddots & \vdots \\ R_{xr_{N-1}\ r_0} & R_{xr_{N-1}\ r_1} & \cdots & R_{xr_{N-1}\ r_{N-1}} \end{bmatrix} \tag{3.36}$$

representa a matriz de correlação cruzada entre as entradas $xr_i(n)$ e $r_i(n)$.

É conveniente introduzir nesse ponto, a matriz de correlação cruzada entre as entradas $r_i(n)$ e $xr_i(n)$, como sendo:

$$\mathbf{E}[\underline{\mathbf{r}}(n)\underline{\mathbf{xr}}^{T}(n)] = \mathbf{E}\begin{bmatrix} r_0(n)xr_0(n) & r_0(n)xr_1(n) & \cdots & r_0(n)xr_{N-1}(n) \\ r_1(n)xr_0(n) & r_1(n)xr_1(n) & \cdots & r_1(n)xr_{N-1}(n) \\ \vdots & \cdots & \ddots & \vdots \\ r_{N-1}(n)xr_0(n) & r_{N-1}(n)xr_1(n) & \cdots & r_{N-1}(n)xr_{N-1}(n) \end{bmatrix} \tag{3.37}$$

$$\underline{\mathbf{R}}_{r\ xr} = \begin{bmatrix} R_{r_0\ xr_0} & R_{r_0\ xr_1} & \cdots & R_{r_0\ xr_{N-1}} \\ R_{r_1\ xr_0} & R_{r_1\ xr_1} & \cdots & R_{r_1\ xr_{N-1}} \\ \vdots & \cdots & \ddots & \vdots \\ R_{r_{N-1}\ xr_0} & R_{r_{N-1}\ xr_1} & \cdots & R_{r_{N-1}\ xr_{N-1}} \end{bmatrix} \tag{3.38}$$

que nada mais é do que $\underline{\mathbf{R}}_{xr\ r}^{T}$.

Substituindo-se as Eqs.(3.23), (3.24), (3.26), (3.28.b), (3.31.b) e (3.34) na Eq.(3.22) obtém-se uma expressão para a Função Custo J relacionada ao sistema com entrada decomposta, cuja equação é dada por:

$$J(\underline{\mathbf{w}},\underline{\mathbf{ww}}) = \sigma_d^2 - 2\underline{\mathbf{w}}^T\,\underline{\mathbf{p}} + \underline{\mathbf{w}}^T\,\underline{\mathbf{R}}_{xr}\,\underline{\mathbf{w}} - 2\underline{\mathbf{ww}}^T\,\underline{\mathbf{pp}} + \underline{\mathbf{ww}}^T\,\underline{\mathbf{R}}_r\,\underline{\mathbf{ww}} + 2\underline{\mathbf{w}}^T\,\underline{\mathbf{R}}_{xr\,r}\,\underline{\mathbf{ww}} \tag{3.39}$$

3.4 Geração do Sistema de Equações Lineares a Partir da Aplicação do Vetor Gradiente na Função Custo $J(\underline{\mathbf{w}},\underline{\mathbf{ww}})$

A forma de se determinar os valores reais de $\{w_i\}$ e $\{ww_i\}$ para o qual a função real de multivariáveis $J(\underline{\mathbf{w}},\underline{\mathbf{ww}})$ passa por um valor mínimo, é idêntica a forma utilizada no Capítulo 2. Deve-se aplicar o operador gradiente ∇ dado pela Eq.(3.41) na função $J(\underline{\mathbf{w}},\underline{\mathbf{ww}})$ representada pela Eq.(3.40) e impor que cada uma das componentes do vetor $\nabla J(\underline{\mathbf{w}},\underline{\mathbf{ww}})$ seja igual a zero a fim de minimizá-la [19,39,51].

Utilizando-se a Eq.(3.13.c) para agrupar os dois vetores de pesos, tem-se que:

$$\mathrm{J}(\underline{\boldsymbol{\theta}}) = \mathrm{J}(\underline{\mathbf{w}},\underline{\mathbf{ww}}) = \sigma_d^2 - 2\underline{\mathbf{w}}^T\,\underline{\mathbf{p}} + \underline{\mathbf{w}}^T\,\underline{\mathbf{R}}_{xr}\,\underline{\mathbf{w}} - 2\underline{\mathbf{ww}}^T\,\underline{\mathbf{pp}} + \underline{\mathbf{ww}}^T\,\underline{\mathbf{R}}_r\,\underline{\mathbf{ww}} + 2\underline{\mathbf{w}}^T\,\underline{\mathbf{R}}_{xr\,r}\,\underline{\mathbf{ww}} \tag{3.40}$$

Seja o vetor gradiente ∇ que define a operação derivada de um escalar por um vetor $\underline{\mathbf{w}}$, expresso pela Eq.(2.45) e reescrito a seguir como sendo:

$$\frac{\partial(.)}{\partial\underline{\boldsymbol{\theta}}} = \begin{bmatrix} \frac{\partial(.)}{\partial \mathrm{w}_0} & \frac{\partial(.)}{\partial \mathrm{w}_1} & \cdots & \frac{\partial(.)}{\partial \mathrm{w}_{N-1}} & \frac{\partial(.)}{\partial \mathrm{ww}_0} & \frac{\partial(.)}{\partial \mathrm{ww}_1} & \cdots & \frac{\partial(.)}{\partial \mathrm{ww}_{N-1}} \end{bmatrix}^T \tag{3.41}$$

Aplicando-se o operador gradiente sobre $\mathrm{J}(\underline{\boldsymbol{\theta}})$ tem-se que:

$$\frac{\partial(J)}{\partial\underline{\boldsymbol{\theta}}} = \begin{bmatrix} \frac{\partial(J)}{\partial w_0} & \frac{\partial(J)}{\partial w_1} & \cdots & \frac{\partial(J)}{\partial w_{N-1}} & \frac{\partial(J)}{\partial ww_0} & \frac{\partial(J)}{\partial ww_1} & \cdots & \frac{\partial(J)}{\partial ww_{N-1}} \end{bmatrix}^T \tag{3.42}$$

e igualando-se a zero o vetor dado pela Eq.(3.42), tem-se:

$$\frac{\partial(J)}{\partial\underline{\boldsymbol{\theta}}} = \underline{\mathbf{0}} \tag{3.43.a}$$

onde $\underline{\mathbf{0}}$ denota o vetor coluna constituído de N zeros. Portanto, tem-se que:

$$\nabla_k J = 0 \tag{3.43.b}$$

Considerando-se que $\underline{\mathbf{w}}^T\underline{\mathbf{p}}$, $\underline{\mathbf{w}}^T\underline{\mathbf{R}}_{xr}\underline{\mathbf{w}}$, $\underline{\mathbf{ww}}^T\underline{\mathbf{pp}}$, $\underline{\mathbf{ww}}^T\underline{\mathbf{R}}_r\underline{\mathbf{ww}}$ e $\underline{\mathbf{w}}^T\underline{\mathbf{R}}_{xr\,r}\underline{\mathbf{ww}}$ são escalares e que $\boldsymbol{\sigma}_d^2$ independe de $\underline{\mathbf{w}}$ ou de $\underline{\mathbf{w\,w}}$, portanto, tem-se que:

$$\frac{\partial J}{\partial\underline{\mathbf{w}}} = -2\frac{\partial\left(\underline{\mathbf{w}}^T\underline{\mathbf{p}}\right)}{\partial\underline{\mathbf{w}}} + \frac{\partial\left(\underline{\mathbf{w}}^T\underline{\mathbf{R}}_{xr}\underline{\mathbf{w}}\right)}{\partial\underline{\mathbf{w}}} + 2\frac{\partial\left(\underline{\mathbf{w}}^T\underline{\mathbf{R}}_{xr\,r}\underline{\mathbf{ww}}\right)}{\partial\underline{\mathbf{w}}} = \underline{\mathbf{0}} \tag{3.44.a}$$

$$\frac{\partial J}{\partial\underline{\mathbf{ww}}} = -2\frac{\partial\left(\underline{\mathbf{ww}}^T\underline{\mathbf{pp}}\right)}{\partial\underline{\mathbf{ww}}} + \frac{\partial\left(\underline{\mathbf{ww}}^T\underline{\mathbf{R}}_r\underline{\mathbf{ww}}\right)}{\partial\underline{\mathbf{ww}}} + 2\frac{\partial\left(\underline{\mathbf{w}}^T\underline{\mathbf{R}}_{xr\,r}\underline{\mathbf{ww}}\right)}{\partial\underline{\mathbf{ww}}} = \underline{\mathbf{0}} \tag{3.44.b}$$

As Eqs.(3.44.a) e (3.44.b) possuem algumas derivadas que serão calculadas a seguir:

a)Para calcular a derivada

$$\frac{\partial\left(\underline{\mathbf{w}}^T\underline{\mathbf{p}}\right)}{\partial\underline{\mathbf{w}}} \tag{3.45}$$

deve-se utilizar a Eq.(2.45), resultando em:

$$\frac{\partial\left(\underline{\mathbf{w}}^T\underline{\mathbf{p}}\right)}{\partial\underline{\mathbf{w}}} = \left[\frac{\partial\left(\underline{\mathbf{w}}^T\underline{\mathbf{p}}\right)}{\partial w_0} \quad \frac{\partial\left(\underline{\mathbf{w}}^T\underline{\mathbf{p}}\right)}{\partial w_1} \quad \cdots \quad \frac{\partial\left(\underline{\mathbf{w}}^T\underline{\mathbf{p}}\right)}{\partial w_{N-1}}\right]^T \tag{3.46}$$

Além disso, tem-se:

$$\frac{\partial\left(\underline{\mathbf{w}}^T\,\underline{\mathbf{p}}\right)}{\partial w_0}=\frac{\partial\left(w_0p_0+w_1p_1+\ldots+w_{N-1}p_{N-1}\right)}{\partial w_0}$$

$$\frac{\partial\left(\underline{\mathbf{w}}^T\,\underline{\mathbf{p}}\right)}{\partial w_0}=p_0$$

logo, basta estender esse procedimento a todos os filtros para se obter:

$$\frac{\partial\left(\underline{\mathbf{w}}^T\,\underline{\mathbf{p}}\right)}{\partial\underline{\mathbf{w}}}=\begin{bmatrix}p_0 & p_1 & \cdots & p_{N-1}\end{bmatrix}^T \tag{3.47.a}$$

$$\frac{\partial\left(\underline{\mathbf{w}}^T\,\underline{\mathbf{p}}\right)}{\partial\underline{\mathbf{w}}}=\underline{\mathbf{p}} \tag{3.47.b}$$

b) Para se calcular a derivada

$$\frac{\partial\left(\underline{\mathbf{ww}}^{\mathrm{T}}\,\underline{\mathbf{pp}}\right)}{\partial\underline{\mathbf{ww}}}$$

procede-se de forma análoga ao caso a) resultando em:

$$\frac{\partial\left(\underline{\mathbf{ww}}^T\,\underline{\mathbf{pp}}\right)}{\partial\underline{\mathbf{ww}}}=\begin{bmatrix}pp_0 & pp_1 & \cdots & pp_{N-1}\end{bmatrix}^T \tag{3.48.a}$$

$$\frac{\partial\left(\underline{\mathbf{ww}}^T\,\underline{\mathbf{pp}}\right)}{\partial\underline{\mathbf{ww}}}=\underline{\mathbf{pp}} \tag{3.48.b}$$

c) Para se calcular a derivada

$$\frac{\partial\left(\underline{\mathbf{w}}^T\,\underline{\mathbf{R}}_{xr}\,\underline{\mathbf{w}}\right)}{\partial\,\mathbf{w}} \tag{3.49}$$

Utilizando-se as Eqs.(3.28.a) e (3.28.b), de modo que:

$$\underline{\mathbf{w}}^T \underline{\mathbf{R}}_{xr} \underline{\mathbf{w}} = \underline{\mathbf{w}}^T \mathbf{E}[\underline{\mathbf{x}\,\mathbf{r}}(n)\underline{\mathbf{x}\,\mathbf{r}}^T(n)]\ \underline{\mathbf{w}}$$

$$\underline{\mathbf{w}}^T \underline{\mathbf{R}}_{xr} \underline{\mathbf{w}} = \mathbf{E}\{[\underline{\mathbf{w}}^T \underline{\mathbf{x}\,\mathbf{r}}(n)][\underline{\mathbf{x}\,\mathbf{r}}^{\mathbf{T}}(n)\ \underline{\mathbf{w}}]\} \tag{3.50}$$

logo, tem-se:

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{R}}_{\mathrm{xr}} \underline{\mathbf{w}}\right)}{\partial \underline{\mathbf{w}}} = \frac{\partial\{\mathbf{E}[(\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{x}\,\mathbf{r}}(\mathrm{n}))(\underline{\mathbf{x}\,\mathbf{r}}^{\mathbf{T}}(\mathrm{n})\ \underline{\mathbf{w}})]\}}{\partial \underline{\mathbf{w}}} \tag{3.51.a}$$

Invertendo-se a ordem entre a esperança e a diferenciação por serem operações lineares, obtém-se:

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{R}}_{\mathrm{xr}} \underline{\mathbf{w}}\right)}{\partial \underline{\mathbf{w}}} = \mathbf{E}\left\{\frac{\partial[(\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{x}\,\mathbf{r}}(\mathrm{n}))(\underline{\mathbf{x}\,\mathbf{r}}^{\mathbf{T}}(\mathrm{n})\underline{\mathbf{w}})]}{\partial \underline{\mathbf{w}}}\right\} \tag{3.51.b}$$

Pode-se aplicar a regra da cadeia da diferenciação na Eq.(3.51.b), observando-se que $\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{x}\,\mathbf{r}}(\mathrm{n})$ e $\underline{\mathbf{x}\,\mathbf{r}}^{\mathbf{T}}(\mathrm{n})\underline{\mathbf{w}}$ são escalares, portanto, tem-se:

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{R}}_{\mathrm{xr}} \underline{\mathbf{w}}\right)}{\partial \underline{\mathbf{w}}} = \mathbf{E}\left\{\frac{\partial(\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{x}\,\mathbf{r}}(\mathrm{n}))}{\partial \underline{\mathbf{w}}}(\underline{\mathbf{x}\,\mathbf{r}}^{\mathrm{T}}(\mathrm{n})\ \underline{\mathbf{w}}) + (\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{x}\,\mathbf{r}}(\mathrm{n}))\frac{\partial(\underline{\mathbf{x}\,\mathbf{r}}^{\mathrm{T}}(\mathrm{n})\ \underline{\mathbf{w}})}{\partial \underline{\mathbf{w}}}\right\}$$

Além disso, como $\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{x}\,\mathbf{r}}(\mathrm{n}) = \underline{\mathbf{x}\,\mathbf{r}}^{\mathbf{T}}(\mathrm{n})\ \underline{\mathbf{w}}$,

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{R}}_{\mathrm{xr}} \underline{\mathbf{w}}\right)}{\partial \underline{\mathbf{w}}} = 2\,\mathbf{E}\left\{\frac{\partial(\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{x}\,\mathbf{r}}(\mathrm{n}))}{\partial \underline{\mathbf{w}}}(\underline{\mathbf{x}\,\mathbf{r}}^{\mathbf{T}}(\mathrm{n})\ \underline{\mathbf{w}})\right\} \tag{3.52}$$

Por analogia ao resultado da Eq.(3.47.b), tem-se que:

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\,\mathbf{R}_{\mathrm{xr}}\,\underline{\mathbf{w}}\right)}{\partial\underline{\mathbf{w}}} = 2\,\mathbf{E}\left\{\underline{\mathbf{x}\,\mathbf{r}}(\mathrm{n})(\underline{\mathbf{x}\,\mathbf{r}}^{\mathrm{T}}(\mathrm{n})\ \underline{\mathbf{w}})\right\}$$

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\underline{\mathbf{R}}_{\mathrm{xr}}\,\underline{\mathbf{w}}\right)}{\partial\underline{\mathbf{w}}} = 2\underline{\mathbf{R}}_{\mathrm{xr}}\,\underline{\mathbf{w}} \tag{3.53}$$

d) Para se calcular a derivada

$$\frac{\partial\left(\underline{\mathbf{ww}}^{T}\,\underline{\mathbf{R}}_{r}\,\underline{\mathbf{ww}}\right)}{\partial\,\mathbf{ww}}$$

procede-se de forma análoga ao caso c), resultando em:

$$\frac{\partial\left(\underline{\mathbf{ww}}^{\mathrm{T}}\,\underline{\mathbf{R}}_{\mathrm{r}}\,\underline{\mathbf{ww}}\right)}{\partial\underline{\mathbf{ww}}} = 2\underline{\mathbf{R}}_{\mathrm{r}}\,\underline{\mathbf{ww}} \tag{3.54}$$

e) Para se calcular a derivada

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\,\underline{\mathbf{R}}_{\mathrm{xr\ r}}\,\underline{\mathbf{ww}}\right)}{\partial\underline{\mathbf{w}}}$$

Utilizando-se as Eqs.(3.34.a) e a (3.34.b), tem-se:

$$\underline{\mathbf{w}}^{\mathrm{T}}\,\underline{\mathbf{R}}_{\mathrm{xr\ r}}\,\underline{\mathbf{ww}} = \underline{\mathbf{w}}^{\mathrm{T}}\mathbf{E}[\underline{\mathbf{xr}}(n)\underline{\mathbf{r}}^{\mathrm{T}}(n)]\underline{\mathbf{ww}} \tag{3.55}$$

logo, tem-se:

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\underline{\mathbf{R}}_{\mathrm{xr\ r}}\underline{\mathbf{ww}}\right)}{\partial\underline{\mathbf{w}}}=\frac{\partial\{\mathbf{E}[(\underline{\mathbf{w}}^{\mathrm{T}}\,\underline{\mathbf{x}}\,\underline{\mathbf{r}}(\mathrm{n}))(\underline{\mathbf{r}}^{\mathbf{T}}(\mathrm{n})\ \underline{\mathbf{ww}})]\}}{\partial\underline{\mathbf{w}}} \tag{3.56}$$

Invertendo-se a ordem entre a esperança e a diferenciação por serem operações lineares, obtém-se:

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\underline{\mathbf{R}}_{\mathrm{xr\ r}}\underline{\mathbf{ww}}\right)}{\partial\underline{\mathbf{w}}}=\frac{\mathbf{E}\{\partial[(\underline{\mathbf{w}}^{\mathrm{T}}\,\underline{\mathbf{x}}\,\underline{\mathbf{r}}(\mathrm{n}))(\underline{\mathbf{r}}^{\mathbf{T}}(\mathrm{n})\ \underline{\mathbf{ww}})]\}}{\partial\underline{\mathbf{w}}} \tag{3.57}$$

Aplica-se a regra da cadeia da diferenciação na Eq.(3.56) e observando-se que $\underline{\mathbf{w}}^{\mathrm{T}}\,\underline{\mathbf{x}}\,\underline{\mathbf{r}}(\mathrm{n})$ e $\underline{\mathbf{r}}^{\mathbf{T}}(\mathrm{n})\underline{\mathbf{ww}}$ são escalares, obtem-se:

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\underline{\mathbf{R}}_{\mathrm{xr\ r}}\underline{\mathbf{ww}}\right)}{\partial\underline{\mathbf{w}}}=\mathbf{E}\left\{\frac{\partial(\underline{\mathbf{w}}^{\mathrm{T}}\,\underline{\mathbf{x}}\,\underline{\mathbf{r}}(\mathrm{n}))}{\partial\underline{\mathbf{w}}}(\underline{\mathbf{r}}^{\mathbf{T}}(\mathrm{n})\ \underline{\mathbf{ww}})+(\underline{\mathbf{w}}^{\mathrm{T}}\,\underline{\mathbf{x}}\,\underline{\mathbf{r}}(\mathrm{n}))\frac{\partial(\underline{\mathbf{r}}^{\mathbf{T}}(\mathrm{n})\ \underline{\mathbf{ww}})}{\partial\underline{\mathbf{w}}}\right\} \tag{3.58}$$

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\underline{\mathbf{R}}_{\mathrm{xr\ r}}\underline{\mathbf{ww}}\right)}{\partial\underline{\mathbf{w}}}=\mathbf{E}\left\{\underline{\mathbf{x}}\,\underline{\mathbf{r}}(\mathrm{n})\underline{\mathbf{r}}^{\mathbf{T}}(\mathrm{n})\ \underline{\mathbf{ww}}\right\} \tag{3.59}$$

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\underline{\mathbf{R}}_{\mathrm{xr\ r}}\underline{\mathbf{ww}}\right)}{\partial\underline{\mathbf{w}}}=\underline{\mathbf{R}}_{\mathrm{xr\ r}}\underline{\mathbf{ww}} \tag{3.60}$$

f) Para se calcular a derivada

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}}\underline{\mathbf{R}}_{\mathrm{xr\ r}}\underline{\mathbf{ww}}\right)}{\partial\underline{\mathbf{ww}}}$$

Utilizando-se as Eqs.(3.34.a) e a (3.34.b), tem-se:

$$\mathbf{E}[y(n)yy(n)]=\mathbf{E}[yy(n)y(n)]=\mathbf{E}[\underline{\mathbf{w}}^{T}\,\underline{\mathbf{xr}}(\mathrm{n})\ \underline{\mathbf{r}}^{T}(n)\underline{\mathbf{ww}}]=\mathbf{E}[\underline{\mathbf{ww}}^{T}\,\underline{\mathbf{r}}(\mathrm{n})\ \underline{\mathbf{xr}}^{T}(n)\underline{\mathbf{w}}] \tag{3.61}$$

$$\mathbf{E}[y(n)yy(n)] = \underline{\mathbf{w}}^T \underline{\mathbf{R}}_{xr\ r} \underline{\mathbf{ww}} = \underline{\mathbf{ww}}^T \underline{\mathbf{R}}_{r\ xr} \underline{\mathbf{w}} \tag{3.62}$$

então,

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{R}}_{\mathrm{xr\ r}} \underline{\mathbf{ww}}\right)}{\partial \underline{\mathbf{ww}}} = \frac{\partial\left(\underline{\mathbf{ww}}^{\mathrm{T}} \underline{\mathbf{R}}_{\mathrm{r\ xr}} \underline{\mathbf{w}}\right)}{\partial \underline{\mathbf{ww}}} \tag{3.63}$$

Por analogia com e), tem-se que:

$$\frac{\partial\left(\underline{\mathbf{w}}^{\mathrm{T}} \underline{\mathbf{R}}_{\mathrm{xr\ r}} \underline{\mathbf{ww}}\right)}{\partial \underline{\mathbf{ww}}} = \underline{\mathbf{R}}_{\mathrm{r\ xr}} \underline{\mathbf{w}} \tag{3.64}$$

Substituindo-se as Eqs.(3.47.b), (3.53) e (3.60) na Eq.(3.44.a) e substituindo-se as Eqs.(3.48.b), (3.54) e (3.64) na Eq.(3.44.b), obtêm-se respectivamente dois conjuntos de equações conforme apresentado a seguir:

$$\frac{\partial \mathrm{J}}{\partial \underline{\mathbf{w}}} = -2\underline{\mathbf{p}} + 2\underline{\mathbf{R}}_{\mathrm{xr}} \underline{\mathbf{w}} + 2\underline{\mathbf{R}}_{\mathrm{xr\ r}} \underline{\mathbf{ww}} = \underline{\mathbf{0}} \tag{3.65.a}$$

$$\frac{\partial \mathrm{J}}{\partial \underline{\mathbf{ww}}} = -2\underline{\mathbf{pp}} + 2\underline{\mathbf{R}}_{\mathrm{r}} \underline{\mathbf{ww}} + 2\underline{\mathbf{R}}_{\mathrm{r\ xr}} \underline{\mathbf{w}} = \underline{\mathbf{0}} \tag{3.65.b}$$

ou ainda,

$$\underline{\mathbf{R}}_{\mathrm{xr}} \underline{\mathbf{w}} + \underline{\mathbf{R}}_{\mathrm{xr\ r}} \underline{\mathbf{ww}} = \underline{\mathbf{p}} \tag{3.66.a}$$

$$\underline{\mathbf{R}}_{\mathrm{r}} \underline{\mathbf{ww}} + \underline{\mathbf{R}}_{\mathrm{r\ xr}} \underline{\mathbf{w}} = \underline{\mathbf{pp}} \tag{3.66.b}$$

As Eqs.(3.66.a) e (3.66.b) representam um sistema de equações lineares denominado de “sistema de equações D”, que pode ser expressa na seguinte forma:

$$\underline{\mathbf{A}}\underline{\mathbf{x}}=\underline{\mathbf{b}} \tag{3.67.a}$$

Admitindo-se que a matriz $\underline{\mathbf{A}}$ seja não singular, tem-se que a solução da Eq.(3.67.a) é fornecida por:

$$\underline{\mathbf{x}}=\underline{\mathbf{A}}^{-1}\underline{\mathbf{b}} \tag{3.67.b}$$

O vetor $\underline{\mathbf{x}}$ representa as incógnitas do sistema, sendo formado por elementos dos vetores $\underline{\mathbf{w}}$ e $\underline{\mathbf{ww}}$. O vetor $\underline{\mathbf{b}}$ é composto de elementos presentes nos vetores $\underline{\mathbf{p}}$ e $\underline{\mathbf{pp}}$. Finalmente, a matriz $\underline{\mathbf{A}}$ é formada por elementos das matrizes $\underline{\mathbf{R}}_{\mathrm{xr}}$, $\underline{\mathbf{R}}_{\mathrm{r}}$, $\underline{\mathbf{R}}_{\mathrm{xr\,r}}$ e $\underline{\mathbf{R}}_{\mathrm{r\,xr}}$.

Algumas observações a respeito da dimensão do sistema gerado podem ser feitas com relação as Eqs.(3.66.a) e (3.66.b). Considerando que se utilize um valor de $N=1$, não seria possível decompor o sinal através do procedimento de pré-rotação expresso pelas Eqs.(3.16.a) e (3.18.a), de modo que, adotando-se um conjunto de valores nulos para o vetor $\underline{\mathbf{xr}}(n)$ conforme a Eq.(3.18.a) o sistema de equações ficaria reduzido ao sistema dado pela Eq.(2.55.b), isto é, o sistema de Wiener. Considerando que se utilize um valor de $N=2$, seria possível decompor o sinal através do procedimento de pré-rotação expresso pelas Eqs.(3.16.a) e (3.18.a) porém, os valores gerados para o vetor $\underline{\mathbf{xr}}(n)$ seriam todos nulos, de forma que o sistema de equações ficaria novamente reduzido ao sistema dado pela Eq.(2.55.b), ou seja, o sistema de Wiener, pois:

-Na Eq.(3.66.b) $\underline{\mathbf{R}}_{\mathrm{r\,xr}}$ seria uma matriz nula, logo, $\underline{\mathbf{R}}_{\mathrm{r}}=\underline{\mathbf{R}}_{\mathrm{x}}$ e $\underline{\mathbf{pp}}=\underline{\mathbf{r}}_{\mathrm{xd}}$, portanto, o sistema reduzir-se-ia a $\underline{\mathbf{R}}_{\mathrm{x}}\underline{\mathbf{ww}}=\underline{\mathbf{r}}_{\mathrm{xd}}$.

-Não existiria a Eq.(3.66.a) pelo fato de $\underline{\mathbf{R}}_{\mathrm{xr}}=\underline{\mathbf{R}}_{\mathrm{xr\,r}}$ serem matrizes nulas e $\underline{\mathbf{p}}$ ser um vetor coluna nulo.

Utilizando-se um valor de $N=3$, é possível decompor o sinal através do procedimento de pré-rotação e os valores gerados para o vetor $\underline{\mathbf{xr}}(n)$ conforme a Eq.(3.18.b) seriam todos nulos, exceto $xr_1(n)$, de forma que o sistema de equações não mais coincide com o sistema de Wiener, cuja matriz $\underline{\mathbf{R}}_{\mathrm{x}}$ possui ordem 3. Posto que a ordem das matrizes $\underline{\mathbf{R}}_{\mathrm{xr}}$, $\underline{\mathbf{R}}_{\mathrm{r}}$, $\underline{\mathbf{R}}_{\mathrm{xr\,r}}$ e $\underline{\mathbf{R}}_{\mathrm{r\,xr}}$ é 3 por 3, então a dimensão do sistema de equações gerado pode atingir 4, dependendo da utilização ou não de todos os sinais decompostos não nulos. De forma mais explícita, considere o sistema $\underline{\mathbf{A}}\underline{\mathbf{x}}=\underline{\mathbf{b}}$, N = 3 e a utilização de todas as entradas não nulas decompostas. Baseando-se nas Eqs.(3.66.a) e (3.66.b) o sistema de equações é composto por:

$$\begin{array}{ccccccccc} R_{xr_1\,r_0}ww_0 & + & R_{xr_1\,r_1}ww_1 & + & R_{xr_1\,r_2}ww_2 & + & R_{xr_1xr_1}w_1 & = & p_1 \\ R_{r_0r_0}ww_0 & + & R_{r_0r_1}ww_1 & + & R_{r_0r_2}ww_2 & + & R_{r_0\,xr_1}w_1 & = & pp_0 \\ R_{r_1r_0}ww_0 & + & R_{r_1r_1}ww_1 & + & R_{r_1r_2}ww_2 & + & R_{r_1\,xr_1}w_1 & = & pp_1 \\ R_{r_2r_0}ww_0 & + & R_{r_2r_1}ww_1 & + & R_{r_2r_2}ww_2 & + & R_{r_2\,xr_1}w_1 & = & pp_2 \end{array} \tag{3.68}$$

Na forma matricial o sistema é descrito por:

$$\begin{bmatrix} R_{r_0r_0} & R_{r_0r_1} & R_{r_0r_2} & R_{r_0\,xr_1} \\ R_{r_1r_0} & R_{r_1r_1} & R_{r_1r_2} & R_{r_1\,xr_1} \\ R_{r_2r_0} & R_{r_2r_1} & R_{r_2r_2} & R_{r_2\,xr_1} \\ R_{xr_1\,r_0} & R_{xr_1\,r_1} & R_{xr_1\,r_2} & R_{xr_1xr_1} \end{bmatrix} \begin{bmatrix} ww_0 \\ ww_1 \\ ww_2 \\ w_1 \end{bmatrix} = \begin{bmatrix} pp_0 \\ pp_1 \\ pp_2 \\ p_1 \end{bmatrix} \tag{3.69}$$

Comparando-se a Eq.(3.67.a) com a Eq.(3.69), tem-se que:

$$\underline{\mathbf{A}} = \begin{bmatrix} \mathrm{R}_{\mathrm{r_0r_0}} & \mathrm{R}_{\mathrm{r_0r_1}} & \mathrm{R}_{\mathrm{r_0r_2}} & \mathrm{R}_{\mathrm{r_0\,xr_1}} \\ \mathrm{R}_{\mathrm{r_1r_0}} & \mathrm{R}_{\mathrm{r_1r_1}} & \mathrm{R}_{\mathrm{r_1r_2}} & \mathrm{R}_{\mathrm{r_1\,xr_1}} \\ \mathrm{R}_{\mathrm{r_2r_0}} & \mathrm{R}_{\mathrm{r_2r_1}} & \mathrm{R}_{\mathrm{r_2r_2}} & \mathrm{R}_{\mathrm{r_2\,xr_1}} \\ \mathrm{R}_{\mathrm{xr_1\,r_0}} & \mathrm{R}_{\mathrm{xr_1\,r_1}} & \mathrm{R}_{\mathrm{xr_1\,r_2}} & \mathrm{R}_{\mathrm{xr_1xr_1}} \end{bmatrix} \tag{3.70.a}$$

$$\underline{\mathbf{x}} = \begin{bmatrix} \mathrm{ww}_0 \\ \mathrm{ww}_1 \\ \mathrm{ww}_2 \\ \mathrm{w}_1 \end{bmatrix} \tag{3.70.b}$$

$$\underline{\mathbf{b}} = \begin{bmatrix} \mathrm{pp}_0 \\ \mathrm{pp}_1 \\ \mathrm{pp}_2 \\ \mathrm{p}_1 \end{bmatrix} \tag{3.70.c}$$

Pode-se ainda representar esse sistema utilizando-se sub-matrizes, logo, tem-se:

$$\underline{\mathbf{A}} = \begin{bmatrix} \underline{\mathbf{R}}_{r} & \vdots & \underline{\mathbf{R}}_{r\ xr}(:,2) \\ \cdots\ \cdots\ \cdots & \cdots\vdots\cdots & \cdots\ \cdots\ \cdots \\ \underline{\mathbf{R}}_{xr\ r}(2,:) & \vdots & \underline{\mathbf{R}}_{xr}(2,2) \end{bmatrix} \tag{3.70.d}$$

$$\underline{\mathbf{x}} = \begin{bmatrix} \underline{\mathbf{ww}} \\ \cdots\ \cdots \\ \underline{\mathbf{w}}(2) \end{bmatrix} \tag{3.70.e}$$

$$\underline{\mathbf{b}} = \begin{bmatrix} \underline{\mathbf{pp}} \\ \cdots\ \cdots \\ \underline{\mathbf{p}}(2) \end{bmatrix} \tag{3.70.f}$$

onde,

$\underline{\mathbf{R}}_{xr\ r}(2,:)$ é o vetor linha composto pela segunda linha da matriz $\underline{\mathbf{R}}_{xr\ r}$, conforme a Eq.(3.36) .

$\underline{\mathbf{R}}_{r}$ é a matriz definida pela Eq.(3.33).

$\underline{\mathbf{R}}_{xr}(2,2)$ é o elemento da segunda linha e segunda coluna da matriz $\underline{\mathbf{R}}_{xr}$, conforme a Eq.(3.29).

$\underline{\mathbf{R}}_{r\ xr}(:,2)$ é o vetor coluna composto pela segunda coluna da matriz $\underline{\mathbf{R}}_{r\ xr}$,conforme a Eq.(3.38).

$\underline{\mathbf{ww}}$ é o vetor coluna expresso pela Eq.(3.10.b).

$\underline{\mathbf{w}}(2) = w_1$ é o segundo elemento do vetor coluna $\underline{\mathbf{w}}$ conforme a Eq.(3.10.a).

$\underline{\mathbf{p}}(2) = p_1$ é o segundo elemento do vetor coluna $\underline{\mathbf{p}}$ conforme a Eq.(3.25.b).

$\underline{\mathbf{pp}}$ é o vetor coluna expresso pela Eq.(3.27.b).

Como mencionado anteriormente, para N=3 pôde-se formar um sistema com 4 equações e 4 incógnitas. No entanto, pode-se também ter um sistema com 3 equações e três incógnitas ou até mesmo menor. O parâmetro que balizará se as soluções desses sistemas, caso existam, são boas ou não, será dado pela avaliação comparativa entre a MSE gerada utilizando-se a solução de Wiener e utilizando-se a solução das Eqs.(3.66.a) e (3.66.b). Dado um número de amostras, os parâmetros dessas equações podem ser calculados usando-se estimativas das esperanças envolvidas.

Considerando-se ainda N=3, um novo sistema de equações lineares poderia ser formado após a decomposição do vetor $\underline{\mathbf{x}}(n)$. Baseando-se nas Eqs.(3.66.a) e (3.66.b) e escolhendo-se um conjunto menor de sinais, o sistema $\underline{\mathbf{A}}\underline{\mathbf{x}}=\underline{\mathbf{b}}$ seria composto por:

$$\begin{array}{ccccccc} R_{r_0r_0}ww_0 & + & R_{r_0r_1}ww_1 & + & R_{r_0r_2}ww_2 & = & pp_0 \\ R_{r_1r_0}ww_0 & + & R_{r_1r_1}ww_1 & + & R_{r_1r_2}ww_2 & = & pp_1 \\ R_{r_2r_0}ww_0 & + & R_{r_2r_1}ww_1 & + & R_{r_2r_2}ww_2 & = & pp_2 \end{array} \tag{3.71}$$

Na forma matricial, tem-se:

$$\underline{\mathbf{R}}_r\,\underline{\mathbf{ww}} = \underline{\mathbf{pp}} \tag{3.72}$$

Comparando-se a Eq.(3.67.a) com a Eq.(3.72), obtém-se:

$$\underline{\mathbf{A}} = \underline{\mathbf{R}}_r \tag{3.73.a}$$

$$\underline{\mathbf{x}} = \underline{\mathbf{ww}} \tag{3.73.b}$$

$$\underline{\mathbf{b}} = \underline{\mathbf{pp}} \tag{3.73.c}$$

Adotando-se um valor de $N = 4$, é possível decompor o sinal através do procedimento de pré-rotação expresso pela Eq.(3.16.a) e os valores gerados para o vetor $\underline{\mathbf{xr}}(n)$ Eq.(3.18.a) seriam todos nulos, exceto $xr_1(n)$ e $xr_2(n)$. Posto que a ordem das matrizes $\underline{\mathbf{R}}_{\mathrm{xr}}$, $\underline{\mathbf{R}}_{\mathrm{r}}$, $\underline{\mathbf{R}}_{\mathrm{xr\ r}}$ e $\underline{\mathbf{R}}_{\mathrm{r\ xr}}$ é 4 por 4, então a dimensão do sistema de equações gerado pode atingir 6, dependendo da utilização ou não de todos os sinais decompostos. Dessa forma, tem-se uma certa flexibilidade na determinação da ordem das matrizes que formarão o sistema de equações relacionado ao problema da filtragem linear. A única possibilidade de se ter um sistema de equações coincidente com o sistema de Wiener é para o caso trivial em que N é igual a 1 ou 2, conforme explicado anteriormente.

De forma análoga ao exemplo anterior, considere o sistema $\underline{\mathbf{A}}\underline{\mathbf{x}}=\underline{\mathbf{b}}$, N = 4 e a utilização de todas as entradas não nulas decompostas. Baseando-se nas Eqs.(3.66.a) e (3.66.b) tem-se o seguinte sistema de equações:

$$
\begin{array}{lllllllllllll}
R_{xr_1\,r_0}ww_0 & + & R_{xr_1\,r_1}ww_1 & + & R_{xr_1\,r_2}ww_2 & + & R_{xr_1\,r_3}ww_3 & + & R_{xr_1xr_1}w_1 & + & R_{xr_1xr_2}w_2 & = & p_1 \\
R_{xr_2\,r_0}ww_0 & + & R_{xr_2\,r_1}ww_1 & + & R_{xr_2\,r_2}ww_2 & + & R_{xr_2\,r_3}ww_3 & + & R_{xr_2xr_1}w_1 & + & R_{xr_2xr_2}w_2 & = & p_2 \\
R_{r_0\,r_0}ww_0 & + & R_{r_0\,r_1}ww_1 & + & R_{r_0\,r_2}ww_2 & + & R_{r_0\,r_3}ww_3 & + & R_{r_0xr_1}w_1 & + & R_{r_0xr_2}w_2 & = & pp_0 \\
R_{r_1\,r_0}ww_0 & + & R_{r_1\,r_1}ww_1 & + & R_{r_1\,r_2}ww_2 & + & R_{r_1\,r_3}ww_3 & + & R_{r_1xr_1}w_1 & + & R_{r_1xr_2}w_2 & = & pp_1 \\
R_{r_2\,r_0}ww_0 & + & R_{r_2\,r_1}ww_1 & + & R_{r_2\,r_2}ww_2 & + & R_{r_2\,r_3}ww_3 & + & R_{r_2xr_1}w_1 & + & R_{r_2xr_2}w_2 & = & pp_2 \\
R_{r_3\,r_0}ww_0 & + & R_{r_3\,r_1}ww_1 & + & R_{r_3\,r_2}ww_2 & + & R_{r_3\,r_3}ww_3 & + & R_{r_3xr_1}w_1 & + & R_{r_3xr_2}w_2 & = & pp_3
\end{array}
\tag{3.74}
$$

Na forma matricial o sistema é descrito por:

$$
\begin{bmatrix}
R_{r_0r_0} & R_{r_0r_1} & R_{r_0r_2} & R_{r_0r_3} & R_{r_0\,xr_1} & R_{r_0\,xr_2} \\
R_{r_1r_0} & R_{r_1r_1} & R_{r_1r_2} & R_{r_1r_3} & R_{r_1\,xr_1} & R_{r_1\,xr_2} \\
R_{r_2r_0} & R_{r_2r_1} & R_{r_2r_2} & R_{r_2r_3} & R_{r_2\,xr_1} & R_{r_2\,xr_2} \\
R_{r_3r_0} & R_{r_3r_1} & R_{r_3r_2} & R_{r_3r_3} & R_{r_3\,xr_1} & R_{r_3\,xr_2} \\
R_{xr_1\,r_0} & R_{xr_1\,r_1} & R_{xr_1\,r_2} & R_{xr_1\,r_3} & R_{xr_1\,xr_1} & R_{xr_1\,xr_2} \\
R_{xr_2\,r_0} & R_{xr_2\,r_1} & R_{xr_2\,r_2} & R_{xr_2\,r_3} & R_{xr_2\,xr_1} & R_{xr_2\,xr_2}
\end{bmatrix}
\begin{bmatrix} ww_0 \\ ww_1 \\ ww_2 \\ ww_3 \\ w_1 \\ w_2 \end{bmatrix}
=
\begin{bmatrix} pp_0 \\ pp_1 \\ pp_2 \\ pp_3 \\ p_1 \\ p_2 \end{bmatrix}
\tag{3.75}
$$

Comparando-se a Eq.(3.67.a) com a Eq.(3.75), tem-se que:

$$\underline{\mathbf{A}} = \begin{bmatrix} R_{r_0 r_0} & R_{r_0 r_1} & R_{r_0 r_2} & R_{r_0 r_3} & R_{r_0\ xr_1} & R_{r_0\ xr_2} \\ R_{r_1 r_0} & R_{r_1 r_1} & R_{r_1 r_2} & R_{r_1 r_3} & R_{r_1\ xr_1} & R_{r_1\ xr_2} \\ R_{r_2 r_0} & R_{r_2 r_1} & R_{r_2 r_2} & R_{r_2 r_3} & R_{r_2\ xr_1} & R_{r_2\ xr_2} \\ R_{r_3 r_0} & R_{r_3 r_1} & R_{r_3 r_2} & R_{r_3 r_3} & R_{r_3\ xr_1} & R_{r_3\ xr_2} \\ R_{xr_1\ r_0} & R_{xr_1\ r_1} & R_{xr_1\ r_2} & R_{xr_1\ r_3} & R_{xr_1\ xr_1} & R_{xr_1\ xr_2} \\ R_{xr_2\ r_0} & R_{xr_2\ r_1} & R_{xr_2\ r_2} & R_{xr_2\ r_3} & R_{xr_2\ xr_1} & R_{xr_2\ xr_2} \end{bmatrix} \tag{3.76.a}$$

$$\underline{\mathbf{x}} = \begin{bmatrix} ww_0 \\ ww_1 \\ ww_2 \\ ww_3 \\ w_1 \\ w_2 \end{bmatrix} \tag{3.76.b}$$

$$\underline{\mathbf{b}} = \begin{bmatrix} pp_0 \\ pp_1 \\ pp_2 \\ pp_3 \\ p_1 \\ p_2 \end{bmatrix} \tag{3.76.c}$$

O sistema linear apresentado pelas Eq.(3.75) e (3.76) é formado por seis equações e seis incógnitas, isto é, o maior sistema possível de ser formado considerando-se que N é igual a 4. No entanto, pode-se também ter um sistema com 5 equações e cinco incógnitas ou até mesmo menor, bastando para isso, eliminar uma ou mais entradas decompostas e por conseguinte a respectiva incógnita peso.

CAPÍTULO 4

Sistema de Equação Linear Baseado na Decomposição da Entrada, Aplicado em Predição Linear

4.1 Introdução

Os processos autoregressivos são muito importantes por serem capazes de modelar eventos aleatórios encontrados na prática, como por exemplo, sinais eletroencefalográficos, eletromiográficos ou segmentos estacionários de voz [30,40,54]. Para muitas simulações existentes a partir desse capítulo, o sinal de entrada $x(n)$ e o sinal desejado $d(n)$ definidos nas Figs.(2.1) e (3.3), serão funções amostras oriundas de processos aleatórios autoregressivos. Portanto, esse capítulo inicia-se definindo processos autoregressivos e sua relação com preditores progressivos lineares, cuja otimização resulta numa solução muito conhecida, que emprega as equações de Wiener-Hopf .

No Capítulo 3, obteve-se um sistema de equações lineares denominadas de sistema D, a partir da minimização do erro entre o sinal desejado $d(n)$ e múltiplas entradas, oriundas de um processo de decomposição do sinal de entrada original $x(n)$, Fig.(3.3). Essas equações podem ser empregadas na solução de problemas típicos em processamento digital de sinais, como por exemplo, predição linear. Uma série de situações envolvendo esse tópico será apresentada e as soluções provenientes do sistema de equações proposto serão avaliadas, tomando-se como referência a solução obtida pelas equações de Wiener-Hopf.

Conforme mencionado no final do Capítulo 3, diversas soluções podem ser obtidas para o sistema de equações D. Isso vai depender do número de entradas, da entrada decomposta envolvida no processo de minimização, e consequentemente, do sistema de equações gerado e em última instância, do menor valor de erro (MSE) associado a cada uma dessas soluções. Um outro aspecto interessante diz respeito a solução obtida quando o número de amostras utilizadas na estimação dos parâmetros estatísticos empregados no sistema de equações, é variável. Isso ocorre em sinais que exibem características estacionárias durante um intervalo de tempo (ou número de amostras fixada a taxa de amostragem) e deseja-se extrair características desses sinais utilizando-se as equações de Wiener-Hopf. Nesse sentido, pode-se fazer comparações entre as soluções obtidas por um e por outro método para diferentes números de amostras.

4.2 Inter-Relações Entre Processos Autoregressivos e Predição Linear em um Contexto de Filtragem de Wiener

Um processo aleatório {x(n)} é dito ser um processo autoregressivo de ordem “p” AR(p) se suas funções amostras (sinais variantes no tempo) $x_k(n)$ puderem ser geradas a partir da seguinte equação diferença [7,19,38,39,40,54,55]:

$$x_k(n) = -\sum_{i=1}^{p} a_i x_k(n-i) + \upsilon_k(n) \tag{4.1}$$

onde os a_i' s são os coeficientes do modelo AR.

$\upsilon_k(n)$ é uma função amostra de um processo aleatório ruído branco {ν(n)} com média nula, também denominada de processo inovação de {x(n)}.

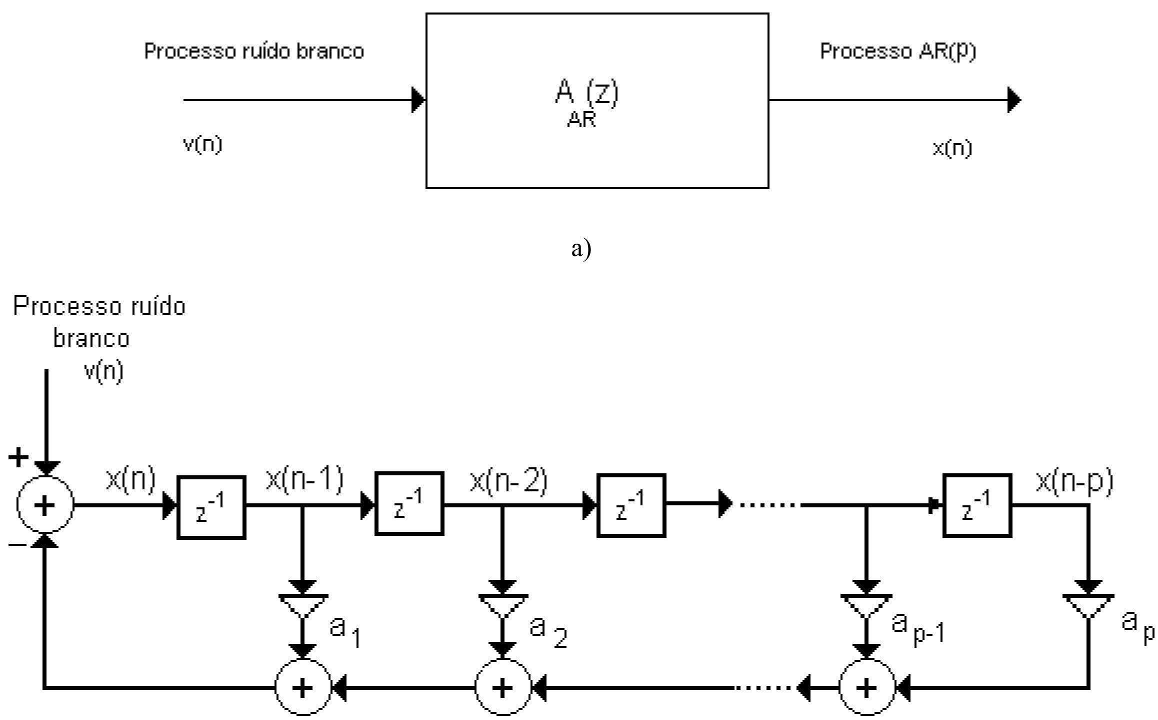
Nas Figs.(4.1.a) e (4.1.b) considerando-se que $\upsilon(n) = \upsilon_k(n)$ seja uma realização específica do processo {ν(n)} e $x(n) = x_k(n)$ ao correspondente sinal de resposta, logo, pertencente ao processo {x(n)}, então, a Eq.(4.1) expressa cada novo valor do sinal $x(n)$ como uma combinação linear de seus p valores mais recentes $x(n-1), x(n-2) \cdots x(n-p)$, adicionados à amostra atual da inovação $\upsilon(n)$, conforme apresenta a Fig.(4.1.b). A parcela de informação contida em $x(n)$ que está associada a $\upsilon(n)$, não possui qualquer correlação com amostras passadas de $x(n)$ ou mesmo das amostras passadas de $\upsilon(n)$, pelo fato dessa informação pertencer a um processo ruído branco.

A forma de gerar um processo autoregressivo AR(p) é filtrando-se um processo ruído branco através de um filtro IIR de ordem “p” e por esse motivo ele é também denominado de processo de somente pólos. Assim, para uma dada realização $\upsilon(n)$ de um processo ruído branco, obtém-se um respectivo sinal aleatório $x(n)$ na saída desse filtro. Uma função de transferência que relaciona esses dois sinais pode ser obtida através da aplicação da transformada Z na Eq.(4.1), logo:

$$X(z) = -\sum_{i=1}^{p} a_i X(z) z^{-i} + V(z)$$

$$A_{AR}(z) = \frac{X(z)}{V(z)} = \frac{1}{[1 + \sum_{i=1}^{p} a_i z^{-i}]} \tag{4.2}$$

Portanto, a função de transferência do sistema $A_{AR}(z)$ que gera o processo AR(p) é definida pela Eq.(4.2) que corresponde a um filtro IIR com "p" polos.



a)

b)

Fig.4.1.Geração do processo autoregressivo.

a)Geração de um processo AR, onde $A_{AR}(z)$ é um filtro IIR com "p" pólos.

b)Filtro IIR de "p" pólos utilizado na geração de um processo AR(p).

A predição do valor futuro $x(n)$ relativo a um processo aleatório discreto no tempo e estacionário, a partir da combinação linear das amostras passadas $x(n-1)$, $x(n-2) \cdots x(n-p)$ designa-se de predição linear progressiva de um passo Eq.(4.3.a) e (4.3.b) e especifica a estimação de uma amostra à frente em relação ao tempo $n-1$. De acordo com a Fig.(4.2.a), um preditor progressivo de um passo com ordem "p" [39,40,54,55] consiste de um filtro transversal $P^f(z)$ com "p" derivações, cada uma delas alimentadas pelas seguintes entradas: $x(n-1), x(n-2), \cdots x(n-p)$ e cujos coeficientes são definidos pelo vetor coluna $\underline{\mathbf{h}}^f$ Eq.(4.4), onde o sobreescrito "f" foi utilizado como referência a palavra "forward" significando um passo "à frente" ou "futuro".

$$\hat{x}(n) = \sum_{i=1}^{p} h_i^f x(n-i) \tag{4.3.a}$$

$$\hat{x}(n) = \underline{\mathbf{h}}^{f^T} \underline{\mathbf{x}}(n-1) \tag{4.3.b}$$

onde

$$\underline{\mathbf{x}}(n-1) = [x(n-1) \quad x(n-2) \quad \cdots \quad x(n-p)]^T$$

$$\underline{\mathbf{h}}^f = [h_1^f \quad h_2^f \quad \cdots \quad h_p^f]^T \tag{4.4}$$

A Eq.(4.1) refere-se a um processo AR(p) e mostra que cada nova amostra do sinal $x(n)$ é função apenas das "p" amostras passadas mais recentes, adicionadas a uma parcela de informação relativa ao processo de inovação $\upsilon(n)$. Como $\upsilon(n)$ pertence a um processo ruído branco, ele não pode ser previsto a partir de valores passados de $x(n)$ ou mesmo de $\upsilon(n)$, então tem-se que o **melhor preditor linear** de $x(n)$ baseado nas "p" amostras mais recentes nada mais é do que a somatória existente na parte direita da Eq.(4.1),

$$\hat{x}(n) = -\sum_{i=1}^{p} a_i x(n-i) \tag{4.5}$$

Por comparação direta entre as Eqs(4.3.a) com a (4.5), tem-se que os coeficientes do preditor linear ótimo serão dados por:

$$h_i^f = -a_i \quad para \quad 1 \le i \le p \tag{4.6.a}$$

ou

$$\underline{\mathbf{h}}^f = [h_1^f \quad h_2^f \quad \cdots \quad h_p^f]^T = [-a_1 \quad -a_2 \quad \cdots \quad -a_p]^T \tag{4.6.b}$$

A fim de encontrar uma relação entre $\hat{x}(n)$ e $x(n)$ no domínio Z, aplica-se a transformada Z na Eq.(4.5),

$$\hat{X}(z) = -a_1 X(z) z^{-1} - a_2 X(z) z^{-2} - \ \cdots \ - a_{p-1} X(z) z^{-p+1} - a_p X(z) z^{-p}$$

$$\frac{\hat{X}(z)}{X(z)} = z^{-1}[-a_1 - a_2 z^{-1} - \ \cdots \ - a_{p-1} z^{-(p-2)} - a_p z^{-(p-1)}] \tag{4.7.a}$$

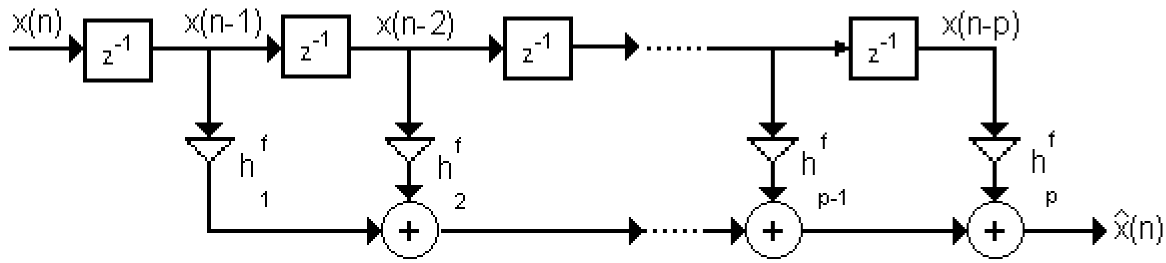
$$\frac{\hat{X}(z)}{X(z)} = z^{-1}[h_1^f + h_2^f z^{-1} + \ \cdots \ + h_{p-1}^f z^{-(p-2)} + h_p^f z^{-(p-1)}] \tag{4.7.b}$$

$$P^f(z) = z^{-1} W(z) \tag{4.7.c}$$

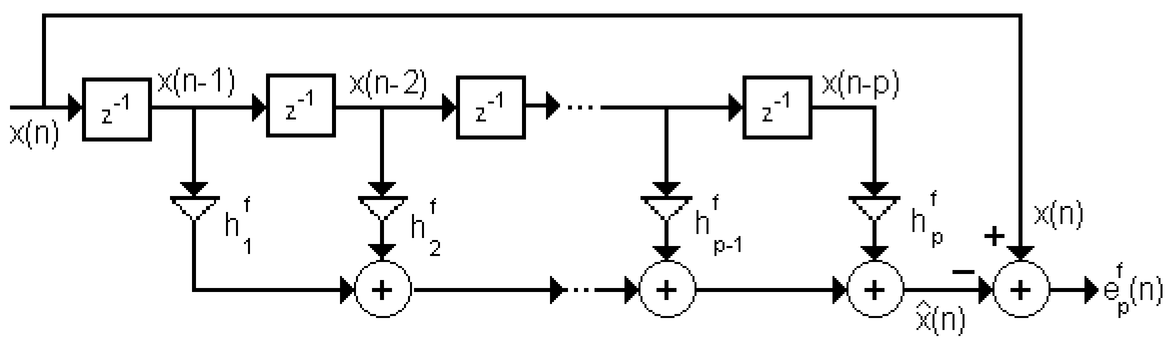
onde $P^f(z)$ representa a função de transferência do preditor ótimo Fig.(4.2.a) e $W(z)$ representa o filtro de Wiener, cujos coeficientes são dado pelos $h_i^f = -a_i$ [54,55].

A partir da definição do filtro preditor progressivo de ordem p, pode-se definir o filtro erro de predição progressivo $EP^f(z)$ Fig.(4.2.b), como sendo o filtro cuja saída é a diferença entre o sinal desejado, que nesse caso é o próprio sinal $x(n)$, e o sinal estimado $\hat{x}(n)$. A essa diferença denomina-se de erro de predição progressivo $e^f(n)$ dada por:

$$e^f(n) = x(n) - \hat{x}(n) \tag{4.8}$$



a)



b)

Fig.4.2. a) Filtro preditor progressivo de ordem "p", $P^f(z)$

b) Filtro erro de predição progressivo de ordem "p", $EP^f(z)$.

A função de transferência do filtro erro de predição ótimo $EP^f(z)$ pode ser obtida aplicando-se a transformada Z na Eq.(4.8), logo,

$$E^f(z) = X(z) - \hat{X}(z) \tag{4.9.a}$$

$$E^f(z) = X(z) - P^f X(z)$$

$$E^f(z) = X(z) - z^{-1}W(z)X(z)$$

$$E^f(z) = X(z)[1 - z^{-1}W(z)]$$

$$\frac{E^f(z)}{X(z)} = [1 + a_1 z^{-1} + a_2 z^{-2} + \ \cdots \ + a_{p-1} z^{-p+1} + a_p z^{-p}] \tag{4.9.b}$$

$$EP^f(z) = [1 + \sum_{i=1}^{p} a_i z^{-i}] \qquad (4.9.c)$$

Tem-se que a relação entre os coeficientes do filtro $EP^f(z)$ dado pelo vetor coluna $\underline{\mathbf{ep}}^f$, dos coeficientes do preditor $P^f(z)$ dado por $\underline{\mathbf{h}}^f$ e dos coeficientes do filtro $A_{AR}(z)$ é :

$$\underline{\mathbf{ep}}^f = [ep_0^f \quad ep_1^f \quad \cdots \quad ep_p^f]^T = [1 \quad a_1 \quad \cdots \quad a_p]^T = \begin{bmatrix} 1 \\ -\underline{\mathbf{h}}^f \end{bmatrix} \qquad (4.10)$$

Comparando-se a função de transferência do sistema $A_{AR}(z)$ Eq.(4.2) que gera o processo AR(p), com a função de transferência do filtro erro de predição $EP^f(z)$ Eq.(4.9.c), observa-se que os zeros da última são exatamente iguais aos polos da primeira. Isso significa que um filtro erro de predição de ordem p, excitado por um processo AR(p), produzirá em sua saída um sinal erro de predição $e^f(n)$ idêntico ao ruído branco $\upsilon(n)$. Por esse motivo esse filtro é denominado de filtro branqueador. Portanto, se um determinado sinal puder ser caracterizado como pertencente a um dado processo autoregressivo, então os parâmetros AR, isto é, os coeficientes do filtro IIR que geram esse processo, poderão ser estimados utilizando-se predição linear [54,55].

Com base no anteriormente exposto, a Fig.(4.3) apresenta de forma clara essa relação bastante estreita entre processos autoregressivos, predição linear e filtragem de Wiener. Ainda observando a Fig.(4.3), estruturando o problema da filtragem de Wiener em um contexto de predição linear, têm-se que os coeficientes obtidos para o filtro de Wiener $W(z)$ serão os próprios coeficientes do filtro preditor ótimo. Dessa forma, a otimização do preditor linear de ordem "p" é obtida minimizando o valor quadrático médio do erro de predição progressivo, logo:

$$J = \mathbf{E}[e^{f^2}(n)] \qquad (4.11)$$

Os coeficientes do filtro de predição que minimizam a Eq.(4.11) são calculados utilizando-se as

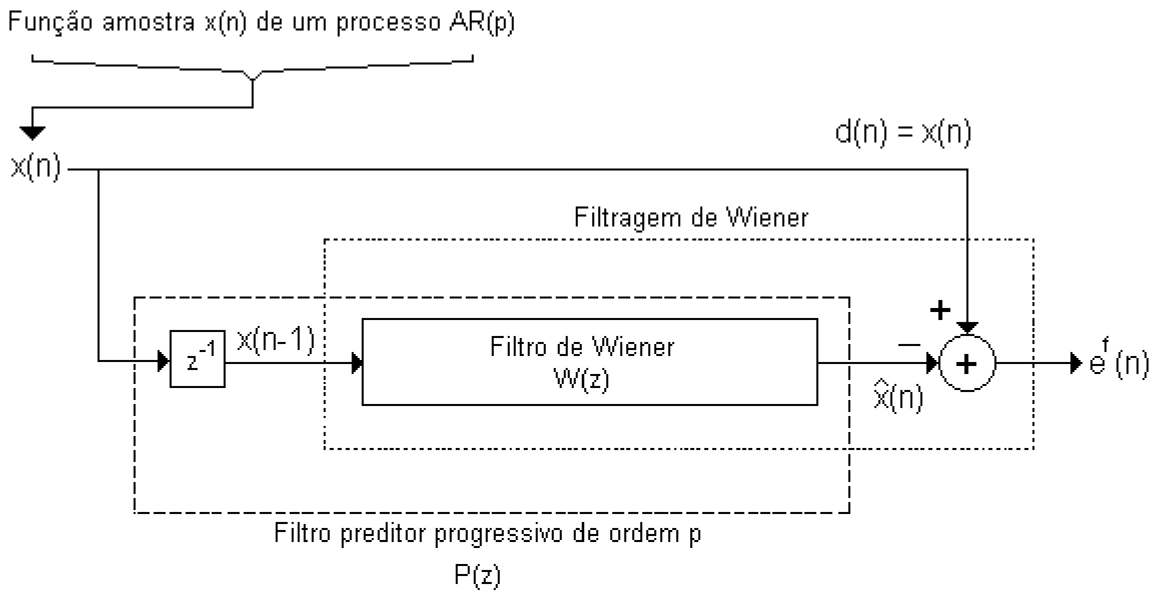


Fig.4.3. Relação entre o filtro preditor progressivo de uma amostra e ordem "p" e o filtro erro de predição progressivo em um contexto de filtragem de Wiener, onde o sinal desejado $d(n) = x(n)$ e o sinal de entrada $x(n-1)$ são funções amostras de um processo AR(p).

equações de Wiener-Hopft. A equação da matriz de autocorrelação $\underline{\mathbf{R}}_{\mathbf{x}}$ permanece inalterada, sendo fornecida pela Eq.(2.42),

$$\mathbf{E}[\underline{\mathbf{x}}(n\text{-}1)\underline{\mathbf{x}}^{\mathrm{T}}(n\text{-}1)]=\mathbf{E}[\underline{\mathbf{x}}(n)\underline{\mathbf{x}}^{\mathrm{T}}(n)]=\underline{\mathbf{R}}_{\mathbf{x}}=\begin{bmatrix} r_x(0) & r_x(1) & \dots & r_x(p-1) \\ r_x(1) & r_x(0) & \cdots & r_x(p-2) \\ \vdots & \vdots & \ddots & \vdots \\ r_x(p-1) & r_x(p-2) & \cdots & r_x(0) \end{bmatrix} \tag{4.12}$$

onde

$$\underline{\mathbf{x}}(n-1) = [x(n-1) \quad x(n-2) \quad \cdots \quad x(n-p)]^T \tag{4.13.a}$$

$$\underline{\mathbf{x}}(n) = [x(n) \quad x(n-1) \quad \cdots \quad x(n-p+1)]^T \tag{4.13.b}$$

A Eq.(4.12) é verdadeira pelo fato do processo aleatório $\mathbf{x}(n)$ ser estacionário. O vetor coluna, correlação cruzada entre a entrada e a resposta desejada, é alterado em relação Eq.(2.38.a) para:

$$\underline{\mathbf{r}}_{dx} = \mathbf{E}[d(n)\underline{\mathbf{x}}(n-1)] \tag{4.14.a}$$

$$\underline{\mathbf{r}}_{dx} = \mathbf{E}[x(n)\underline{\mathbf{x}}(n-1)] \tag{4.14.b}$$

$$\underline{\mathbf{r}}_{dx} = \mathbf{E}\begin{bmatrix} x(n)x(n-1) \\ x(n)x(n-2) \\ \vdots \\ x(n)x(n-p) \end{bmatrix} = \begin{bmatrix} r_x(1) \\ r_x(2) \\ \vdots \\ r_x(p) \end{bmatrix} \tag{4.14.c}$$

e portanto,

$$\underline{\mathbf{R}}_{\mathbf{x}}\underline{\mathbf{h}}^{f} = \underline{\mathbf{r}}_{\mathbf{dx}} \tag{4.15.a}$$

$$\underline{\mathbf{h}}^{f} = \underline{\mathbf{R}}_{\mathbf{x}}^{-1}\underline{\mathbf{r}}_{\mathbf{dx}} \tag{4.15.b}$$

que são iguais aos coeficientes ótimos do preditor linear progressivo de ordem “p”, fornecidos anteriormente pela Eq.(4.5) via uma justificativa plausível. Adequando-se a Eq.(2.59.b) para o presente caso, tem-se que o valor mínimo de J é :

$$J(\underline{\mathbf{h}}^{f}) = \sigma_d^2 - \sum_{i=1}^{p} h_i^f r_{xd}(i) \tag{4.16.a}$$

$$J_{Min} = J(\underline{\mathbf{h}}^{f}) = \sigma_d^2 - \underline{\mathbf{h}}^{f^T}\underline{\mathbf{r}}_{xd} = R_x(0) - \underline{\mathbf{h}}^{f^T}\underline{\mathbf{r}}_{xd} \tag{4.16.b}$$

Uma outra forma de obter o erro de predição mínimo e a sua variância é substituindo a Eq.(4.5) do preditor ótimo na Eq.(4.8) do erro de predição, logo:

$$e^f(n) = x(n) - \hat{x}(n) = x(n) + \sum_{i=1}^{p} a_i x(n-i) = \upsilon(n) \tag{4.17.a}$$

e portanto,

$$J_{Min} = \mathrm{E}[e^{f^2}] = \sigma_{e^f}^2 = \sigma_\upsilon^2 \tag{4.17.b}$$

4.3 Simulações Envolvendo Predição Progressiva Linear : comparação entre os resultados obtidos pelo sistema D e os de referência fornecidos pelo sistema de Wiener-Hopf

Esta seção apresentará uma série de simulações envolvendo predição linear de processos autoregressivos. As soluções utilizando-se as equações de Wiener-Hopf são apresentadas inicialmente para servirem de referência em relação aos resultados encontrados a partir do sistema de equações D.

A primeira simulação consiste na determinação dos valores ótimos dos coeficientes de um preditor linear progressivo. O sinal aleatório de entrada é fornecido por uma função amostra pertencente a um processo autoregressivo AR(1) dado pela seguinte equação:

$$x(n) = 0.99x(n-1) + \upsilon(n) \tag{4.18.a}$$

onde $a_1 = -0.99$. Pela Eq.(4.15.a), tem-se que:

$$\underline{\mathbf{R}}_{\mathbf{x}}\underline{\mathbf{h}}^{f} = \underline{\mathbf{r}}_{\mathbf{dx}}$$

$$R_x(0)h_1^f = r_x(1)$$

$$R_x(0)h_1^f = R_x(1) \tag{4.18.b}$$

$$\text{ou} \quad h_1^f = \frac{R_x(1)}{R_x(0)} \tag{4.18.c}$$

Por ser um processo AR(1), os valores acima são calculados [40] através de:

$$R_x(0) = \frac{\sigma_\upsilon^2}{1-h_1^{f\,2}} = \frac{1}{1-0.99^2} = 50.2512 \tag{4.19.a}$$

pois foi utilizada uma variância de $\upsilon(n)$ igual a 1.

$$R_x(k) = \frac{\sigma_\upsilon^2}{1-h_1^{f\,2}}[-(-h_1^f)^k] \quad \text{para } 0 \prec \text{k} \quad \text{ou} \tag{4.19.b}$$

$$R_x(1) = \frac{\sigma_\upsilon^2}{1 - h_1^{f\,2}} h_1^f = 50.2512 * 0.99 = 49.7487 \qquad (4.19.c)$$

que são os valores teóricos obtidos para a função autocorrelação, a serem utilizados na Eq.(4.18.c)

$$h_1^f = \frac{R_x(1)}{R_x(0)} = \frac{49.7487}{50.2512} = 0.99 \qquad (4.20.a)$$

portanto,

$$\hat{x}(n) = 0.99x(n-i) \qquad (4.20.b)$$

Naturalmente, pela Eq.(4.5) já era de se esperar que o valor calculado de h_1^f pela Eq.(4.20.a) fosse igual a 0.99. Segundo a Eq. (4.16.b) o valor mínimo de J é fornecido por:

$$J_{Min} = \sigma_x^2 - h_1^f R_x(1) = 50.2512 - 0.99 * 49.7487 = 1 \qquad (4.21)$$

Esse resultado demonstra que sendo um filtro preditor ótimo, o erro de predição progressivo $e^f(n)$ será mínimo e igual ao ruído branco $\upsilon(n)$ e portanto J_{Min} deve ser igual a variância de $\upsilon(n)$, como apresentado pela Eq.(4.17.b).

No entanto, em muitas situações práticas não se conhece a priori a estatística do sinal, portanto, torna-se necessário fazer uma estimação, a partir do conjunto de amostras disponíveis, dos valores utilizados tanto no sistema de equações de Wiener-Hopf quanto no sistema de equações D. A Eq.(4.22) [39] pode ser empregada como um estimador dos valores da função autocorrelação, sendo expressa por:

$$\hat{R}_x(k) = \frac{1}{NA} \sum_{n=0}^{NA-1-k} x^*(n)x(n+k) \qquad ; \quad k = 0,1 \cdots NA-1 \qquad (4.22)$$

onde NA é o número de amostras utilizadas pelo estimador. Substituindo-se a Eq.(4.22) na Eq.(4.18.c) tem-se:

$$h_{1\ ap}^{f} = \frac{\hat{R}_x(1)}{\hat{R}_x(0)} \tag{4.23}$$

que é o valor aproximado de h_1^f.

Similarmente ao filtro preditor apresentado nas Figs.(4.2.b) e (4.3) tem-se o diagrama do preditor apresentado na Fig.(4.4), cuja estrutura se baseia na decomposição do sinal de entrada, como visto no capitulo 3. A entrada $x(n)$ a ser decomposta é obtida através da passagem da função amostra $x_1(n)$ através de um bloco de retardo unitário e o sinal desejado $d(n)$ é a própria função amostra $x_1(n)$. Logo, o erro de predição progressivo será dado por:

$$e^f(n) = e_1^f(n) = d(n) - \hat{d}(n) \tag{4.24.a}$$

$$e^f(n) = d(n) - y(n) - yy(n) \tag{4.24.b}$$

$$e^f(n) = d(n) - \sum_{i=0}^{N-1} w_i xr_i(n) \ - \sum_{i=0}^{N-1} ww_i r_i(n) \qquad \text{para} \quad n \geq 0 \tag{4.24.c}$$

$$e^f(n) = d(n) - \underline{\mathbf{w}}^T \, \underline{\mathbf{xr}}(n) - \underline{\mathbf{ww}}^T \, \underline{\mathbf{r}}(n) \tag{4.24.d}$$

onde a Eq.(4.24.d) é idêntica a Eq.(3.21.c), apenas que aqui o erro é definido como sendo um erro de predição progressivo, pois o contexto é um problema de predição linear. Portanto, foi obtido um erro de predição para o filtro D, da mesma forma que um erro foi obtido para o preditor apresentado nas Figs.(4.2.b) e (4.3). Pelo mesmo motivo apresentado anteriormente, tem-se que a variância desse erro deve ser igual a variância de $\upsilon(n) = \upsilon_1(n)$, na condição de predição ótima.

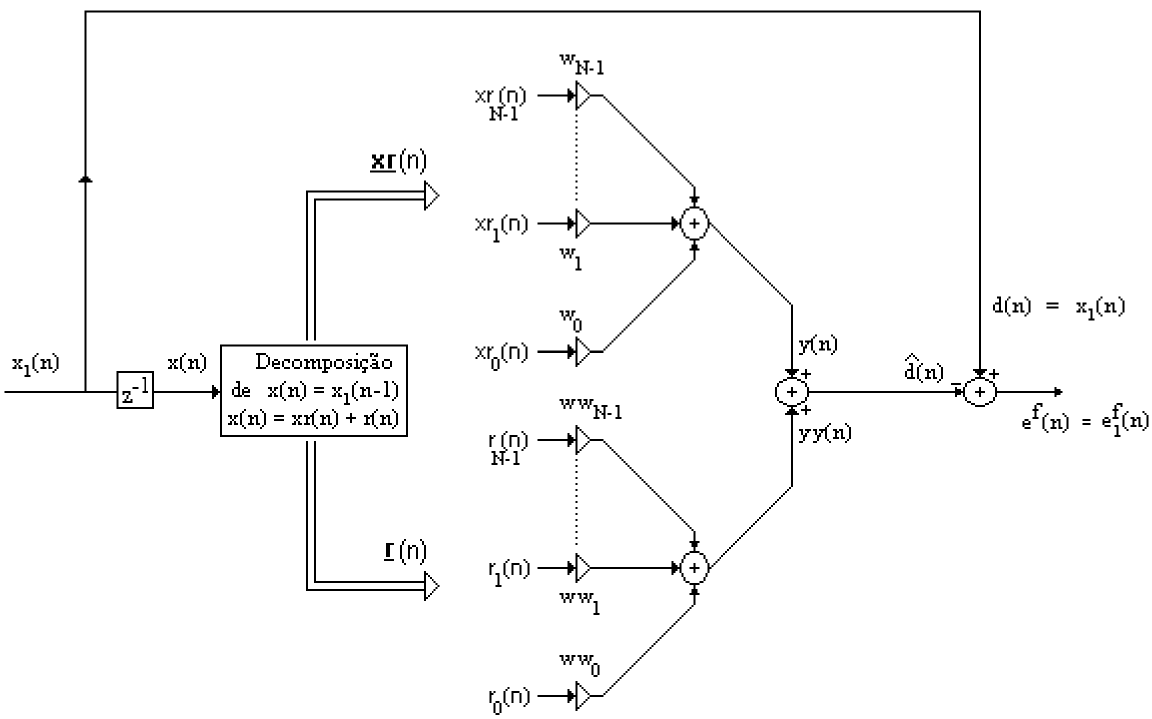


Fig.4.4. Filtro de predição linear cuja estrutura baseia-se na decomposição do sinal de entrada $x(n) = x_1(n-1)$.

Como o sistema de equações D produz diversas soluções, inclusive diferentes das produzidas pelo sistema de Wiener-Hopf, tem-se que uma forma de avaliar essas soluções é através da comparação direta entre as formas de onda do ruído $\upsilon(n)$ e a forma de onda do erro de predição progressivo $e^f(n)$. Um outro critério de avaliação é utilizar o erro absoluto entre a variância do erro de predição progressivo $\sigma_{e^f}^2$ com σ_υ^2, idealmente esses valores devem ser coincidentes. Uma variante desse critério de avaliação é o que utiliza como parâmetro de comparação o erro relativo er definido por:

$$er = \frac{(\hat{\sigma}_\upsilon^2 - \hat{\sigma}_{e^f}^2)}{\hat{\sigma}_\upsilon^2} \tag{4.25}$$

onde $\hat{\sigma}_\upsilon^2$ é a estimativa de σ_υ^2 e, $\hat{\sigma}_{e^f}^2$ é a estimativa de $\sigma_{e^f}^2$.

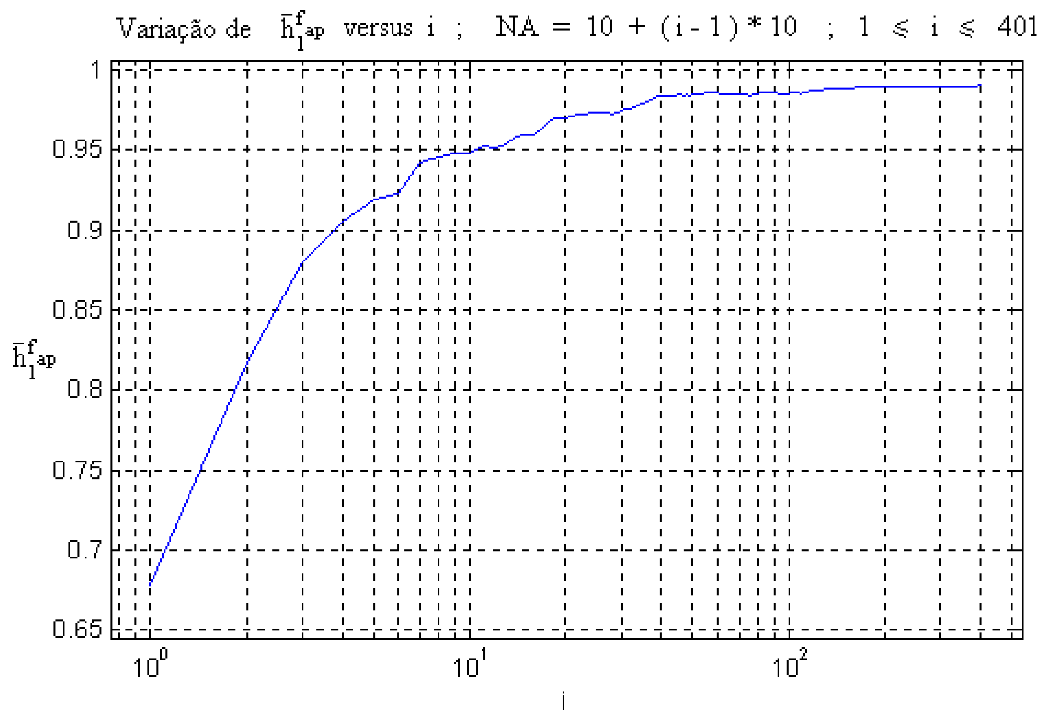
A Fig.(4.5.a) apresenta a variação dos valores médios de $h^f_{1\,ap}$ versus i, onde i relaciona-se com o número de amostras NA através da expressão: $NA = 10 + (i-1)*10$ amostras, para $1 \le i \le 401$. A partir de uma dada função amostra $\upsilon_1(n)$ com NA elementos (amostras), gera-se uma correspondente função amostra $x_1(n)$, de um processo AR(1), também com NA elementos. Como "i" varia de 1 até 401, então NA variará de 10 até 4010 amostras. Considerando $i = 1$, tem-se que o ruído $\upsilon_1(n)$, o sinal $x_1(n)$ e o erro de predição progressivo $e^f_1(n)$ possuirão $NA = 10$ amostras cada.

Utilizando-se a Eq.(4.22) estima-se os valores da função autocorrelação, em seguida, inserindo-os na Eq.(4.23) calcula-se o valor de $h^f_{1\,ap}$. Novamente utilizando-se a Eq.(4.22) com $k = 0$ calcula-se a estimativa da variância de $\upsilon_1(n)$ e $e^f_1(n)$ dadas respectivamente por $\hat{\sigma}^2_\upsilon$ e $\hat{\sigma}^2_{e^f}$. Lembrando-se que para um preditor ótimo essas estimativas devem coincidir, então, emprega-se a Eq.(4.25) para comparar o erro entre elas. Esses cálculos são repetidos de $i = 1, 2, 3, \cdots 401$ ou seja de $NA = 10, 20, 30, \cdots, 4010$ amostras e dessa forma gerou-se um vetor linha de 401 elementos para cada um dos parâmetros $h^f_{1\,ap}$, $\hat{\sigma}^2_\upsilon$, $\hat{\sigma}^2_{e^f}$ e er.

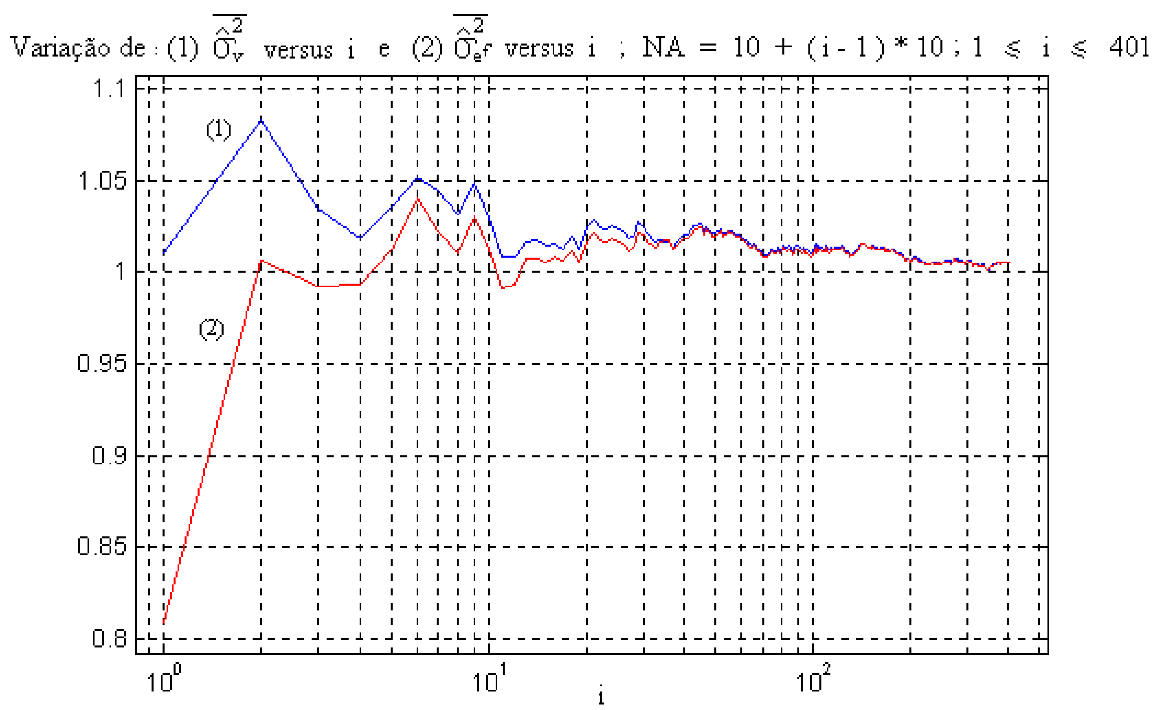
Admitindo-se que em vez de se ter apenas uma dada função amostra $\upsilon_1(n)$ tenha-se NF funções amostras $\upsilon_1(n)$, $\upsilon_2(n)$, $\cdots$, $\upsilon_{NF}(n)$, então pode-se gerar NF vetores linha de 401 elementos cada, ou seja, uma matriz de NF por 401 elementos para cada um dos parâmetros $h^f_{1\,ap}$, $\hat{\sigma}^2_\upsilon$, $\hat{\sigma}^2_{e^f}$ e er, denominadas respectivamente por $\underline{\boldsymbol{H}^f_{1\,ap}}$, $\underline{\hat{\sigma}^2_\upsilon}$, $\underline{\hat{\sigma}^2_{e^f}}$ e $\underline{\mathbf{ER}}$. Os valores médios estimados de $h^f_{1\,ap}$ são representados por $\overline{h}^f_{1\,ap}$ e calculados fazendo-se a média dos elementos pertencentes a cada uma das colunas da matriz $\underline{\boldsymbol{H}^f_{1\,ap}}$ e de forma análoga tem-se o mesmo para os outros parâmetros, representados respectivamente por $\overline{\hat{\sigma}^2_\upsilon}$, $\overline{\hat{\sigma}^2_{e^f}}$ e $\overline{er}$.

Os gráficos da Fig.(4.5) foram obtidos fixando o valor de NF = 20. Ainda com relação a Fig.(4.5.a), observa-se uma maior discrepância entre o valor teórico $h^f_1 = 0.99$ e os valores médios de $h_{1\,ap}^{f}$ no início do gráfico, isto devido ao baixo número de amostras envolvidas no processo de estimação dos valores da função autocorrelação. Nessa figura, a região de $1 \le i \le 10$ correspondente a uma variação de $0.68 \le \overline{h}_{1\,ap}^{f} \le 0.95$, isto é, ocorreu uma mudança significativa nos valores de $\overline{h}_{1\,ap}^{f}$ para uma pequena variação no número de amostras NA, de 10 para 100. Já na região compreendida entre $10 \le i \le 401$, ocorreu uma pequena variação nos valores de $\overline{h}_{1\,ap}^{f}$ para uma grande variação no número de amostras NA, de 100 para 4010, significando a proximidade dos valores de $\overline{h}_{1\,ap}^{f}$ em relação a h_1^f.

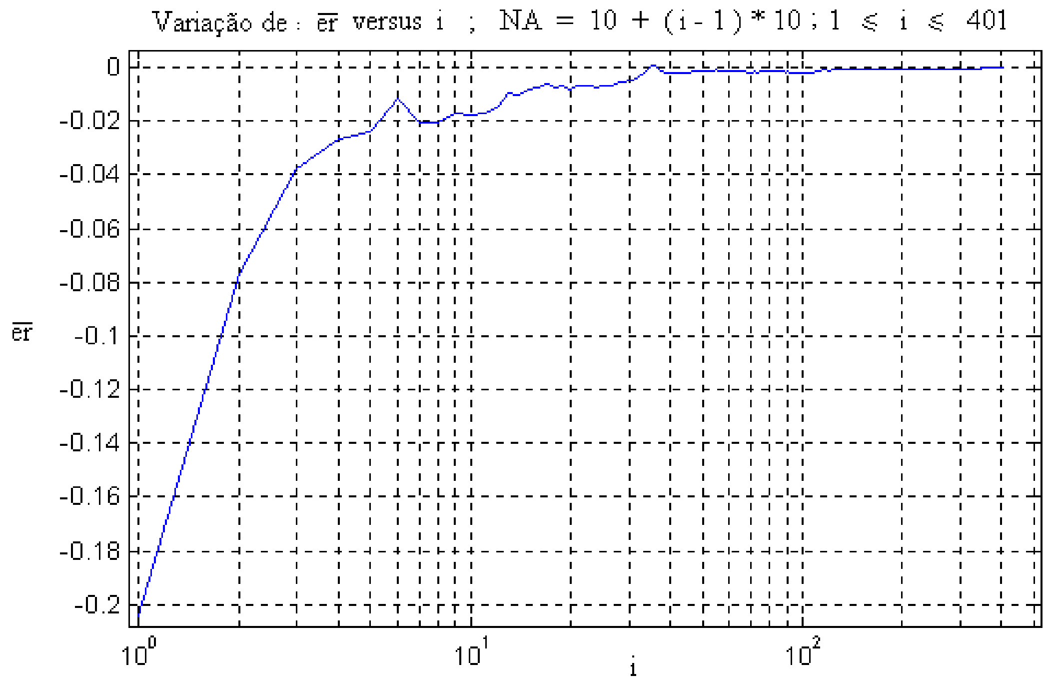
A Fig.(4.5.b) apresenta os gráficos relativos a variação do valores médios de $\hat{\sigma}_{\upsilon}^{2}$ versus i e dos valores médios de $\hat{\sigma}_{e^f}^{2}$ versus i. Supondo-se que para qualquer número de amostras, pequenas ou grandes, resultasse em uma estimação exata de h_1^f, então os gráficos (1) e (2) presentes nessa figura, seriam idênticos. A partir de i = 20, praticamente coincide a variância estimada média do ruído com a variância estimada média do erro de predição progressivo. Uma outra forma de observar isso é através da variação dos valores médios de er versus i, apresentado pela Fig.(4.5.c). Os gráficos contidos nas Figs.(4.6.a), (4.6.b) e (4.6.c) são análogos aos da Fig.(4.5), só que o valor do coeficiente a_1 do processo AR(1) foi modificado de -0.99 para -0.2 e portanto, o preditor ótimo será dado por $\hat{x}(n) = 0.2x(n-1)$ onde $h_1^f = 0.2$.


Variação de $\bar{h}_1^{f_{ap}}$ versus i ; NA = 10 + (i - 1) * 10 ; 1 ⩽ i ⩽ 401
$\bar{h}_1^{f_{ap}}$
1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
10^0
10^1
10^2
i

(a)

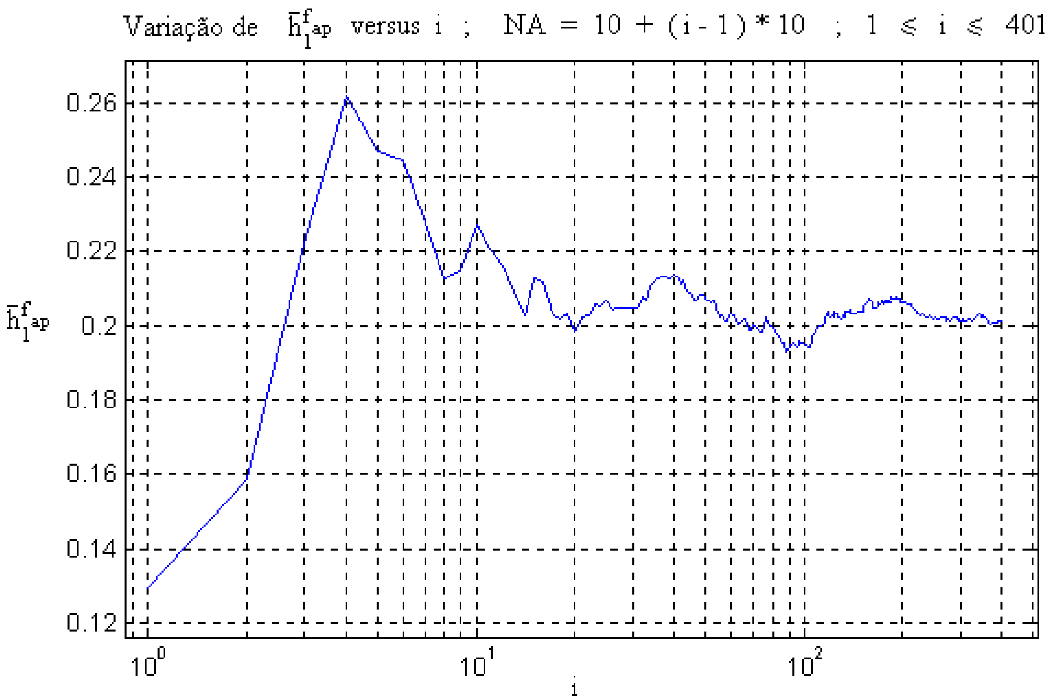

Variação de : (1) $\overline{\hat{\sigma}_v^2}$ versus i e (2) $\overline{\hat{\sigma}_{e^f}^2}$ versus i ; NA = 10 + (i - 1) * 10 ; 1 ⩽ i ⩽ 401
(1)
(2)
1.1
1.05
1
0.95
0.9
0.85
0.8
10^0
10^1
10^2
i

(b)



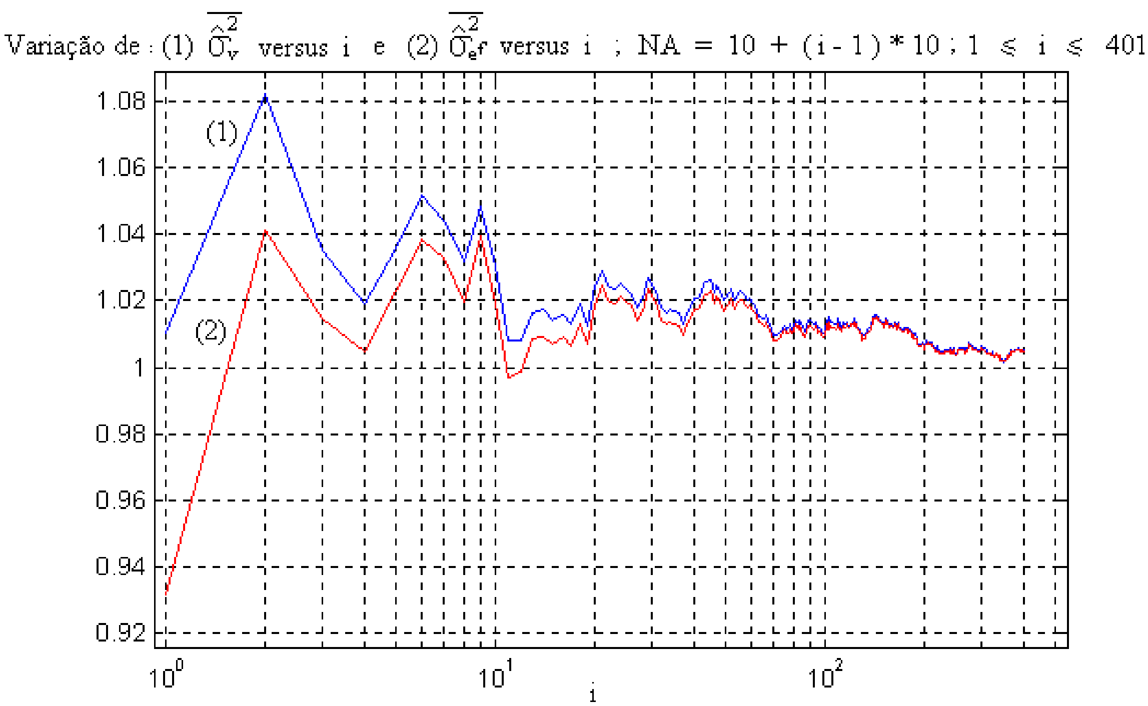
c)

Fig.4.5. Variação de parâmetros médios obtidos de um processo AR(1), onde a1 = -0.99. Os valores médios são calculados considerando que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \leq i \leq 401$ e o valor de NF igual a 20. a) Variação de $\overline{h}^{f}_{1\,ap}$ *versus* i . b) Gráficos relativos a variação dos valores médios: (1) $\overline{\hat{\sigma}^2_{\upsilon}}$ versus i (2) $\overline{\hat{\sigma}^2_{e^f}}$ versus i . c) Variação $\overline{er}$ *versus* i .



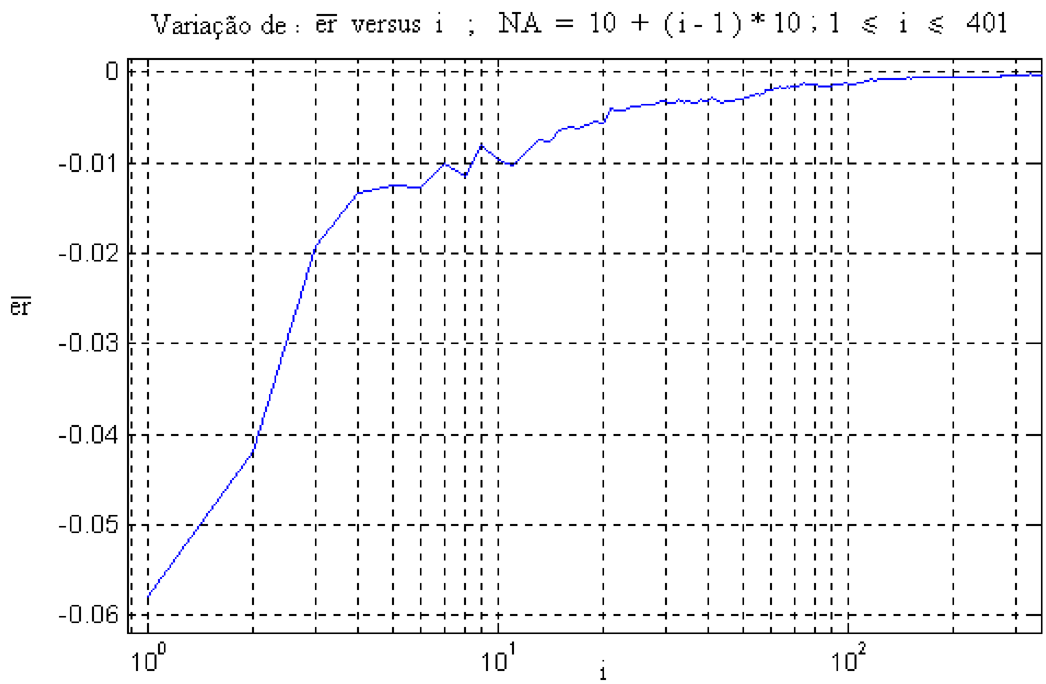
Variação de $\bar{h}_1^{f_{ap}}$ versus i ; NA = 10 + (i - 1) * 10 ; 1 ≤ i ≤ 401
$\bar{h}_1^{f_{ap}}$
0.26
0.24
0.22
0.2
0.18
0.16
0.14
0.12
10^0
10^1
10^2
i

a)



Variação de : (1) $\overline{\hat{\sigma}_v^2}$ versus i e (2) $\overline{\hat{\sigma}_{e^f}^2}$ versus i ; NA = 10 + (i - 1) * 10 ; 1 ≤ i ≤ 401
(1)
(2)
1.08
1.06
1.04
1.02
1
0.98
0.96
0.94
0.92
10^0
10^1
10^2
i

b)



c)

Fig.4.6. Variação de parâmetros médios obtidos de um processo AR(1), onde a1 = -0.2. Os valores médios são calculados considerando-se que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \leq i \leq 401$ e o valor de NF igual a 20. (a) Variação de $\overline{h}^{f}_{1\,ap}$ *versus* i. (b) Gráficos relativos a variação dos valores médios: (1) $\overline{\hat{\sigma}^{2}_{\upsilon}}$ versus i (2) $\overline{\hat{\sigma}^{2}_{e^f}}$ versus i . (c) Variação $\overline{er}$ *versus* i .

A especificação do número de entradas N é o primeiro passo para aplicar as equações D em predição linear de um passo a frente. Como explicado no Capítulo 3, a adoção de N igual a 1 ou 2 gera um sistema de equações idêntico ao sistema de Wiener-Hopf, portanto foi escolhido o valor imediatamente superior, isto é , N = 3. Baseando-se nas Eqs.(3.66.a) e (3.66.b) e utilizando nesse sistema todas as entradas não nulas decompostas, o sistema de equações será constituído por:

$$
\begin{array}{ccccccccccc}
R_{xr_1\, r_0} ww_0 & + & R_{xr_1\, r_1} ww_1 & + & R_{xr_1\, r_2} ww_2 & + & R_{xr_1 xr_1} w_1 & = & p_1 \\
R_{r_0 r_0} ww_0 & + & R_{r_0 r_1} ww_1 & + & R_{r_0 r_2} ww_2 & + & R_{r_0\, xr_1} w_1 & = & pp_0 \\
R_{r_1 r_0} ww_0 & + & R_{r_1 r_1} ww_1 & + & R_{r_1 r_2} ww_2 & + & R_{r_1\, xr_1} w_1 & = & pp_1 \\
R_{r_2 r_0} ww_0 & + & R_{r_2 r_1} ww_1 & + & R_{r_2 r_2} ww_2 & + & R_{r_2\, xr_1} w_1 & = & pp_2
\end{array}
\qquad (4.26.a)
$$

ou na forma matricial $\underline{\mathbf{A}}\underline{\mathbf{x}}=\underline{\mathbf{b}}$ onde

$$\underline{\mathbf{A}} = \begin{bmatrix} R_{xr_1\,r_0} & R_{xr_1\,r_1} & R_{xr_1\,r_2} & R_{xr_1xr_1} \\ R_{r_0r_0} & R_{r_0r_1} & R_{r_0r_2} & R_{r_0\,xr_1} \\ R_{r_1r_0} & R_{r_1r_1} & R_{r_1r_2} & R_{r_1\,xr_1} \\ R_{r_2r_0} & R_{r_2r_1} & R_{r_2r_2} & R_{r_2\,xr_1} \end{bmatrix} \tag{4.26.b}$$

$$\underline{\mathbf{x}} = \begin{bmatrix} ww_0 \\ ww_1 \\ ww_2 \\ w_1 \end{bmatrix} \tag{4.26.c}$$

$$\underline{\mathbf{b}} = \begin{bmatrix} p_1 \\ pp_0 \\ pp_1 \\ pp_2 \end{bmatrix} \tag{4.26.d}$$

A Eq.(4.26.a) é um sistema de 4 equações a 4 incógnitas, isto é, o maior sistema capaz de ser formado considerando N=3. No entanto, o objetivo é encontrar um sistema com a menor dimensão possível e que gere uma solução tal que implique em uma $\sigma^2_{e^f} = \sigma^2_v$, portanto, na menor variância do erro de predição. A diminuição da ordem do sistema foi comentada no Capítulo 3, de forma que a seguir são apresentados vários sistemas com dimensões inferiores ao da Eq.(4.26.a).

a)O primeiro sistema proposto é dado pela Eq.(4.27.a). Ele é obtido através da eliminação da entrada $r_1(n)$ e conseqüentemente da incógnita ww_1. Esse novo sistema a ser *avaliado* é formado eliminando-se a linha 3 e a coluna 2 da Eq.(4.26.b), a linha 2 da Eq.(4.26.c) e a linha 3 da Eq.(4.26.d) conforme apresentado a seguir:

$$\underline{\mathbf{A}}_{32}\,\underline{\mathbf{x}}_{32} = \underline{\mathbf{p}}_{32} \tag{4.27.a}$$

onde

$$\underline{\mathbf{A}}_{32} = \begin{bmatrix} R_{xr_1\, r_0} & R_{xr_1\, r_2} & R_{xr_1 xr_1} \\ R_{r_0 r_0} & R_{r_0 r_2} & R_{r_0\, xr_1} \\ R_{r_2 r_0} & R_{r_2 r_2} & R_{r_2\, xr_1} \end{bmatrix} \tag{4.27.b}$$

$$\underline{\mathbf{x}}_{32} = \begin{bmatrix} ww_0 \\ ww_2 \\ w_1 \end{bmatrix} \tag{4.27.c}$$

$$\underline{\mathbf{p}}_{32} = \begin{bmatrix} p_1 \\ pp_0 \\ pp_2 \end{bmatrix} \tag{4.27.d}$$

A Fig.(4.7.a) apresenta a variação dos valores médios de ww_0, ww_2 e w_1 versus i, onde i relaciona-se com o número de amostras NA através da expressão: $NA = 10 + (i-1)*10$ amostras, para $1 \le i \le 401$. A partir de uma dada função amostra $\upsilon_1(n)$ com NA elementos (amostras), gera-se uma correspondente função amostra $x_1(n)$ de um processo AR(1), também com NA elementos. Como "i" varia de 1 até 401, então NA variará de 10 até 4010 amostras. Considerando $i = 1$, tem-se que o ruído $\upsilon_1(n)$, as entradas obtidas da decomposição de $x(n) = x_1(n-1)$ dadas por $xr_1(n), r_0(n)$ e $r_2(n)$, o sinal desejado $d(n) = x_1(n)$ e o erro de predição progressivo $e_1^f(n)$, possuirão $NA = 10$ amostras cada.

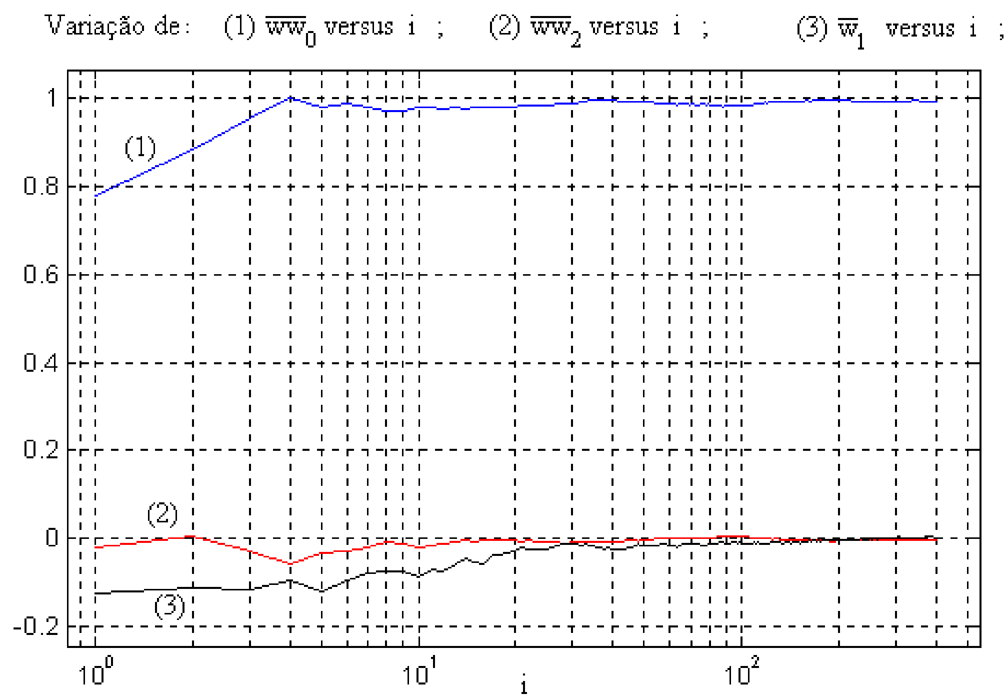
As incógnitas ww_0, ww_2 e w_1 são obtidas através da Eq.(4.27.a). Os valores da função autocorrelação e da correlação cruzada, ambas com deslocamentos nulos, presentes nas Eqs.(4.27.b) e (4.27.d) são estimadas utilizando-se uma expressão similar a Eq.(4.22), sendo dada por:

$$\hat{R}_{xy}(0) = \frac{1}{NA} \sum_{n=0}^{NA-1} x^*(n) y(n) \tag{4.28}$$

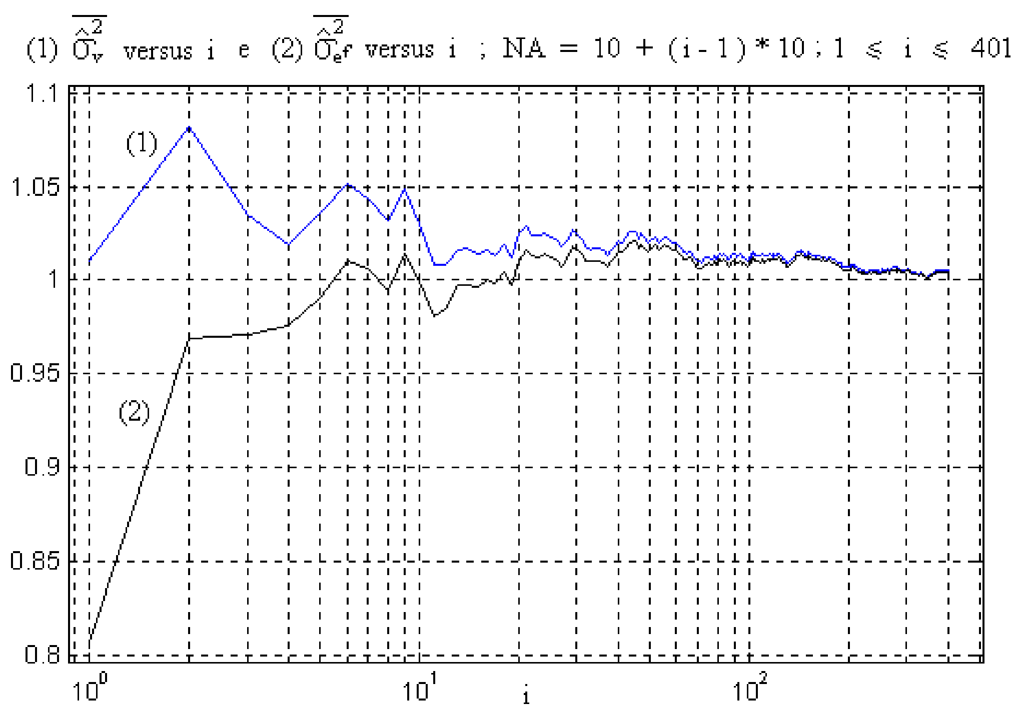
onde $x(n)$ e $y(n)$ são funções amostras quaisquer e NA é o número de amostras utilizadas pelo estimador.

Novamente, utilizando-se a Eq.(4.22) com $k=0$, calcula-se a estimativa da variância de $\upsilon_1(n)$ e $e_1^f(n)$, dadas respectivamente por $\hat{\sigma}_\upsilon^2$ e $\hat{\sigma}_{e^f}^2$. Lembrando-se que para um preditor ótimo essas estimativas devem coincidir, então a Eq.(4.25) é empregada para comparar o erro entre elas. Esses cálculos foram repetidos de $i=1,2,3,\cdots 401$ ou seja de $NA=10,20,30,\cdots,4010$ amostras e dessa forma gerou-se um vetor linha de 401 elementos para cada um dos parâmetros ww_0, ww_2, w_1, $\hat{\sigma}_\upsilon^2$, $\hat{\sigma}_{e^f}^2$ e er. Admitindo-se que, em vez de se ter apenas uma dada função amostra $\upsilon_1(n)$, tenha-se NF funções amostras $\upsilon_1(n)$, $\upsilon_2(n)$, $\cdots$, $\upsilon_{NF}(n)$, então pode-se gerar NF vetores linha de 401 elementos cada, ou seja, uma matriz de NF por 401 elementos para cada um dos parâmetros ww_0, ww_2, w_1, $\hat{\sigma}_\upsilon^2$, $\hat{\sigma}_{e^f}^2$ e er, denominadas respectivamente por $\underline{\boldsymbol{WW}_0}$, $\underline{\boldsymbol{WW}_2}$, $\underline{\boldsymbol{W}_1}$, $\underline{\hat{\sigma}_\upsilon^2}$, $\underline{\hat{\sigma}_{e^f}^2}$ e $\underline{\mathbf{ER}}$. Os valores médios estimados de ww_0 são representados por $\overline{ww_0}$ e calculados fazendo a média dos elementos pertencentes a cada uma das colunas da matriz $\underline{\boldsymbol{WW}_0}$ e de forma análoga tem-se o mesmo para os outros parâmetros, representados respectivamente por $\overline{ww_2}$, $\overline{w_1}$, $\overline{\hat{\sigma}_\upsilon^2}$, $\overline{\hat{\sigma}_{e^f}^2}$ e $\overline{er}$.

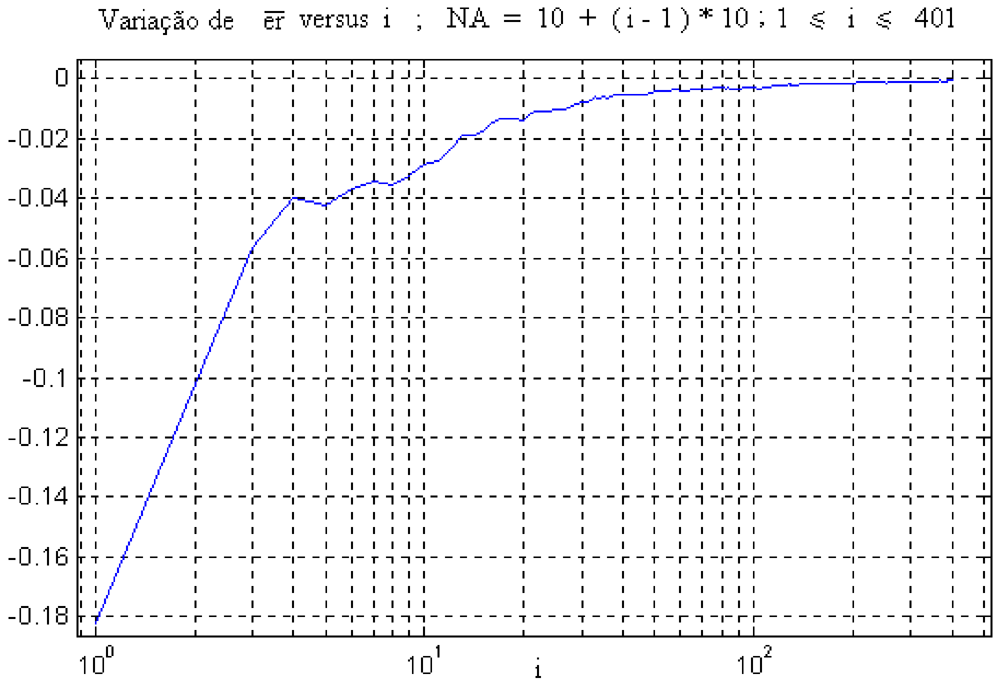
Os gráficos da Fig.(4.7) foram obtidos fixando o valor de NF = 20. Nessa figura, a região de $1 \le i \le 4$ correspondente a uma variação de $0.77 \le \overline{ww_0} \le 1$, isto é, ocorreu uma mudança significativa nos valores de $\overline{ww_0}$ para uma pequena variação no número de amostras NA, de 10 para 40. Já na região compreendida entre $4 \le i \le 401$, ocorreu uma pequena variação nos valores de $\overline{ww_0}$ para uma grande variação no número de amostras NA, de 40 para 4010, significando uma proximidade dos valores de $\overline{ww_0}$ em relação ao cálculo desse parâmetro considerando um número de amostras $NA \to \infty$. A Fig.(4.7.b) apresenta os gráficos relativos a variação do valores médios de $\hat{\sigma}_\upsilon^2$ versus i e dos valores médios de $\hat{\sigma}_{e^f}^2$ versus i, enfatizando que o ideal é obtido com a igualdade entre os gráficos (1) e (2). A partir de i = 20, praticamente coincide a variância estimada média do ruído com a variância estimada média do erro de predição progressivo. Uma outra forma de observar isso é através da variação dos valores médios de er versus i, apresentado pela Fig.(4.7.c).


Variação de : (1) $\overline{ww}_0$ versus i ; (2) $\overline{ww}_2$ versus i ; (3) $\overline{w}_1$ versus i ;
(1)
(2)
(3)
1
0.8
0.6
0.4
0.2
0
-0.2
10^0
10^1
10^2
i

a)


(1) $\overline{\hat{\sigma}^2_v}$ versus i e (2) $\overline{\hat{\sigma}^2_{ef}}$ versus i ; NA = 10 + (i - 1) * 10 ; 1 ⩽ i ⩽ 401
(1)
(2)
1.1
1.05
1
0.95
0.9
0.85
0.8
10^0
10^1
10^2
i

b)



c)

Fig.4.7. Variação de parâmetros médios relacionados ao filtro D. O sinal desejado é obtido de um processo AR(1), onde a1 = -0.99. Os valores médios são calculados considerando que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \leq i \leq 401$ e o valor de NF igual a 20. (a) Variação de: (1) $\overline{ww_0}$ *versus i* ; (2) $\overline{ww_2}$ *versus i* ; (3) $\overline{w_1}$ *versus i* (b) Gráficos relativos a variação dos valores médios: (1) $\overline{\hat{\sigma}_\upsilon^2}$ versus i (2) $\overline{\hat{\sigma}_{e^f}^2}$ versus i . (c) Variação de $\overline{er}$ *versus i* .

b)O segundo sistema proposto é dado pela Eq.(4.29.a). Ele é obtido através da eliminação da entrada $r_0(n)$ e conseqüentemente da incógnita ww_0. Esse novo sistema a ser *avaliado* é formado eliminando-se a linha 2 e a coluna 1 da Eq.(4.26.b), a linha 1 da Eq.(4.26.c) e a linha 2 da Eq.(4.26.d) conforme apresentado a seguir:

$$\underline{\mathbf{A}}_{21}\underline{\mathbf{x}}_{21} = \underline{\mathbf{p}}_{21} \tag{4.29.a}$$

onde

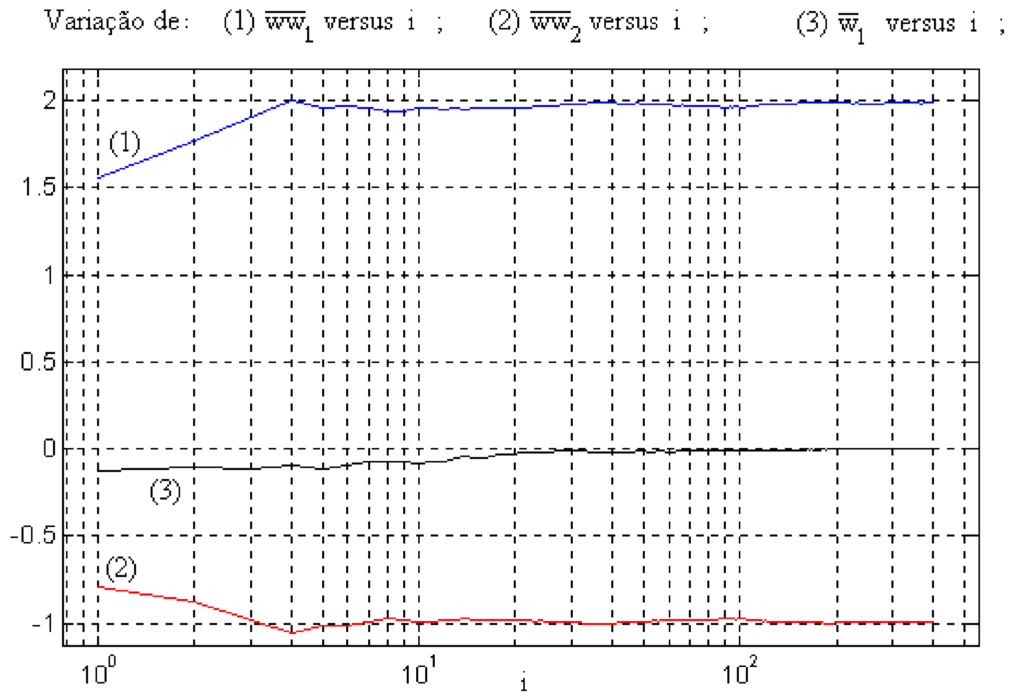
$$\underline{\mathbf{A}}_{21} = \begin{bmatrix} R_{xr_1 r_1} & R_{xr_1 r_2} & R_{xr_1 xr_1} \\ R_{r_1 r_1} & R_{r_1 r_2} & R_{r_1 xr_1} \\ R_{r_2 r_1} & R_{r_2 r_2} & R_{r_2 xr_1} \end{bmatrix} \tag{4.29.b}$$

$$\underline{\mathbf{x}}_{21} = \begin{bmatrix} ww_1 \\ ww_2 \\ w_1 \end{bmatrix} \tag{4.29.c}$$

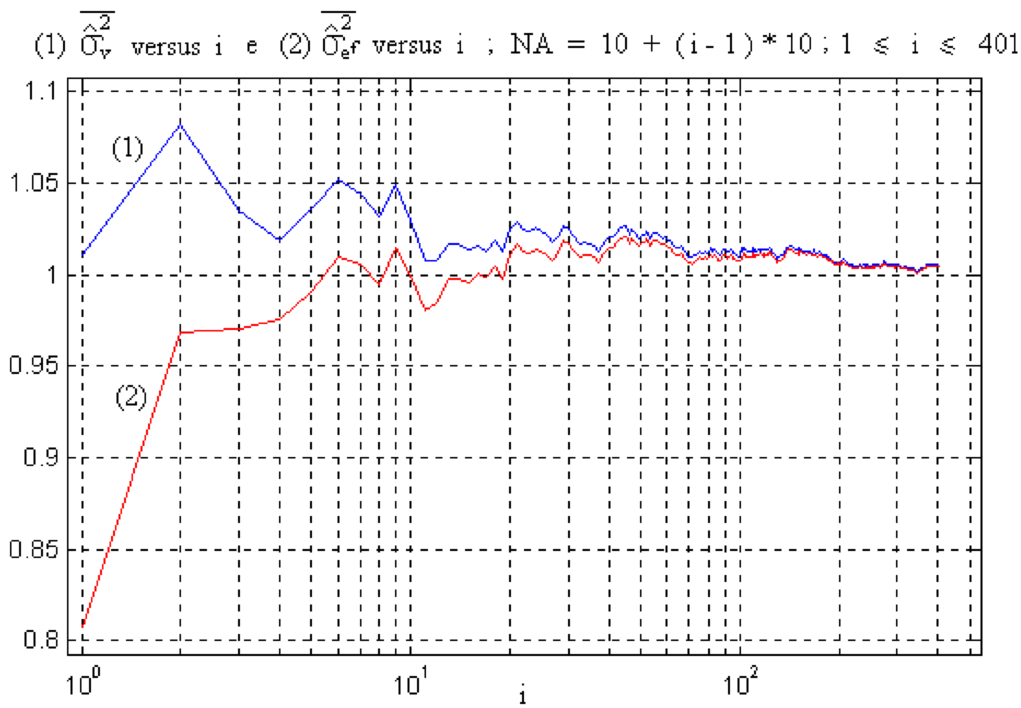
$$\underline{\mathbf{p}}_{21} = \begin{bmatrix} p_1 \\ pp_1 \\ pp_2 \end{bmatrix} \tag{4.29.d}$$

A Fig.(4.8.a) apresenta a variação dos valores médios de ww_1, ww_2 e w_1 versus i, onde i relaciona-se com o número de amostras NA através da expressão: $NA = 10 + (i-1)*10$ amostras, para $1 \leq i \leq 401$. De forma análoga ao sistema D analisado no item (a), tem-se que as entradas obtidas da decomposição de $x(n) = x_1(n-1)$ são dadas por $xr_1(n), r_1(n)$ e $r_2(n)$, o sinal desejado é o mesmo do item (a) $d(n) = x_1(n)$ e o erro de predição progressivo $e_1^f(n)$ é calculado inserindo os valores de ww_1, ww_2, w_1 e suas correspondentes entradas decompostas na Eq.(4.24.d). As incógnitas ww_1, ww_2 e w_1 são obtidas através da Eq.(4.29.a). Os valores da função autocorrelação e da correlação cruzada, ambas com deslocamentos nulos, presentes nas Eqs.(4.29.b) e (4.29.d) são estimadas utilizando a Eq.(4.28).

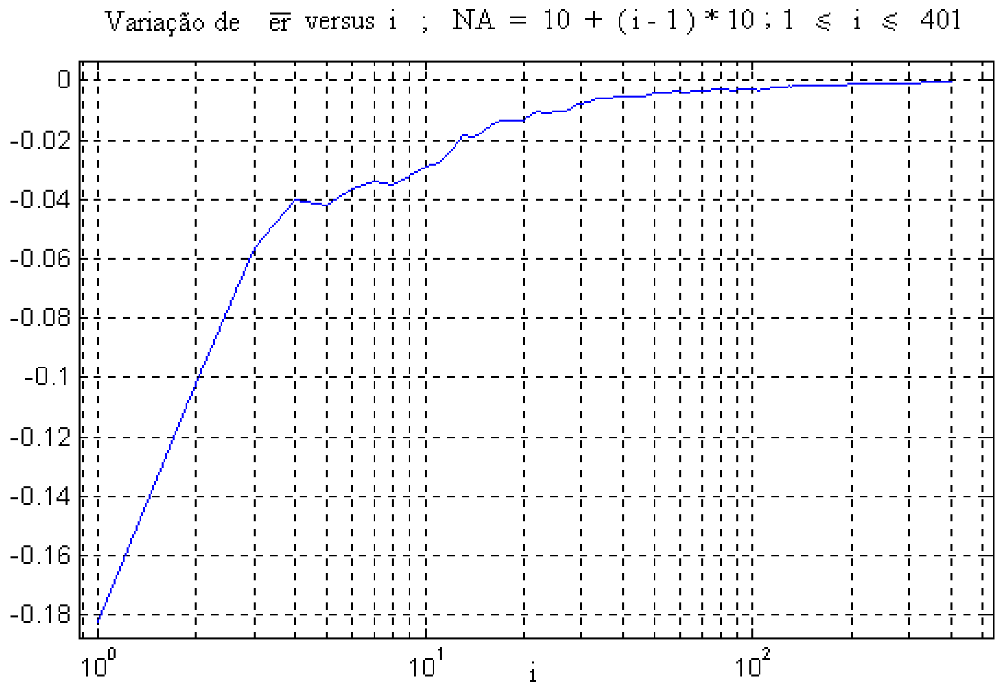
Novamente, utilizando a Eq.(4.22) com $k = 0$, calcula-se a estimativa da variância de $\upsilon_1(n)$ e $e_1^f(n)$, dadas respectivamente por $\hat{\sigma}_{\upsilon}^2$ e $\hat{\sigma}_{e^f}^2$ e por último se emprega a Eq.(4.25) para comparar o erro entre elas. Similarmente aos casos anteriores tem-se os seguintes valores médios estimados: $\overline{ww_1}, \overline{ww_2}$, $\overline{w_1}$, $\overline{\hat{\sigma}_{\upsilon}^2}$, $\overline{\hat{\sigma}_{e^f}^2}$ e $\overline{er}$. A Fig.(4.8.a) apresenta a variação dos valores médios $\overline{ww_1}$ versus i, $\overline{ww_2}$ versus i e $\overline{w_1}$ versus i. A Fig.(4.8.b) apresenta a variação dos valores médios $\overline{\hat{\sigma}_{\upsilon}^2}$ versus i e $\overline{\hat{\sigma}_{e^f}^2}$ versus i. A Fig.(4.8.c) apresenta a variação de $\overline{er}$ versus i.


Variação de: (1) $\overline{ww}_1$ versus i ; (2) $\overline{ww}_2$ versus i ; (3) $\overline{w}_1$ versus i ;
2
1.5
1
0.5
0
-0.5
-1
(1)
(3)
(2)
10^0
10^1
i
10^2

a)


(1) $\overline{\hat{\sigma}_v^2}$ versus i e (2) $\overline{\hat{\sigma}_{ef}^2}$ versus i ; NA = 10 + (i - 1) * 10 ; 1 ≤ i ≤ 401
1.1
1.05
1
0.95
0.9
0.85
0.8
(1)
(2)
10^0
10^1
i
10^2

b)



c)

Fig.4.8. Variação de parâmetros médios relacionados ao filtro D. O sinal desejado é obtido de um processo AR(1), onde a1 = -0.99. Os valores médios são calculados considerando-se que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \leq i \leq 401$ e o valor de NF igual a 20. (a) Variação de: (1) $\overline{ww_1}$ *versus i* ; (2) $\overline{ww_2}$ *versus i* ; (3) $\overline{w_1}$ *versus i* (b) Gráficos relativos a variação dos valores médios: (1) $\overline{\hat{\sigma}_\upsilon^2}$ versus i (2) $\overline{\hat{\sigma}_{e^f}^2}$ versus i . (c) Variação de $\overline{er}$ *versus i* .

c)O terceiro sistema proposto é dado pela Eq.(4.30.a). Ele é obtido através da eliminação da entrada $r_2(n)$ e conseqüentemente da incógnita ww_2. Esse novo sistema a ser *avaliado* é formado eliminando-se a linha 4 e a coluna 3 da Eq.(4.26.b), a linha 3 da Eq.(4.26.c) e a linha 4 da Eq.(4.26.d) conforme apresentado a seguir:

$$\underline{\mathbf{A}}_{43}\underline{\mathbf{x}}_{43} = \underline{\mathbf{p}}_{43} \qquad (4.30.a)$$

onde

$$\underline{\mathbf{A}}_{43} = \begin{bmatrix} R_{xr_1\,r_0} & R_{xr_1\,r_1} & R_{xr_1 xr_1} \\ R_{r_0 r_0} & R_{r_0 r_1} & R_{r_0\,xr_1} \\ R_{r_1 r_0} & R_{r_1 r_1} & R_{r_1\,xr_1} \end{bmatrix} \quad (4.30.b)$$

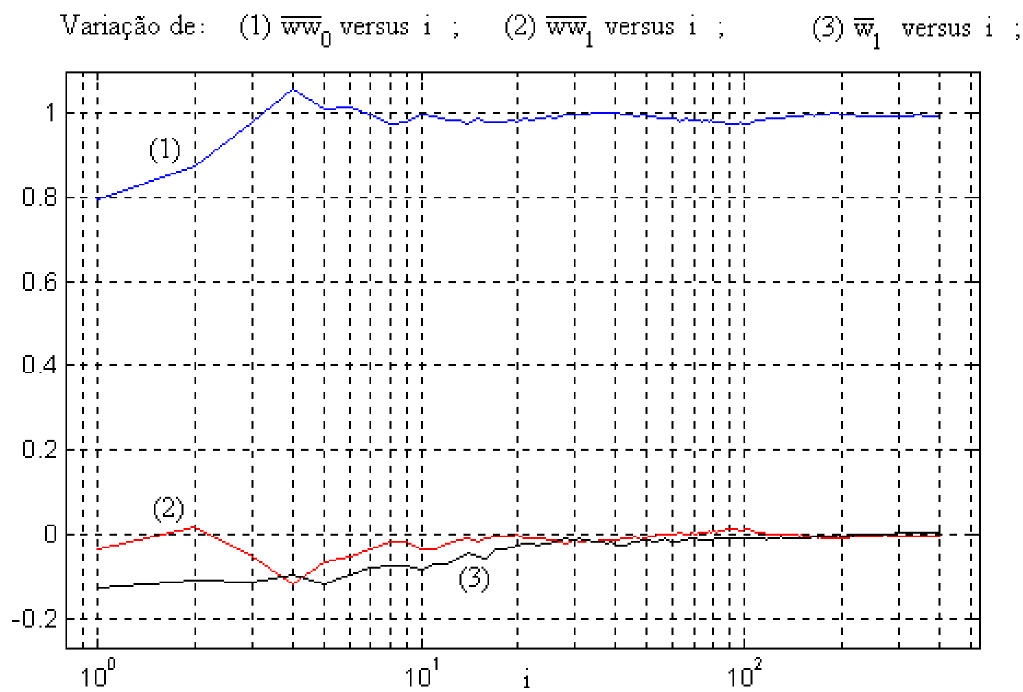
$$\underline{\mathbf{x}}_{43} = \begin{bmatrix} ww_0 \\ ww_1 \\ w_1 \end{bmatrix} \quad (4.30.c)$$

$$\underline{\mathbf{p}}_{43} = \begin{bmatrix} p_1 \\ pp_0 \\ pp_1 \end{bmatrix} \quad (4.30.d)$$

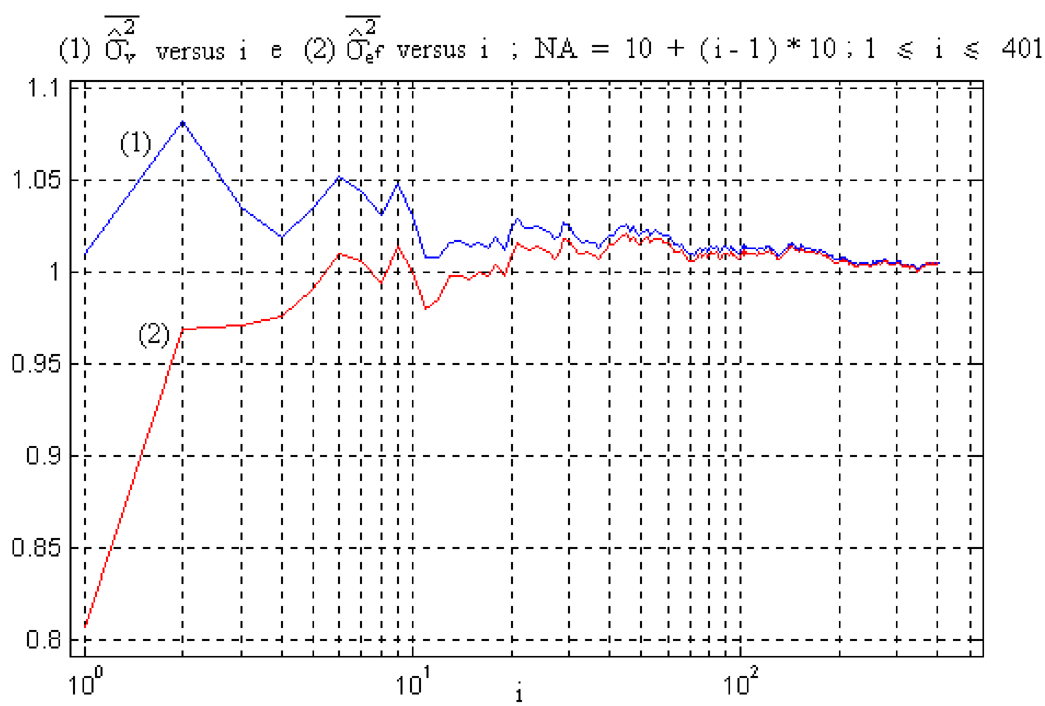
A Fig.(4.9.a) apresenta a variação dos valores médios de ww_0, ww_1 e w_1 versus i, onde i relaciona-se com o número de amostras NA, através da expressão: $NA = 10 + (i-1)*10$ amostras, para $1 \le i \le 401$. De forma análoga ao sistema D analisado nos itens (a) e (b) , tem-se que as entradas obtidas da decomposição de $x(n) = x_1(n-1)$ são dadas por $xr_1(n), r_0(n)$ e $r_1(n)$, o sinal desejado $d(n) = x_1(n)$ é o mesmo do item (a) e (b) e o erro de predição progressivo $e_1^f(n)$ é calculado inserindo os valores de ww_0, ww_1 e w_1 e suas correspondentes entradas decompostas na Eq.(4.24.d). As incógnitas ww_0, ww_1 e w_1 são obtidas através da Eq.(4.30.a). Os valores da função autocorrelação e da correlação cruzada, ambas com deslocamentos nulos, presentes nas Eqs.(4.30.b) e (4.30.d) são estimadas utilizando a Eq.(4.28).

Utilizando-se a Eq.(4.22) com $k = 0$, calcula-se a estimativa da variância de $\upsilon_1(n)$ e $e_1^f(n)$, dadas respectivamente por $\hat{\sigma}_\upsilon^2$ e $\hat{\sigma}_{e^f}^2$ e por último se emprega a Eq.(4.25) para comparar o erro entre elas. Similarmente aos casos anteriores tem-se os seguintes valores médios estimados: $\overline{ww_0}$, $\overline{ww_1}$, $\overline{w_1}$, $\overline{\hat{\sigma}_\upsilon^2}$, $\overline{\hat{\sigma}_{e^f}^2}$ e $\overline{er}$.

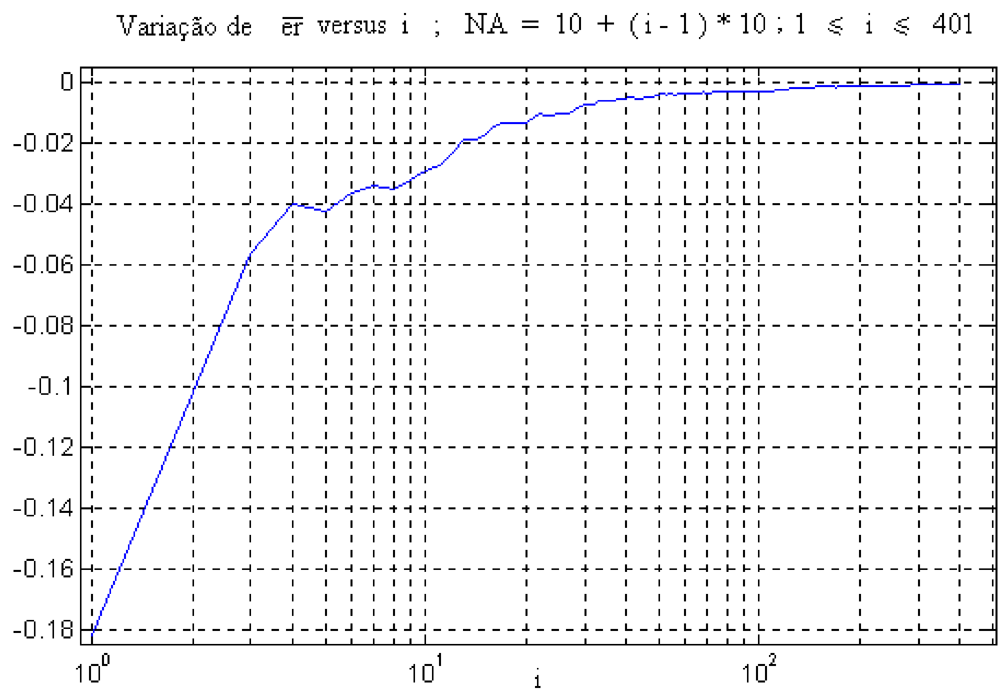
A Fig.(4.9.a) apresenta a variação dos valores médios $\overline{ww_0}$ versus i, $\overline{ww_1}$ versus i e $\overline{w_1}$ versus i. A Fig.(4.9.b) apresenta a variação dos valores médios $\overline{\hat{\sigma}_\upsilon^2}$ versus i e $\overline{\hat{\sigma}_{e^f}^2}$ versus i. A Fig.(4.9.c) apresenta a variação de $\overline{er}$ versus i.


Variação de: (1) $\overline{ww}_0$ versus i ; (2) $\overline{ww}_1$ versus i ; (3) $\overline{w}_1$ versus i ;
(1)
(2)
(3)
1
0.8
0.6
0.4
0.2
0
-0.2
10^0
10^1
i
10^2

a)


(1) $\overline{\hat{\sigma}_v^2}$ versus i e (2) $\overline{\hat{\sigma}_{ef}^2}$ versus i ; NA = 10 + (i - 1) * 10 ; 1 ≤ i ≤ 401
(1)
(2)
1.1
1.05
1
0.95
0.9
0.85
0.8
10^0
10^1
i
10^2

b)



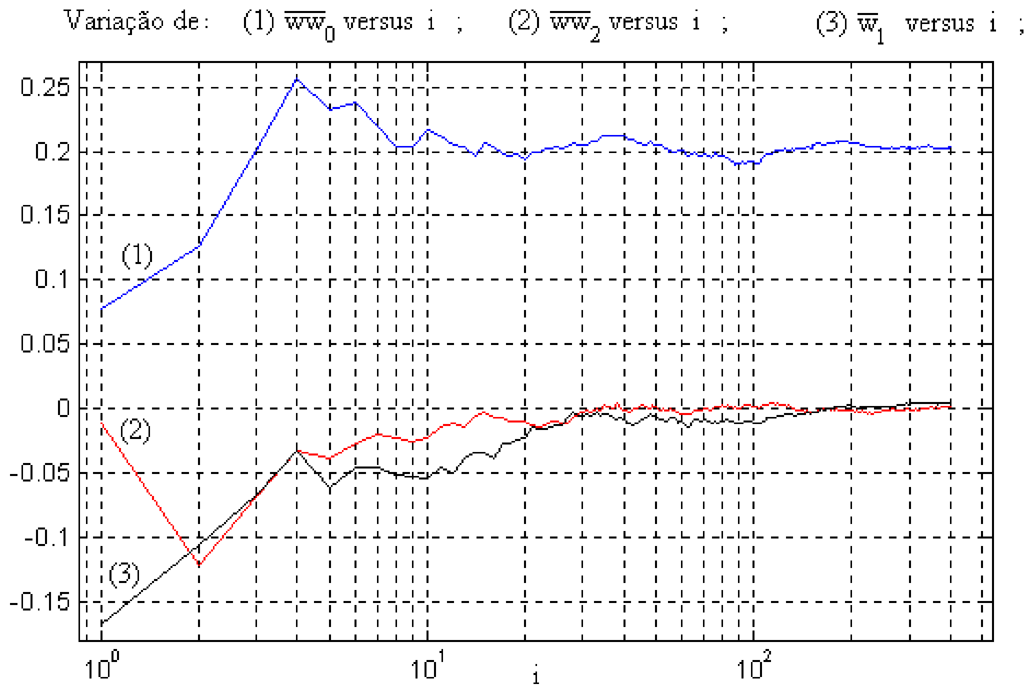
c)

Fig.4.9). Variação de parâmetros médios relacionados ao filtro D. O sinal desejado é obtido de um processo AR(1), onde a1 = -0.99. Os valores médios são calculados considerando que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \le i \le 401$ e o valor de NF igual a 20. (a) Variação de:

(1) $\overline{ww_0}$ *versus i* ; **(2)** $\overline{ww_1}$ *versus i* ; **(3)** $\overline{w_1}$ *versus i* (b) Gráficos relativos a variação dos valores médios:

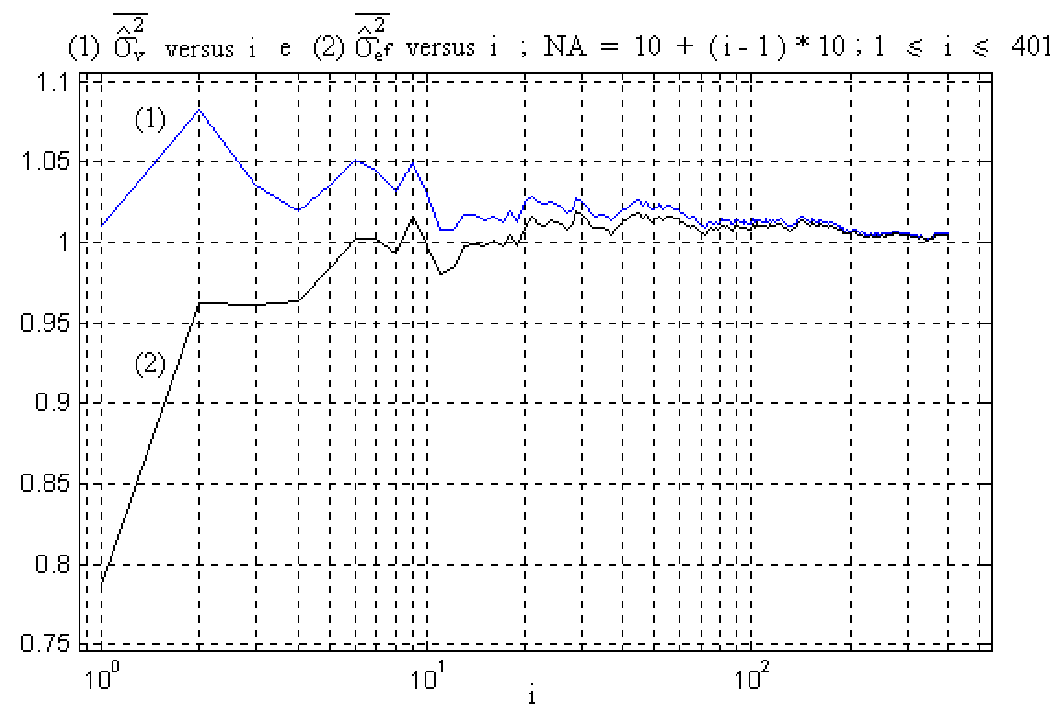
(1) $\overline{\hat{\sigma}_{\upsilon}^{2}}$ versus i (2) $\overline{\hat{\sigma}_{e^f}^{2}}$ versus i . (c) Variação de $\overline{er}$ *versus i* .

Os gráficos contidos nas Figs.(4.10), (4.11) e (4.12) são análogos aos da Fig.(4.7), (4.8) e (4.9), apenas com a ressalva de que o sinal desejado continua sendo um processo AR(1), com a alteração do coeficiente de $a_1 = -0.99$ para $a_1 = -0.2$.



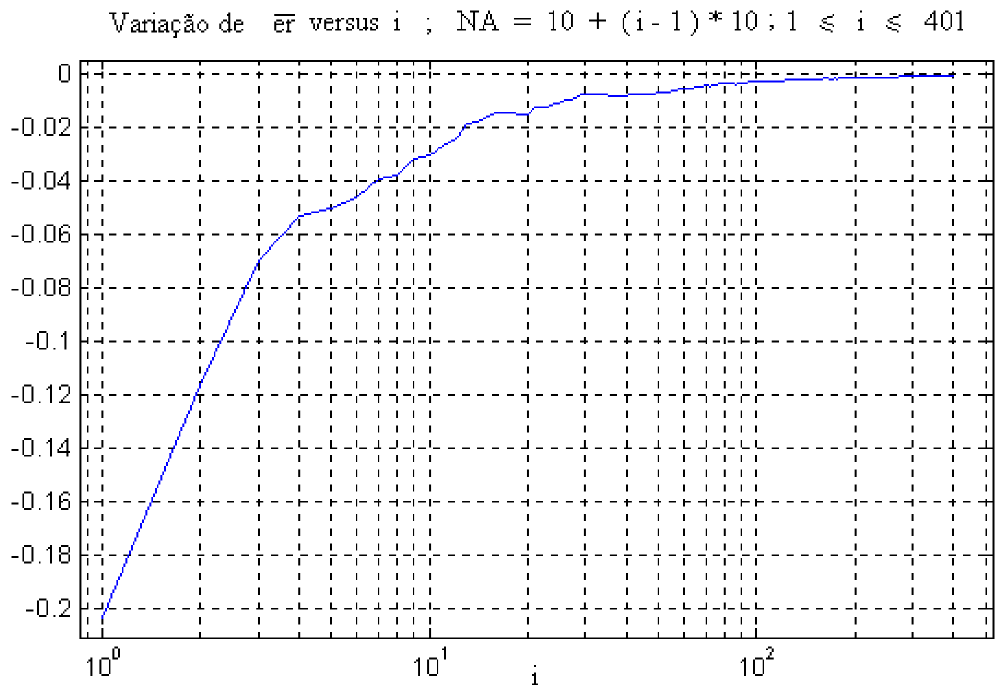
Variação de: (1) $\overline{ww}_0$ versus i ; (2) $\overline{ww}_2$ versus i ; (3) $\overline{w}_1$ versus i ;
(1)
(2)
(3)
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
10^0
10^1
10^2
i

a)



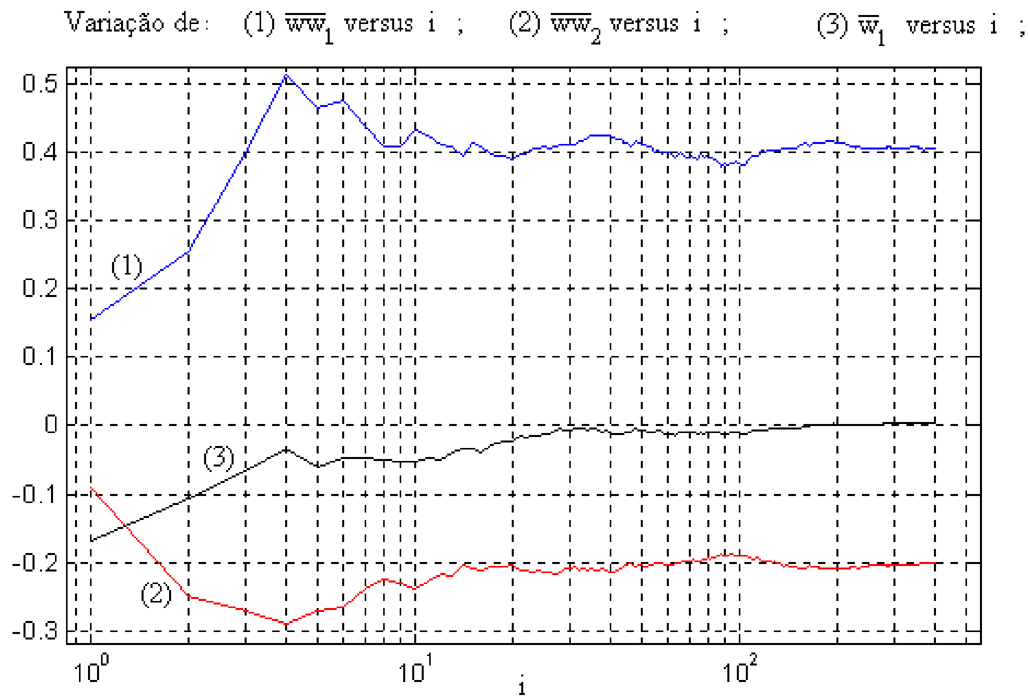
(1) $\overline{\hat{\sigma}_v^2}$ versus i e (2) $\overline{\hat{\sigma}_{ef}^2}$ versus i ; NA = 10 + (i - 1) * 10 ; 1 ≤ i ≤ 401
(1)
(2)
1.1
1.05
1
0.95
0.9
0.85
0.8
0.75
10^0
10^1
10^2
i

b)



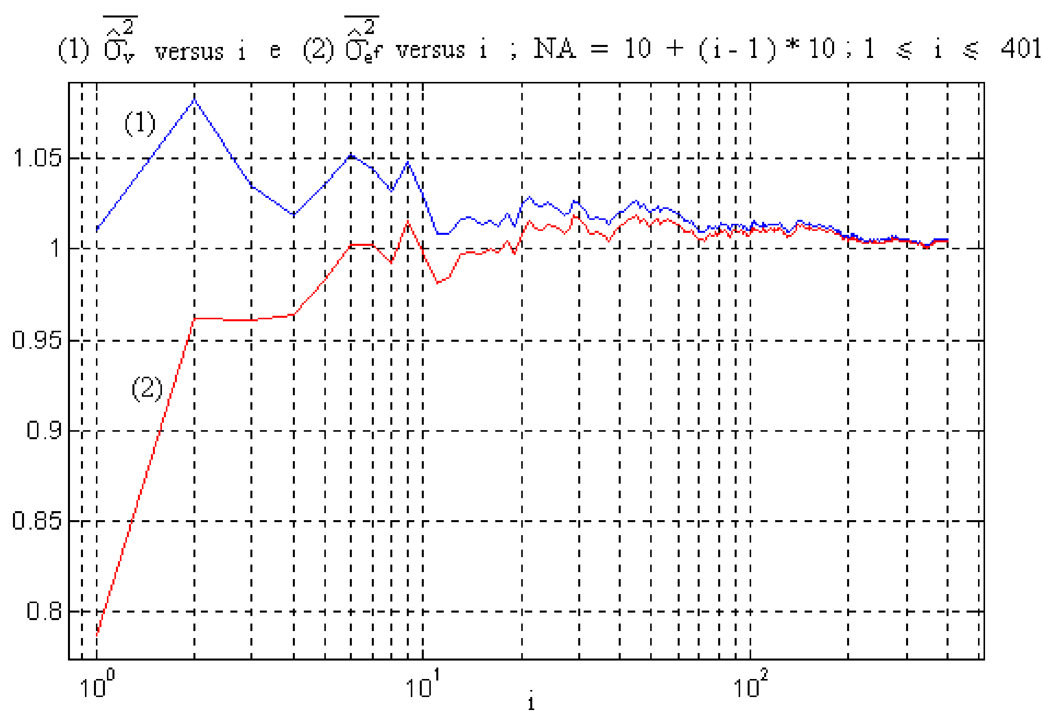
c)

Fig.4.10. Variação de parâmetros médios relacionados ao filtro D. O sinal desejado é obtido de um processo AR(1), onde a1 = -0.2. Os valores médios são calculados considerando que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \leq i \leq 401$ e o valor de NF igual a 20. (a) Variação de: (1) $\overline{ww_0}$ *versus i* ; (2) $\overline{ww_2}$ *versus i* ; (3) $\overline{w_1}$ *versus i* (b) Gráficos relativos a variação dos valores médios: (1) $\overline{\hat{\sigma}_\upsilon^2}$ versus i (2) $\overline{\hat{\sigma}_{e^f}^2}$ versus i . (c) Variação de $\overline{er}$ *versus i* .



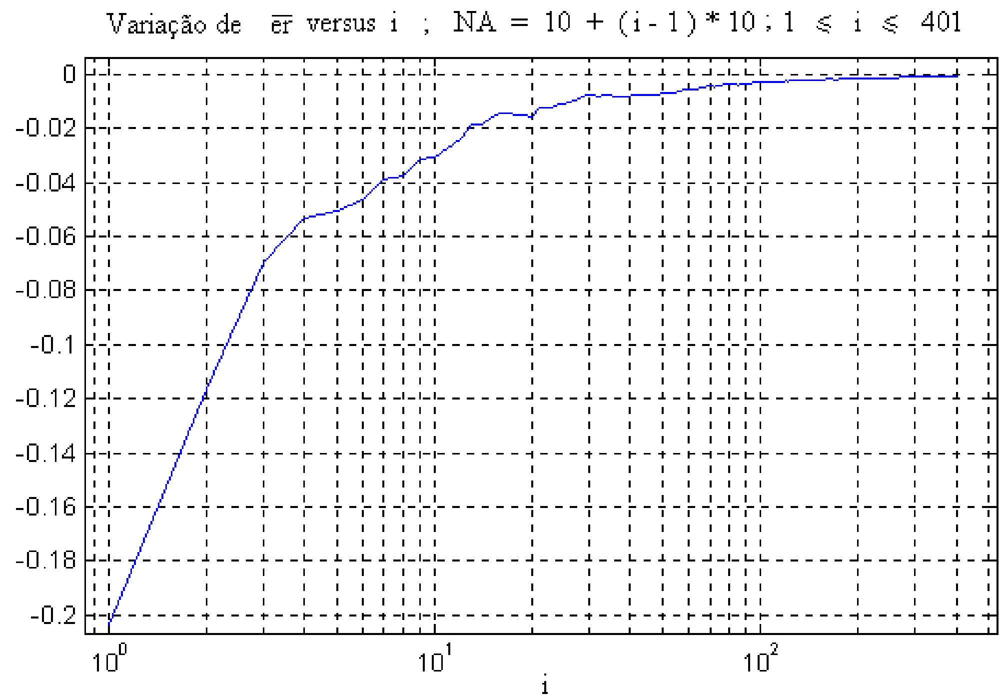
Variação de: (1) $\overline{ww}_1$ versus i ; (2) $\overline{ww}_2$ versus i ; (3) $\overline{w}_1$ versus i ;
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
(1)
(2)
(3)
10^0
10^1
10^2
i

a)



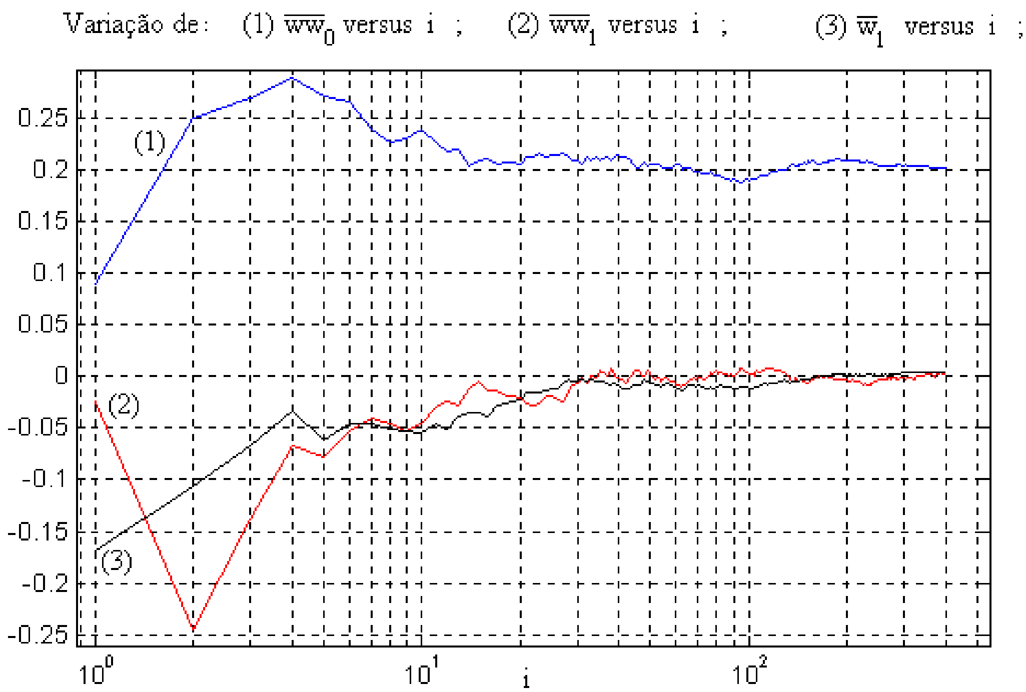
(1) $\overline{\hat{\sigma}^2_v}$ versus i e (2) $\overline{\hat{\sigma}^2_{ef}}$ versus i ; NA = 10 + (i - 1) * 10 ; 1 $\leqslant$ i $\leqslant$ 401
1.05
1
0.95
0.9
0.85
0.8
(1)
(2)
10^0
10^1
10^2
i

b)



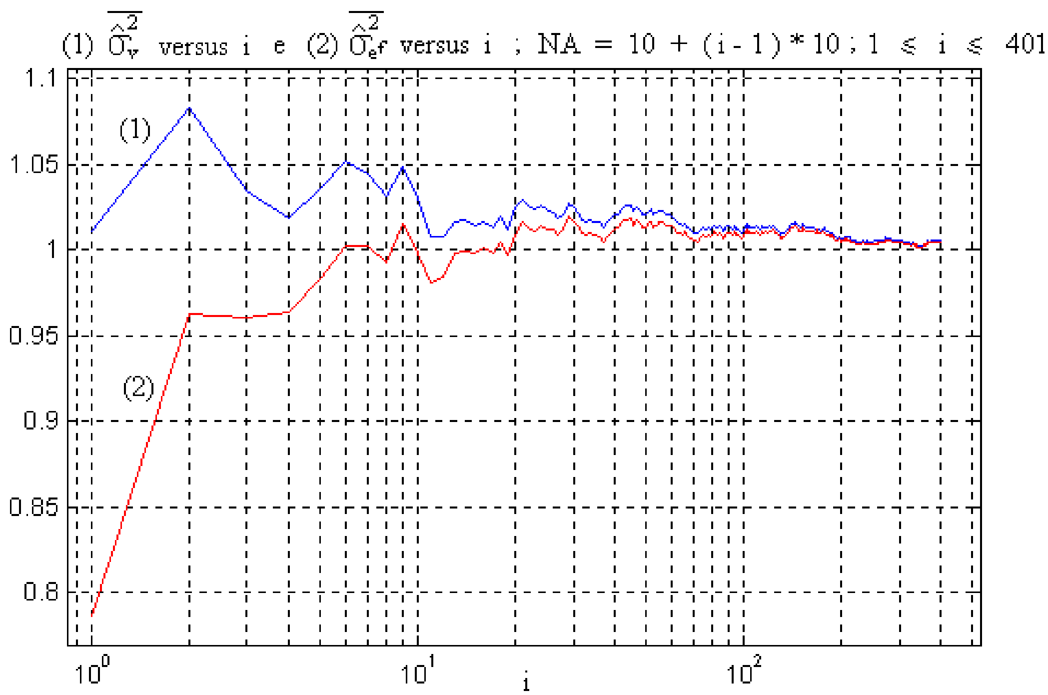
c)

Fig.4.11. Variação de parâmetros médios relacionados ao filtro D. O sinal desejado é obtidos de um processo AR(1), onde a1 = -0.2. Os valores médios são calculados considerando que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \le i \le 401$ e o valor de NF igual a 20. (a) Variação de: **(1)** $\overline{ww_1}$ *versus i* ; **(2)** $\overline{ww_2}$ *versus i* ; **(3)** $\overline{w_1}$ *versus i* (b) Gráficos relativos a variação dos valores médios: (1) $\overline{\hat{\sigma}_{\upsilon}^{2}}$ versus i (2) $\overline{\hat{\sigma}_{e^f}^{2}}$ versus i . (c) Variação de $\overline{er}$ *versus i* .



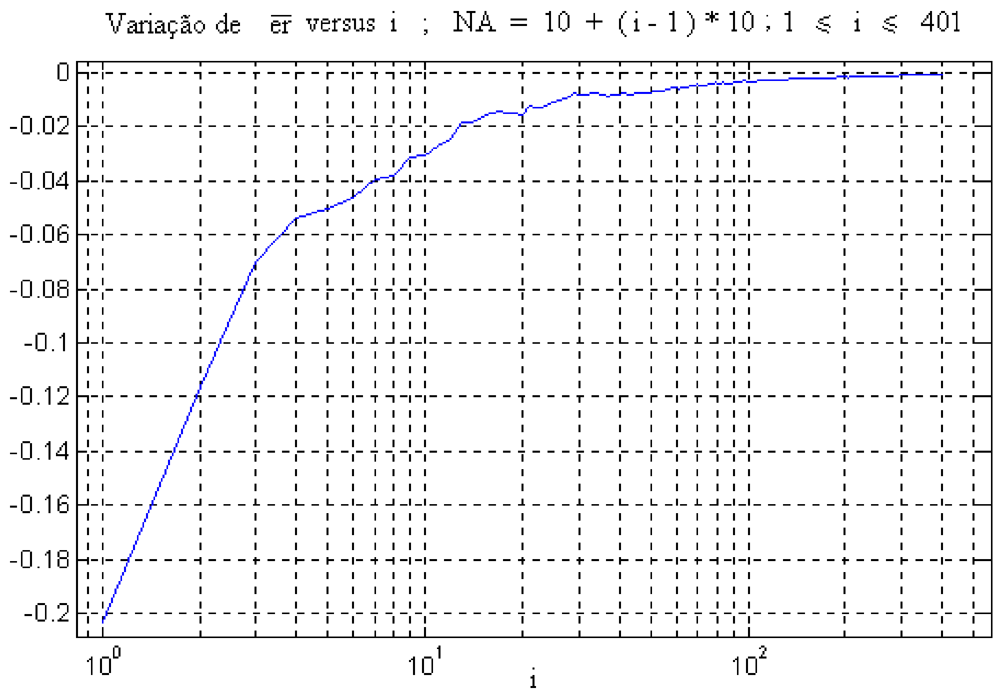
Variação de: (1) $\overline{ww}_0$ versus i ; (2) $\overline{ww}_1$ versus i ; (3) $\overline{w}_1$ versus i ;
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
(1)
(2)
(3)
10^0
10^1
10^2
i

a)



(1) $\overline{\hat{\sigma}_v^2}$ versus i e (2) $\overline{\hat{\sigma}_{ef}^2}$ versus i ; NA = 10 + (i - 1) * 10 ; 1 ≤ i ≤ 401
1.1
1.05
1
0.95
0.9
0.85
0.8
(1)
(2)
10^0
10^1
10^2
i

b)



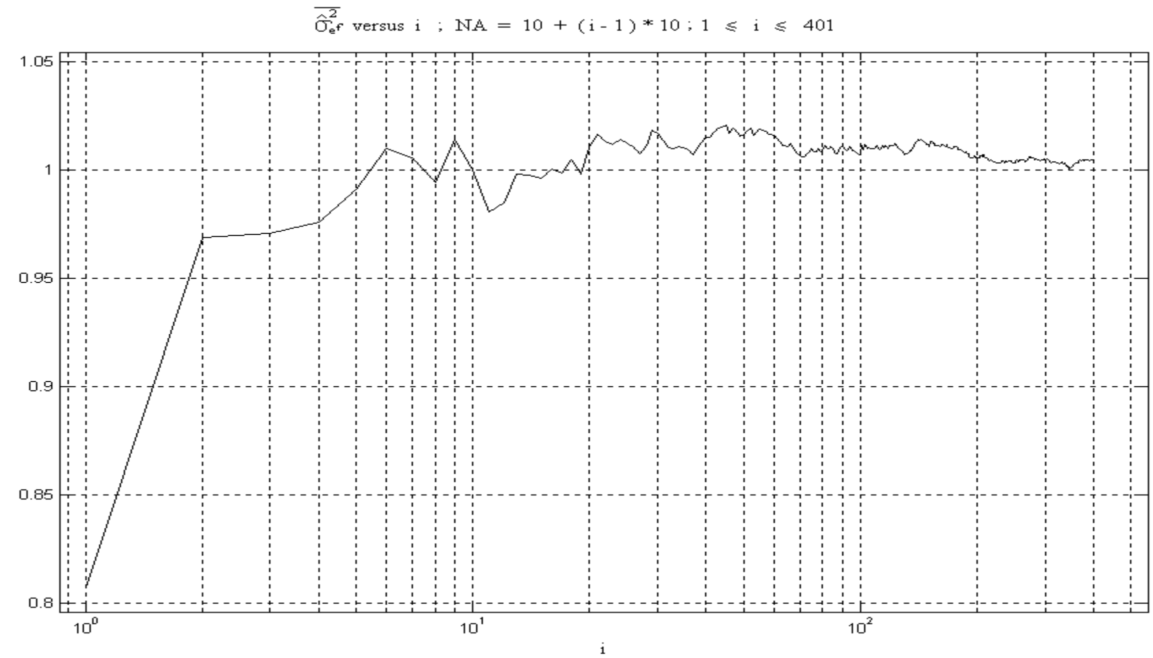
c)

Fig.4.12. Variação de parâmetros médios relacionados ao filtro D. O sinal desejado é obtido de um processo AR(1), onde a1 = -0.2. Os valores médios são calculados considerando que o número de amostras NA varia de 10 até 4010 amostras, o que corresponde a uma faixa de $1 \leq i \leq 401$ e o valor de NF igual a 20. (a) Variação de: (1) $\overline{ww_0}$ *versus* i ; (2) $\overline{ww_1}$ *versus* i ; (3) $\overline{w_1}$ *versus* i (b) Gráficos relativos a variação dos valores médios: (1) $\overline{\hat{\sigma}_\upsilon^2}$ versus i (2) $\overline{\hat{\sigma}_{e^f}^2}$ versus i . (c) Variação de $\overline{er}$ *versus* i .

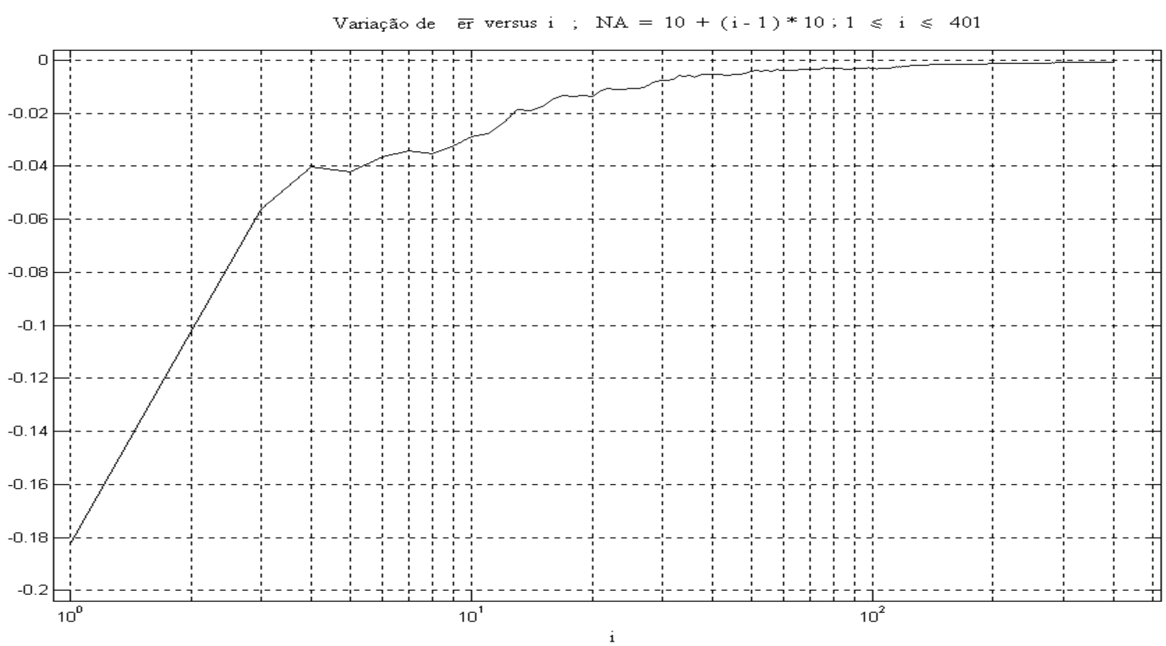
É importante observar que:

1) No contexto de predição linear progressiva visto anteriormente, atribuiu-se ao sinal desejado $d(n)$ um processo AR(1). Através da aplicação do sinal desejado com um retardo de uma amostra a um filtro de predição progressivo, obteve-se em sua saída o sinal estimado $\hat{d}(n)$.
2) Uma das soluções encontradas foi a que utilizou a filtragem de Wiener cujo resultado é apresentado na Fig.(4.5.a). A outra utilizou o sistema de equações D ou filtro D, gerando três soluções, apresentadas nas Figs.(4.7.a), (4.8.a) e (4.9.a).

3) Apesar de se utilizar diferentes sinais decompostos no sistema D, as variações de $\overline{\hat{\sigma}^2_{e^f}}$ versus i apresentadas pelas Figs.(4.7.b), (4.8.b) e (4.9.b) são praticamente idênticas conforme demonstra a Fig.(4.13.a). Portanto, a variação de $\overline{er}$ versus i em cada um dos gráficos apresentados nas Figs.(4.7.c), (4.8.c) e (4.9.c) também serão praticamente iguais, ou seja, traçando essas curvas em um só gráfico, Fig.(4.13.b), elas aparecerão todas sobrepostas.
4) Pode-se acrescentar ainda que mesmo empregando dois métodos diferentes (gerando diferentes soluções para um mesmo problema), constatou-se que com o aumento do número de amostras NA obteve-se um comportamento muito próximo entre os métodos, se comparada as variações de $\overline{\hat{\sigma}^2_{e^f}}$ versus i dada pela Fig.(4.14.a) e $\overline{er}$ versus i apresentada pela Fig.(4.14.b), obtidas por cada um dos métodos.
5) Modificou-se o processo AR(1) através da alteração do coeficiente $a_1 = -0.99$ para $a_1 = -0.2$, isto é, passou-se de um sinal muito correlacionado para outro com pouca correlação. Similarmente a observação 2, foi encontrada uma solução utilizando-se a filtragem de Wiener e o resultado foi apresentado na Fig.(4.6.a). Um conjunto de soluções foi obtido pelo emprego das equações D (ou sistema D) e os resultados foram apresentados nas Figs.(4.10.a), (4.11.a) e (4.12.a). A Fig.(4.15.a) mostra que as variações de $\overline{\hat{\sigma}^2_{e^f}}$ versus i apresentadas pelas Figs.(4.10.b), (4.11.b) e (4.12.b) são praticamente idênticas e portanto apresentam curvas sobrepostas. O mesmo pode-se dizer com relação as variações de $\overline{er}$ versus i exibidas pelas Figs.(4.10.c), (4.11.c) e (4.12.c) que apresentam variações bastante próximas resultando na Fig.(4.15.b).
6) De forma análoga às Figs.(4.14.a) e (4.14.b), tem-se as variações de $\overline{\hat{\sigma}^2_{e^f}}$ versus i dada pela Fig.(4.16.a) e de $\overline{er}$ versus i dada pela Fig.(4.16.b) obtidas por cada um dos métodos, dado que o processo AR(1) sofreu uma alteração no coeficiente $a_1 = -0.99$ para $a_1 = -0.2$.



a)

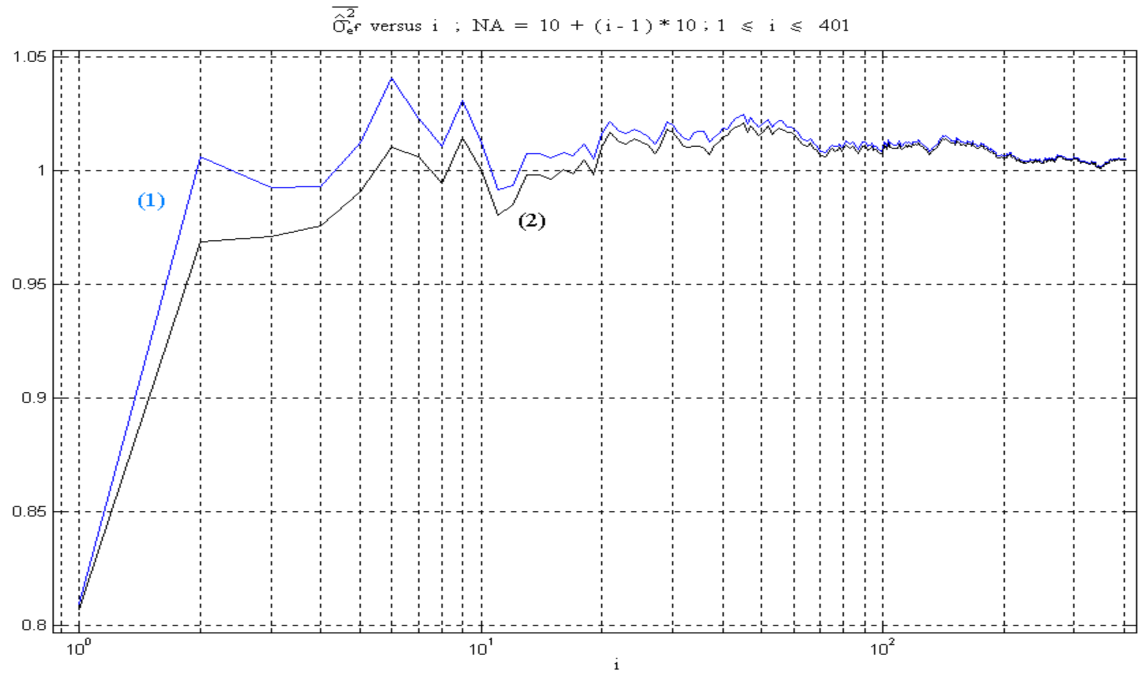


b)

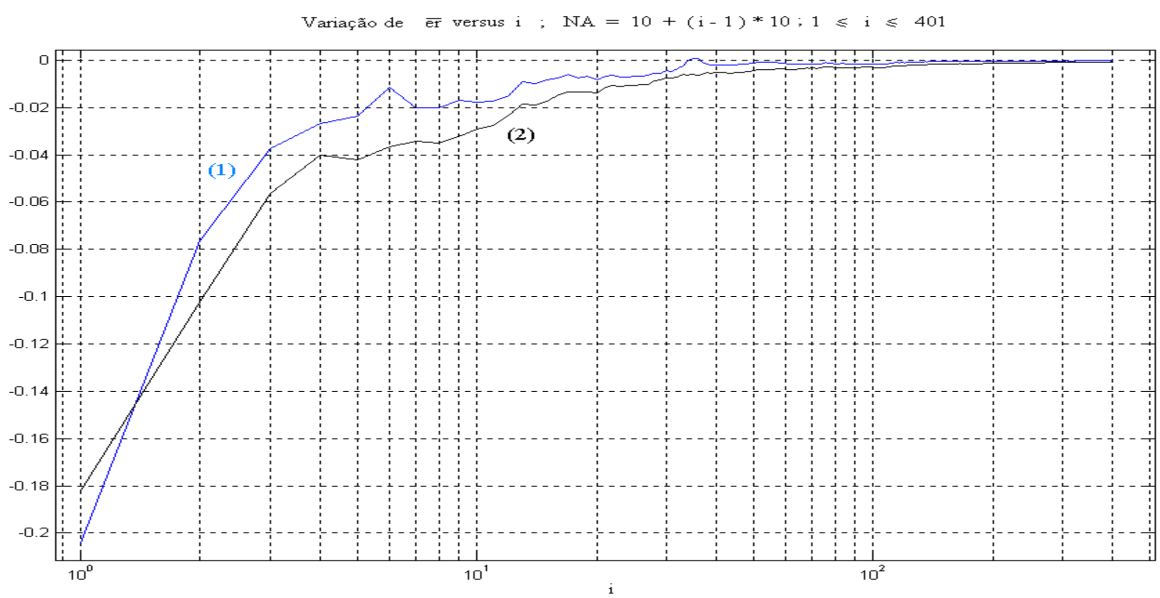
Fig.4.13. Para os sistemas dados pelas Eqs.(4.27.a), (4.29.a), (4.30.a) e processo AR(1) com $a_1 = -0.99$, tem-se que:

a)Sobreposição dos gráficos relativos a variação dos valores médios de $\overline{\hat{\sigma}^2_{e^f}}$ versus i

b)Sobreposição dos gráficos relativos a variação dos valores médios de $\overline{er}$ *versus i* .

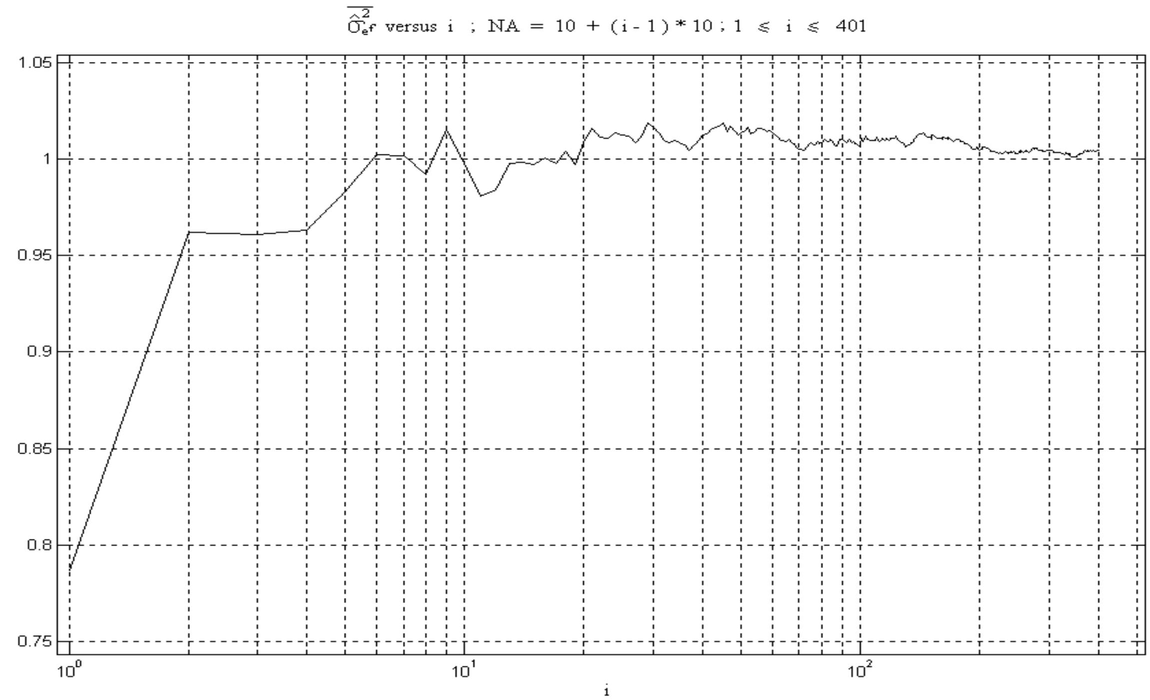


a)

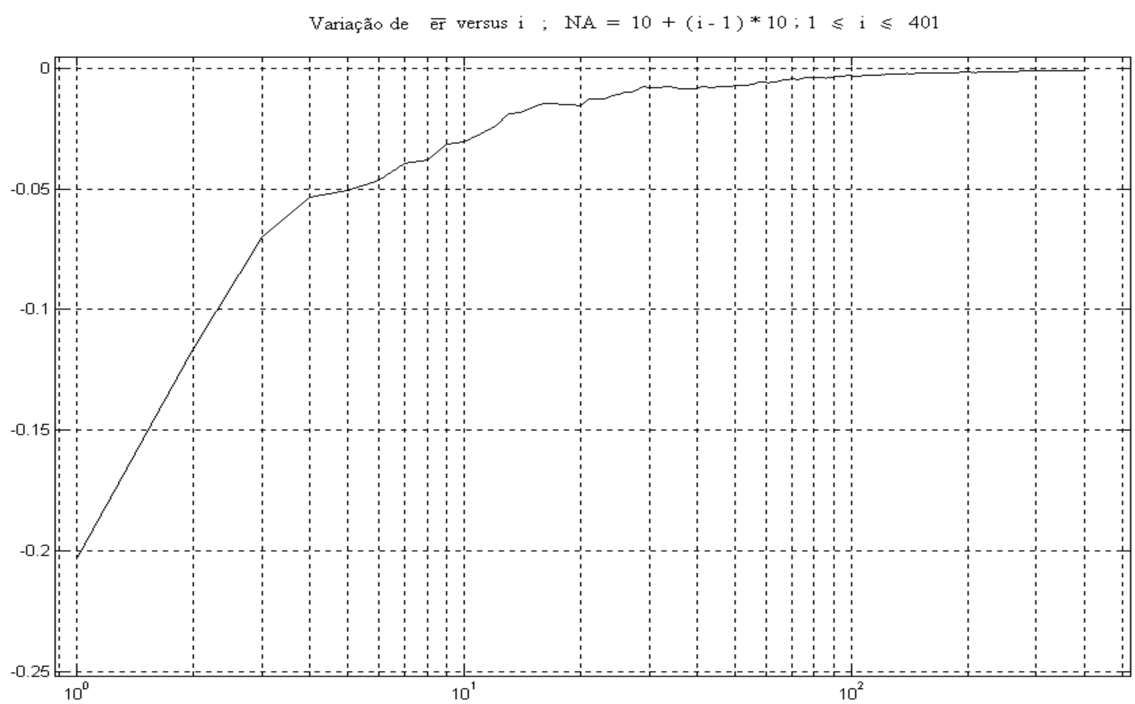


b)

Fig.4.14. Comparação entre parâmetros médios obtidas pelas equações de Wiener-Hopf e pelo sistema D, considerando-se um processo AR(1) com $a_1 = -0.99$. a) A curva (1) representa a variação dos valores médios de $\overline{\hat{\sigma}^2_{e^f}}$ versus i utilizando-se a equação de Wiener-Hopf, enquanto a curva (2) é a variação dos valores médios de $\overline{\hat{\sigma}^2_{e^f}}$ versus i relacionada ao sistema D. b) A curva (1) representa a variação dos valores médios de $\overline{er}$ *versus i* utilizando-se a equação de Wiener-Hopf, enquanto a curva (2) é a variação dos valores médios de $\overline{er}$ *versus i* versus i relacionada ao sistema D.



a)

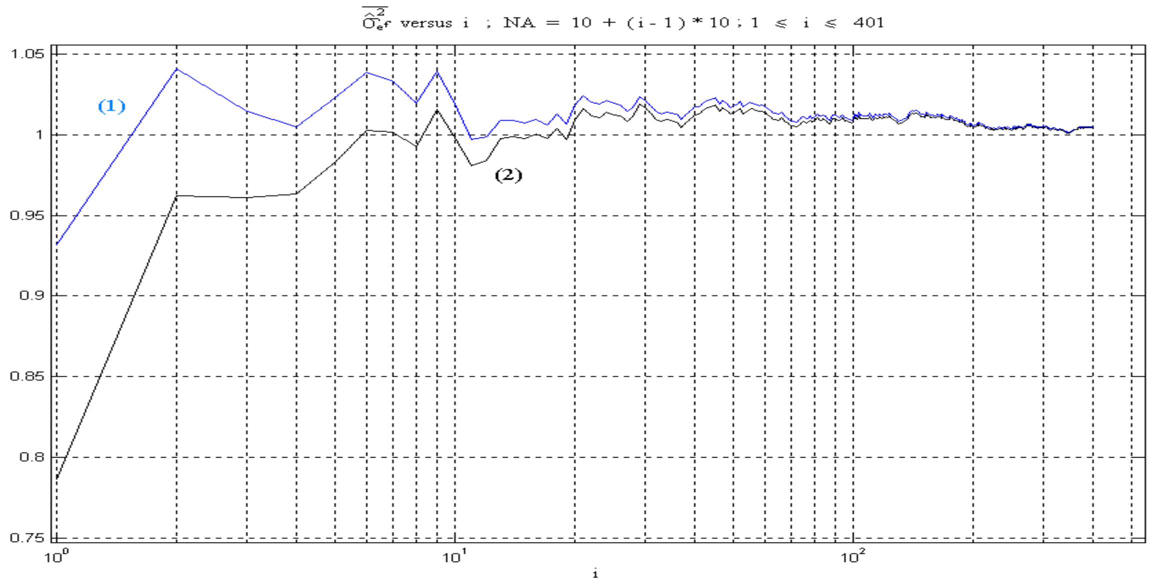


b)

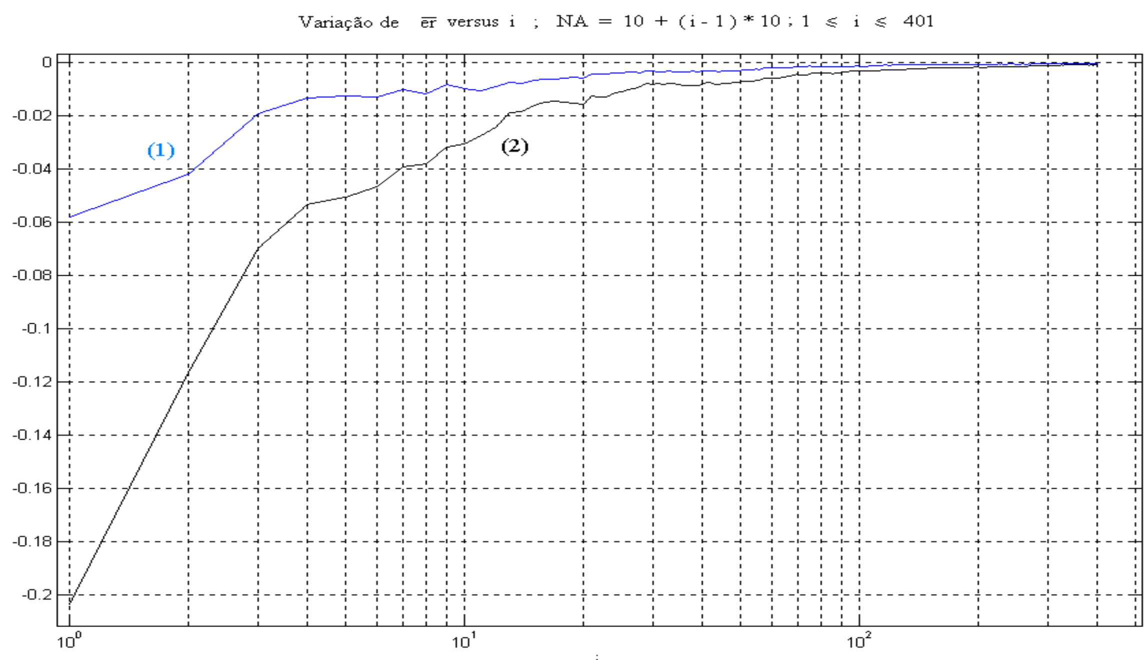
Fig.4.15. Para os sistemas dados pelas Eqs.(4.27.a), (4.29.a), (4.30.a) e processo AR(1) com a_1 = -0.2, tem-se que:

a) Sobreposição dos gráficos relativos a variação dos valores médios de $\overline{\hat{\sigma}^2_{ef}}$ versus i .

b) Sobreposição dos gráficos relativos a variação dos valores médios de $\overline{er}$ *versus i* .



a)



b)

Fig.4.16. Comparação entre parâmetros médios obtidas pelas equações de Wiener-Hopf e pelo sistema D, considerando-se um processo AR(1) com $a_1 = -0.2$. a) A curva (1) representa a variação dos valores médios de $\overline{\hat{\sigma}^2_{e^f}}$ versus i utilizando-se a equação de Wiener-Hopf, enquanto a curva (2) é a variação dos valores médios de $\overline{\hat{\sigma}^2_{e^f}}$ versus i relacionada ao sistema D. b) A curva (1) representa a variação dos valores médios de $\overline{er}$ *versus i* utilizando-se a equação de Wiener-Hopf, enquanto a curva (2) é a variação dos valores médios de $\overline{er}$ *versus i* relacionada ao sistema D.

CAPÍTULO 5

Implementação do Algoritmo Adaptativo LMS Incorporando-se a Decomposição do Sinal de Entrada

5.1 Introdução

Propõe-se neste capítulo, apresentar uma estrutura de filtro adaptativo baseado no algoritmo LMS, que incorpore o processo de decomposição do sinal de entrada apresentado no Capítulo 3. Essa inclusão no algoritmo LMS pode ser implementada de duas formas, a primeira é feita utilizando-se diretamente os sinais obtidos do processo de decomposição, gerando-se o algoritmo adaptativo denominado de LMS com decomposição do sinal de entrada, ou simplesmente, D-LMS; a segunda é feita aplicando-se sobre os sinais decompostos a Transformada Discreta Seno, gerando-se o algoritmo LMS no domínio da freqüência associado à Transformada Seno Discreta com Pré-rotação de Eixos, ou simplesmente, DSTr-LMS.

A fim de expor esses tópicos com clareza, será apresentada uma breve introdução sobre a origem e operação do algoritmo LMS no domínio do tempo e da freqüência, necessários para uma melhor compreensão e comparação com os algoritmos D-LMS e DSTr-LMS.

5.2 Solução Iterativa das Equações de Wiener-Hopf

As subseções seguintes tratam da solução iterativa das equações Wiener-Hopf. Inicia-se apresentando o método do gradiente descendente e a sua aplicação na solução das equações de Wiener-Hopf. Em seguida, expõe-se a geração do algoritmo LMS tradicional partindo-se da equação do gradiente descendente. Após isso, apresentam-se algumas variações existentes do algoritmo LMS no domínio do tempo e da freqüência.

5.2.1Algoritmo Gradiente Descendente Utilizado na Solução das Equações de Wiener-Hopf

A Eq.(2.56) calcula o conjunto de coeficientes do filtro transversal que minimiza o índice de desempenho J. Para se obter esse conjunto, necessita-se conhecer previamente a função autocorrelação $\underline{\mathbf{R}}$ ou $\underline{\mathbf{R}}_x$, a função correlação cruzada $\underline{\mathbf{p}}$ ou $\underline{\mathbf{r}}_{xd}$ e por último, implementar a inversão da matriz $\underline{\mathbf{R}}$. De posse dessas informações, são citadas algumas desvantagens [16,39] de ordem prática inerentes a essa solução:

a)Deve-se ter o conhecimento prévio da estatística do sinal envolvido, uma vez que as funções $\underline{\mathbf{R}}$ e $\underline{\mathbf{p}}$ são utilizadas para o cálculo dos coeficientes ótimos;

b)Caso a estatística dos sinais envolvidos sejam desconhecidas, como ocorre em muitos casos práticos, então, para a solução da Eq.(2.56) torna-se necessária a estimação de $\underline{\mathbf{R}}$ e $\underline{\mathbf{p}}$ através das amostras dos processos envolvidos, como implementado por exemplo, pela Eq.(4.22). O grande empecilho de se fazer essa estimação é o atraso acarretado de N amostras;

c) Mesmo considerando-se que d(n) e x(n) sejam estacionários, um valor elevado do número de coeficientes do filtro pode tornar impraticável a solução direta via Eq.(2.56) e nesse caso, a solução obtida por um algoritmo adaptativo torna-se bastante atrativa;

d)Ainda considerando-se que os sinais envolvidos sejam estacionários, tem-se que, caso $\underline{\mathbf{R}}$ seja uma matriz mal condicionada (quase singular), então a solução pelas equações de Wiener-Hopf será afetada por erros de arredondamento;

e)A inversão de matrizes é uma operação que exige muito tempo de processamento;

f)Se os processos representados por d(n) e x(n) não forem estacionários, então tanto $\underline{\mathbf{R}}$ quanto $\underline{\mathbf{p}}$ estarão se modificando com o tempo, logo, os valores dos coeficientes que minimizam $J=\mathbf{E}[e^2(n)]$ serão variáveis com o tempo. Portanto, exige-se constantes repetições do cálculo dos coeficientes através da Eq.(2.56).

O algoritmo gradiente descendente ou algoritmo da máxima descida [18, 19, 39, 40] resolve de forma iterativa as equações de Wiener-Hopf sem a necessidade de inverter a matriz $\underline{\mathbf{R}}$. Ele é um método que utiliza, em sua busca pela solução de Wiener, a seguinte equação recursiva [19]:

$$\underline{\mathbf{w}}(n+1) = \underline{\mathbf{w}}(n) + \frac{1}{2}\mu[-\nabla J(n)] \tag{5.1}$$

onde,

$\underline{\mathbf{w}}(n+1)$ é o vetor peso cujos valores correspondem ao instante n+1.

$\underline{\mathbf{w}}(n)=[\, w_0(n), w_1(n), ..., w_{(N-1)}(n)\,]^T$ é o vetor peso cujos valores correspondem ao instante n .
μ é uma constante real positiva que determina a taxa de convergência e a estabilidade.
∇ J(n) é o vetor gradiente cujos valores correspondem ao instante n.

De acordo com a Eq.(5.1), a atualização do vetor peso no instante n+1 é calculada utilizando-se os valores do vetor peso e do vetor gradiente no instante n e dessa forma o processo é repetido à medida que "n" cresce. No início do processo, costuma-se adotar valores nulos para o vetor peso $\underline{\mathbf{w}}$(n), caso não se tenha uma estimativa da possível localização onde ocorre o valor quadrático médio mínimo. As correções no vetor peso são feitas na direção negativa do vetor gradiente ∇ J(n) e a constante μ denominada de tamanho ou passo do degrau, controla a intensidade dessa correção.

O vetor de pesos $\underline{\mathbf{w}}$ na Eq.(2.57) é invariável com o tempo e considerando-se que ele assuma o seu valor ótimo, então o índice de desempenho J atinge o valor mínimo. Substituindo-o pelo vetor peso variável $\underline{\mathbf{w}}$(n), cria-se um correspondente índice de desempenho J(n) variável com o instante n, sendo expresso por:

$$J(n) = \sigma_d^2 - 2\underline{\mathbf{w}}^T(n)\underline{\mathbf{p}} + \underline{\mathbf{w}}^T(n)\ \underline{\mathbf{R}}\ \underline{\mathbf{w}}(n) \tag{5.2}$$

logo, o valor do vetor gradiente no instante "n", será dado por:

$$\nabla J(n) = \frac{\partial J(n)}{\partial \underline{\mathbf{w}}(n)} \tag{5.3.a}$$

$$\nabla J(n) = \left[\frac{\partial J(n)}{\partial w_0(n)} \quad \frac{\partial J(n)}{\partial w_1(n)} \quad \cdots \quad \frac{\partial J(n)}{\partial w_{(N-1)}(n)} \right]^T \tag{5.3.b}$$

$$\nabla J(n) = -2\underline{\mathbf{p}} + 2\underline{\mathbf{R}}\ \underline{\mathbf{w}}(n) \tag{5.3.c}$$

ou por:

$$\nabla J(n) = \frac{\partial \mathbf{E}[e^2(n)]}{\partial \underline{\mathbf{w}}(n)} \tag{5.4.a}$$

$$\nabla J(n) = \mathbf{E}[2e(n)\frac{\partial}{\partial \underline{\mathbf{w}}(n)}[d(n) - \underline{\mathbf{w}}^T(n)\underline{\mathbf{x}}(n)]] \tag{5.4.b}$$

$$\nabla J(n) = -\mathbf{E}[2e(n)\underline{\mathbf{x}}(n)] \qquad (5.4.c)$$

Substituindo-se a Eq.(5.3.c) na Eq.(5.1), tem-se o algoritmo do gradiente descendente expresso por:

$$\underline{\mathbf{w}}(\mathrm{n}+1) = \underline{\mathbf{w}}(\mathrm{n}) + \mu[\underline{\mathbf{p}} - \underline{\mathbf{R}}\ \underline{\mathbf{w}}(\mathrm{n})], \qquad \mathrm{n} = 0,1,2,\cdots \qquad (5.5)$$

Uma outra expressão é obtida substituindo-se a Eq.(5.4.c) na Eq.(5.1), logo, tem-se que:

$$\underline{\mathbf{w}}(\mathrm{n}+1) = \underline{\mathbf{w}}(\mathrm{n}) + \mu\mathbf{E}[\mathrm{e}(\mathrm{n})\underline{\mathbf{x}}(\mathrm{n})], \qquad \mathrm{n} = 0,1,2,\cdots \qquad (5.6)$$

5.2.2 Algoritmos LMS no Domínio do Tempo e da Freqüência

Observando a Eq.(5.5), percebe-se que esse algoritmo continua necessitando do conhecimento prévio da função correlação cruzada **p** e da função autocorrelação **R**, portanto atrasos ocorrerão a fim de se estimar esses parâmetros.

O algoritmo "*Least Mean Square*" [18, 19, 20, 39] é gerado a partir do algoritmo gradiente descendente. A diferença é que o LMS faz uma estimativa instantânea da função autocorrelação **R** e da correlação cruzada **p**, isto é, ele utiliza para fazer essa estimação apenas a amostra do sinal desejado d(n) no instante "n", e as amostras presentes nas derivações do filtro fornecidas por **x**(n), logo:

$$\hat{\underline{\mathbf{R}}}(n) = \underline{\mathbf{x}}(n)\underline{\mathbf{x}}^T(n) \qquad (5.7)$$

e

$$\hat{\underline{\mathbf{p}}}(n) = \underline{\mathbf{x}}(n)d(n) \qquad (5.8)$$

onde $\hat{\underline{\mathbf{R}}}(n)$ é a estimativa da função autocorrelação $\underline{\mathbf{R}}$ e $\hat{\underline{\mathbf{p}}}(n)$ é a estimativa da correlação cruzada $\underline{\mathbf{p}}$. Substituindo-se $\underline{\mathbf{R}}$ e $\underline{\mathbf{p}}$ na Eq.(5.5), respectivamente por $\hat{\underline{\mathbf{R}}}(n)$ e $\hat{\underline{\mathbf{p}}}(n)$, tem-se uma nova fórmula de atualização do vetor peso que se baseia em valores estimados sendo expressa por:

$$\hat{\underline{\mathbf{w}}}(n+1) = \hat{\underline{\mathbf{w}}}(n) + \mu[\underline{\mathbf{x}}(n)d(n) - \underline{\mathbf{x}}(n)\underline{\mathbf{x}}^T(n)\ \hat{\underline{\mathbf{w}}}(n)] \tag{5.9.a}$$

$$\hat{\underline{\mathbf{w}}}(n+1) = \hat{\underline{\mathbf{w}}}(n) + \mu\ \underline{\mathbf{x}}(n)\ [\ d(n) - \underline{\mathbf{x}}^T(n)\ \hat{\underline{\mathbf{w}}}(n)] \tag{5.9.b}$$

$$\hat{\underline{\mathbf{w}}}(n+1) = \hat{\underline{\mathbf{w}}}(n) + \mu\ e(n)\underline{\mathbf{x}}(n) \tag{5.9.c}$$

A Eq.(5.9.c) utiliza apenas a amostra do erro no instante n e as amostras presentes nas derivações do filtro fornecidas por $\underline{\mathbf{x}}$(n), em outras palavras, com a estimação instantânea eliminou-se o operador esperança estatística **E** presente na Eq.(5.6), obtendo-se a Eq.(5.9.c).
A Fig.(5.1) apresenta o diagrama em blocos de um filtro adaptativo baseado no algoritmo LMS tradicional. A estrutura é formada por um filtro FIR cujos coeficientes do vetor peso $\underline{\mathbf{w}}$(n) são atualizados utilizando-se a Eq.(5.9.c). As amostras de entrada de cada derivação do filtro transversal são multiplicadas pelos correspondentes coeficientes w_i(n) e os produtos resultantes são somados, para formarem o sinal de saída y(n).

A fim de garantir convergência do processo adaptativo, a seguinte condição [19,39] deve ser satisfeita:

$$0 < \mu < \frac{2}{tr(\underline{\mathbf{R}})} \tag{5.10}$$

onde, $tr(\underline{\mathbf{R}})$ denota o traço da matriz $\underline{\mathbf{R}}$.

Uma das dificuldades encontradas no algoritmo LMS é a especificação do valor do parâmetro μ, uma vez que o mesmo depende do conhecimento da matriz $\underline{\mathbf{R}}$ e essa é desconhecida. No entanto, tratando-se de processo de entrada estacionário tem-se que a soma dos elementos da diagonal principal da matriz $\underline{\mathbf{R}}$ é igual ao número de derivações vezes o valor quadrático médio do sinal de entrada [19, 39], isto é, $tr(\underline{\mathbf{R}})=N\sigma_x^2$, logo, a condição de convergência dada pela Eq.(5.10) é reescrita como:

$$0 < \mu < \frac{2}{N\sigma_x^2} \tag{5.11}$$

onde, $N\sigma_x^2$ é igual a soma do valor quadrático médio de cada uma das derivações do filtro. O valor quadrático médio dado por $\sigma_x^2=\mathbf{E}[x^2(n)]$ pode ser estimado através de uma média temporal dada por:

$$\hat{\sigma}_x^2 = \hat{\mathbf{E}}[x^2(n)] = \frac{1}{N}\sum_{k=0}^{N-1} x^2(n-k) \tag{5.12.a}$$

$$\hat{\sigma}_x^2 = \frac{1}{N}\underline{\mathbf{x}}^T(n)\underline{\mathbf{x}}(n) \tag{5.12.b}$$

$$\hat{\sigma}_x^2 = \frac{1}{N}\left\|\underline{\mathbf{x}}(n)\right\|^2 \tag{5.12.c}$$

onde, $\left\|\underline{\mathbf{x}}(n)\right\|^2$ representa o quadrado da norma Euclidiana. Substituindo-se σ_x^2 presente na Eq.(5.11) por sua estimativa dada pela Eq.(5.12.c), tem-se um novo limite de convergência para μ, dado por:

$$0 < \mu < \frac{2}{\mathbf{x}^T(n)\mathbf{x}(n)} \tag{5.13}$$

Uma variação do LMS é o algoritmo LMS normalizado (NLMS), que é gerado a partir da substituição de μ na Eq.(5.9.c) por um passo de degrau variável $\mu(n)$, cuja expressão incorpora o limite de convergência dado pela Eq.(5.13). Logo, tem-se que :

$$\mu(n) = \frac{\alpha}{\sum_{k=0}^{N-1} x^2(n-k) + \delta} \tag{5.14}$$

onde $0<\alpha<2$ e δ é um pequeno valor positivo, necessário na prevenção de uma amplificação excessiva em momentos que a somatória tende a zero, isto é, quando o nível do sinal de entrada passa por um desvanecimento. Quando o sinal passa por valores elevados, a amplificação excessiva do vetor gradiente "*gradient noise amplification*" é diminuída com a normalização por $\|\underline{\mathbf{x}}(n)\|^2$. Portanto, a Eq.(5.14) denota uma normalização de μ em função do quadrado da norma Euclidiana, tornando o algoritmo NLMS robusto às variações em amplitude do processo de entrada. O sinal de voz é um exemplo de sinal cuja magnitude pode variar em uma ampla faixa de valores desde picos elevados, para momentos de silêncio.

A Fig.(5.2) apresenta uma outra variação do algoritmo LMS, denominado de algoritmo LMS no domínio da transformada [19, 30, 31, 32, 35, 56], ou simplesmente, TDAF. A operação desse algoritmo é muito parecida com a do LMS convencional. A diferença básica é que uma transformada ortogonal $\mathbb{T}$ de N pontos, por exemplo a DST ou a DCT fornecidas no Apêndice A, é aplicada sobre as amostras do sinal de entrada presentes nas N derivações do filtro transversal, gerando saídas menos correlacionadas.

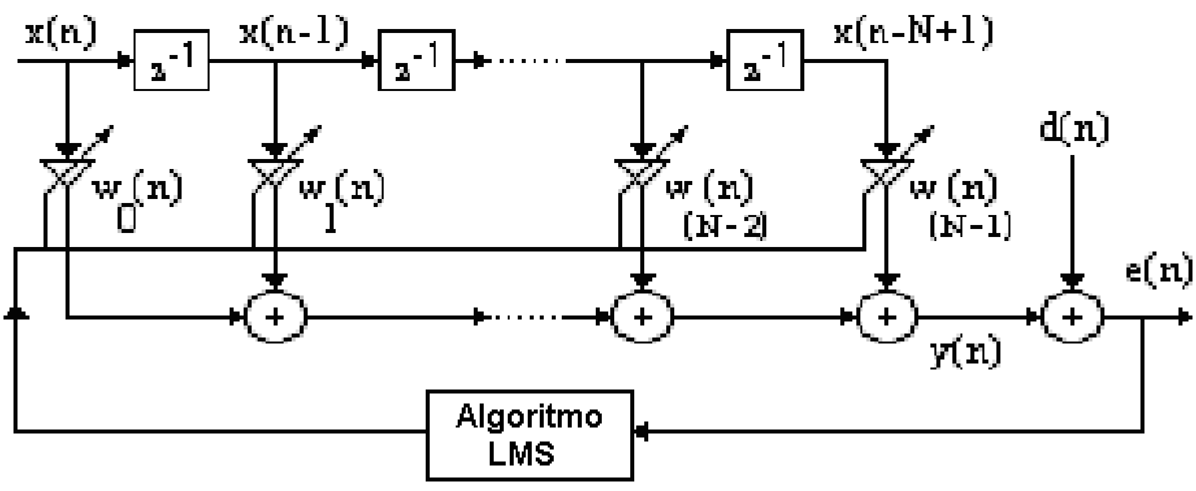


Fig.5.1. Diagrama em blocos do filtro adaptativo FIR utilizando o algoritmo LMS.

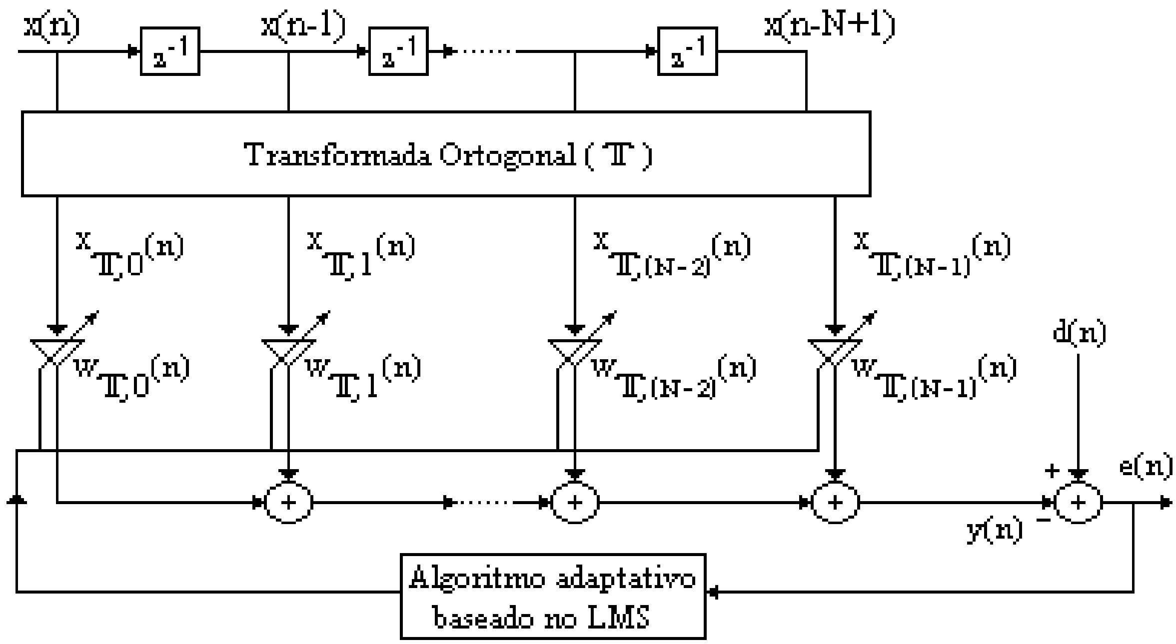


Fig.5.2. Diagrama em blocos do filtro adaptativo FIR utilizando o algoritmo LMS no domínio da freqüência.

De fato, a transformada ortogonal age como um banco de filtros, produzindo N sinais localizados em diferentes faixas de freqüências. Caso os filtros passa-faixas fossem ideais e sem sobreposições de faixas, então as saídas transformadas seriam completamente não correlatas. Posteriormente, as saídas transformadas são introduzidas em um filtro transversal cujos coeficientes são atualizados utilizando-se a seguinte equação:

$$\underline{\mathbf{w}}_{\mathrm{T}}(n+1) = \underline{\mathbf{w}}_{\mathrm{T}}(n) + \underline{\boldsymbol{\mu}}(n)e(n)\underline{\mathbf{x}}_{\mathrm{T}}(n) \tag{5.15}$$

onde $\underline{\mathbf{x}}(n)$ representa o vetor das amostras presentes na linha de atraso do filtro expresso por:

$$\underline{\mathbf{x}}(n) = [x(n), x(n-1), \cdots, x(n-N+1)]^T \tag{5.16}$$

$\underline{\mathbf{x}}_{\mathrm{T}}(n)$ denota o vetor constituído das amostras de $\underline{\mathbf{x}}(n)$ transformadas por $\mathbb{T}$, sendo que nessa tese utilizou-se apenas as matrizes unitárias $\mathbb{T}$ referentes a DCT-II e a DST-I [57, 58], logo:

$$\underline{\mathbf{x}}_{\mathrm{T}}(n) = \mathrm{T}[\underline{\mathbf{x}}(n)] \tag{5.17}$$

$$\underline{\mathbf{x}}_{\mathrm{T}}(n) = [x_{\mathrm{T},0}(n), x_{\mathrm{T},1}(n), \cdots, x_{\mathrm{T},(N-1)}(n)] \tag{5.18}$$

$\underline{\mathbf{w}}_{\mathrm{T}}(n)$ é o vetor de pesos, cujas componentes são dadas por:

$$\underline{\mathbf{w}}_{\mathrm{T}}(n) = [w_{\mathrm{T},0}(n), w_{\mathrm{T},1}(n), \cdots, w_{\mathrm{T},(N-1)}(n)] \tag{5.19}$$

$$y(n) = \underline{\mathbf{w}}_{\mathrm{T}}^{T}(n)\underline{\mathbf{x}}_{\mathrm{T}}(n) \tag{5.20}$$

$$e(n) = d(n) - y(n) \tag{5.21}$$

$\underline{\boldsymbol{\mu}}(n)$ é a matriz diagonal de pesos definida por:

$$\underline{\boldsymbol{\mu}}(n) = \mu \; diag^{-1}\left(\hat{\sigma}^2_{x_{\mathrm{T},0}}(n), \hat{\sigma}^2_{x_{\mathrm{T},1}}(n), \cdots, \hat{\sigma}^2_{x_{\mathrm{T},(N-1)}}(n) \right) \tag{5.22}$$

e

$\hat{\sigma}^2_{x_{\mathrm{T},i}}(n)$ é a estimativa de $\sigma^2_{x_{\mathrm{T},i}}(n) = \mathbf{E}[x^2_{\mathrm{T},i}(n)]$. O cálculo dessa estimativa é feito através da seguinte equação [56]:

$$\hat{\sigma}^2_{x_{\mathrm{T},i}}(n) = \beta\hat{\sigma}^2_{x_{\mathrm{T},i}}(n-1) + (1-\beta)x^2_{\mathrm{T},i}(n) \qquad i = 0,1,\cdots,N-1 \tag{5.23}$$

onde $0 < \beta < 1$ e β é uma constante escolhida com valor próximo de 1.

Pela Eq.(5.22), tem-se uma divisão dada por $\mu / \hat{\sigma}^2_{x_{\mathrm{T},i}}(n)$, então, para prevenir uma amplificação excessiva (instabilidade do LMS) em momentos que a $\hat{\sigma}^2_{x_{\mathrm{T},i}}(n)$ tende à zero, adiciona-se um pequeno valor positivo δ ao denominador.

5.3 Incorporação do Processo de Decomposição na Estrutura do Algoritmo LMS

O objetivo desta seção é apresentar uma variação do algoritmo LMS baseado nas equações de decomposição do sinal de entrada. De forma análoga às seções anteriores, apresenta-se a equação iterativa do gradiente descendente e se deduz o vetor gradiente, a partir de uma versão da Eq.(3.40) variável com o tempo. Em seguida, partindo-se da equação iterativa do gradiente descendente, apresenta-se as variações do algoritmo LMS tradicional, a saber: o algoritmo D-LMS no domínio do tempo e o algoritmo DSTr-LMS no domínio da freqüência.

5.3.1 Geração do Algoritmo Gradiente Descendente a Partir da Função Custo J Relativa ao Sistema de Equações D

Inicia-se essa seção denominando-se $\underline{\boldsymbol{\theta}}$(n) de vetor peso e reescrevendo-se a Eq.(5.1) em função desse vetor. Logo, tem-se que:

$$\underline{\theta}(n+1) = \underline{\theta}(n) + \frac{1}{2}\mu[-\nabla J(n)] \tag{5.24}$$

onde J(n) é a notação simplificada de J($\underline{\boldsymbol{\theta}}$(n)).

O vetor $\underline{\mathbf{x}}$ expresso pela Eq.(3.67.b) representa as incógnitas do sistema visto no Capítulo 3, podendo ser formado por elementos dos vetores $\underline{\mathbf{w}}$ e $\underline{\mathbf{ww}}$ expressos pela Eq.(3.13.c). Esse vetor é invariável com o tempo e corresponde a posição onde ocorre um mínimo no índice de desempenho J fornecido pela Eq.(3.40). Substituindo-se $\underline{\mathbf{w}}$ e $\underline{\mathbf{ww}}$ presentes na Eq.(3.40) pelos vetores pesos variáveis $\underline{\mathbf{w}}(n)$ e $\underline{\mathbf{ww}}(n)$ fornecidos respectivamente pelas Eqs.(5.25.a) e (5.25.b), cria-se um correspondente índice de desempenho J(n) fornecido pela Eq.(5.26), variável com o instante “n”.

$$\underline{\mathbf{w}}(n) = [w_0(n)\ w_1(n)\ \ w_2(n)\ \ \ldots\ \ w_{N\text{-}1}(n)]^T \tag{5.25.a}$$

$$\underline{\mathbf{ww}}(n) = [ww_0(n)\ ww_1(n)\ \ ww_2(n)\ \ \ldots\ \ ww_{N\text{-}1}(n)]^T \tag{5.25.b}$$

$$\begin{aligned} J(\underline{\theta}(n)) = J(\underline{\mathbf{w}}(n), \underline{\mathbf{ww}}(n)) = \sigma_d^2 - 2\underline{\mathbf{w}}^T(n)\underline{\mathbf{p}} + \underline{\mathbf{w}}^T(n)\underline{\mathbf{R}}_{xr}\underline{\mathbf{w}}(n) - 2\underline{\mathbf{ww}}^T(n)\underline{\mathbf{pp}} + \\ + \underline{\mathbf{ww}}^T(n)\underline{\mathbf{R}}_r\underline{\mathbf{ww}}(n) + 2\underline{\mathbf{w}}^T(n)\underline{\mathbf{R}}_{xr\,r}\underline{\mathbf{ww}}(n) \end{aligned} \tag{5.26}$$

onde $\underline{\boldsymbol{\theta}}$(n) é um vetor que agrupa os elementos de $\underline{\mathbf{w}}(n)$ e $\underline{\mathbf{ww}}(n)$ expresso a seguir por:

$$\underline{\theta}(n) = [w_0(n)\ w_1(n)\ \ \ldots\ \ w_{N\text{-}1}(n)\ ww_0(n)\ ww_1(n)\ \ \ldots\ \ ww_{N\text{-}1}(n)]^T \tag{5.27.a}$$

$$\underline{\theta}(n) = [\theta_0(n)\quad \theta_1(n)\quad \cdots\quad \theta_{2N\text{-}1}(n)]^T \tag{5.27.b}$$

Agrupando-se em um só vetor as componentes pertencentes às Eqs.(3.18.b) e (3.16.b), tem-se que:

$$\underline{\mathbf{u}}(n) = \begin{bmatrix} \underline{\mathbf{xr}}(n) \\ \cdots\cdots \\ \underline{\mathbf{r}}(n) \end{bmatrix} \tag{5.28}$$

$$\underline{\mathbf{u}}(n) = [xr_0(n) \;\; xr_1(n) \;\; \mathbf{....} \;\; xr_{(N-1)}(n) \;\; r_0(n) \;\; r_1(n) \;\; \mathbf{....} \;\; r_{(N-1)}(n)]^T \tag{5.29.a}$$

$$\underline{\mathbf{u}}(n) = [u_0(n) \;\; u_1(n) \;\; \cdots \;\; u_{2N-1}(n)]^T \tag{5.29.b}$$

logo, pode-se reescrever a Eq.(3.21.c) utilizando-se as Eqs.(5.27.a) e (5.29.a) como:

$$e(n) = d(n) - \underline{\theta}^T(n)\underline{\mathbf{u}}(n) \tag{5.30}$$

Baseando-se na Eq.(5.26) e de forma similar às Eq.(5.3.a), (5.3.b) e (5.3.c), tem-se que o valor do vetor gradiente no instante "n" será dado por:

$$\nabla J(n) = \frac{\partial J(n)}{\partial \underline{\theta}(n)} \tag{5.31.a}$$

$$\frac{\partial \mathrm{J(n)}}{\partial \underline{\theta}\mathrm{(n)}} = \left[\frac{\partial \mathrm{J(n)}}{\partial \mathrm{w_0(n)}} \;\; \frac{\partial \mathrm{J(n)}}{\partial \mathrm{w_1(n)}} \;\; \cdots \;\; \frac{\partial \mathrm{J(n)}}{\partial \mathrm{w_{N-1}(n)}} \;\; \frac{\partial \mathrm{J(n)}}{\partial \mathrm{ww_0(n)}} \;\; \frac{\partial \mathrm{J(n)}}{\partial \mathrm{ww_1(n)}} \;\; \cdots \;\; \frac{\partial \mathrm{J(n)}}{\partial \mathrm{ww_{N-1}(n)}} \right]^{\mathrm{T}} \tag{5.31.b}$$

$$\frac{\partial \mathrm{J(n)}}{\partial \underline{\theta}\mathrm{(n)}} = \begin{bmatrix} \dfrac{\partial \mathrm{J(n)}}{\partial \underline{\mathbf{w}}\mathrm{(n)}} \\ \cdots\cdots\cdots \\ \dfrac{\partial \mathrm{J(n)}}{\partial \underline{\mathbf{ww}}\mathrm{(n)}} \end{bmatrix} \tag{5.31.c}$$

$$\frac{\partial J(n)}{\partial \underline{\theta}(n)} = \begin{bmatrix} -2\underline{\mathbf{p}} + 2\underline{\mathbf{R}}_{xr}\,\underline{\mathbf{w}}(n) + 2\underline{\mathbf{R}}_{xr\ r}\,\underline{\mathbf{ww}}(n) \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ -2\underline{\mathbf{pp}} + 2\underline{\mathbf{R}}_{r}\,\underline{\mathbf{ww}}(n) + 2\underline{\mathbf{R}}_{r\ xr}\,\underline{\mathbf{w}}(n) \end{bmatrix} \tag{5.31.d}$$

Recordando-se as definições dadas pelas Eqs.(3.24), (3.26), (3.28.a), (3.31.a), (3.34.a) e (3.37), pode-se reescrever a Eq.(5.31.d) na seguinte forma:

$$\frac{\partial J(n)}{\partial \underline{\theta}(n)} = \mathbf{E}\begin{bmatrix} -2d(n)\underline{\mathbf{xr}}(n) + 2\underline{\mathbf{xr}}(n)\underline{\mathbf{xr}}^{T}(n)\underline{\mathbf{w}}(n) + 2\underline{\mathbf{xr}}(n)\underline{\mathbf{r}}^{T}(n)\underline{\mathbf{ww}}(n) \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ -2d(n)\underline{\mathbf{r}}(n) + 2\underline{\mathbf{r}}(n)\underline{\mathbf{r}}^{T}(n)\underline{\mathbf{ww}}(n) + 2\underline{\mathbf{r}}(n)\underline{\mathbf{xr}}^{T}(n)\underline{\mathbf{w}}(n) \end{bmatrix} \tag{5.32.a}$$

$$\frac{\partial J(n)}{\partial \underline{\theta}(n)} = \mathbf{E}\begin{Bmatrix} -2\underline{\mathbf{xr}}(n)[d(n) - \underline{\mathbf{xr}}^{T}(n)\underline{\mathbf{w}}(n) - \underline{\mathbf{r}}^{T}(n)\underline{\mathbf{ww}}(n)] \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ -2\underline{\mathbf{r}}(n)[d(n) - \underline{\mathbf{r}}^{T}(n)\underline{\mathbf{ww}}(n) - \underline{\mathbf{xr}}^{T}(n)\underline{\mathbf{w}}(n)] \end{Bmatrix} \tag{5.32.b}$$

$$\frac{\partial J(n)}{\partial \underline{\theta}(n)} = \mathbf{E}\begin{bmatrix} -2e(n)\underline{\mathbf{xr}}(n) \\ \cdots\cdots\cdots \\ -2e(n)\underline{\mathbf{r}}(n) \end{bmatrix} \tag{5.32.c}$$

$$\frac{\partial J(n)}{\partial \underline{\theta}(n)} = \mathbf{E}\left[-2e(n)\underline{\mathbf{u}}(n)\right] \tag{5.32.d}$$

Pode-se ainda, utilizando-se a Eq.(5.30) deduzir a Eq.(5.32.d) de forma análoga às Eqs.(5.4.a), (5.4.b), (5.4.c). Logo, o valor do vetor gradiente no instante “n” será dado por:

$$\nabla J(n) = \frac{\partial \mathbf{E}[e^{2}(n)]}{\partial \underline{\theta}(n)} \tag{5.33.a}$$

$$\nabla J(n) = \mathbf{E}\left\{2e(n)\frac{\partial}{\partial \underline{\theta}(n)}[d(n) - \underline{\theta}^{T}(n)\underline{\mathbf{u}}(n)]\right\} \tag{5.33.b}$$

$$\nabla J(n) = -\mathbf{E}[2e(n)\underline{\mathbf{u}}(n)] \tag{5.33.c}$$

e substituindo-se a Eq.(5.33.c) na Eq.(5.24), tem-se a seguinte expressão:

$$\underline{\theta}(\mathrm{n}+1)=\underline{\theta}(\mathrm{n})+\frac{1}{2}\mu * \mathbf{E}[2\mathrm{e}(\mathrm{n})\underline{\mathbf{u}}(\mathrm{n})] \tag{5.34}$$

que representa a equação do gradiente descendente, expressa em função do erro $e(n)$ e das diversas entradas decomposta que constituem o vetor $\underline{\mathbf{u}}(n)$.

5.3.2 Algoritmos LMS no Domínio do Tempo e da Freqüência com a Inclusão da Decomposição do Sinal de Entrada

Apresenta-se na Fig.(5.3), uma proposta de algoritmo adaptativo que se baseia na decomposição do sinal de entrada e na utilização do algoritmo LMS denominado de algoritmo D-LMS . A operação desse algoritmo é muito parecida com a do LMS convencional, a diferença é que as amostras do sinal de entrada presentes nas N derivações do filtro transversal sofrem uma decomposição gerando 2*N amostras, conforme explicado no Capítulo 3. Cada amostra oriunda dessa decomposição é multiplicada por um peso cujos valores são ajustados utilizando-se o algoritmo LMS normalizado. De fato, o vetor $\underline{\mathbf{x}}$(n) a ser decomposto é formado pelas N amostras presentes na linhas de atraso e devido à sua decomposição são gerados os vetores $\underline{\mathbf{xr}}(n)$ e $\underline{\mathbf{r}}(n)$, cujas equações já vistas no Capítulo 3 são reescritas a seguir para maior clareza:

$$\underline{\mathbf{x}}(n)=[x_0(n)\;\; x_1(n)\;\; x_2(n)\;\; \mathbf{....}\;\; x_{N-1}(n)]^T \tag{5.35.a}$$

$$x_i(n)=x(n-i) \quad para \quad \begin{matrix} 0\le n \\ e \\ 0\le i\le N-1 \end{matrix} \tag{5.35.b}$$

$$r_i(n)=\frac{x_0(n)-x_{(N-1)}(n)}{N-1}(N-1-i)+x_{(N-1)}(n) \quad \text{para} \quad \begin{matrix} 0\le \mathrm{n} \\ \mathrm{e} \\ 0\le i\le N-1 \end{matrix} \tag{5.35.c}$$

que representa a amostra da seqüência r_i no instante “n”. Logo,

$$\underline{\mathbf{r}}(n) = [r_0(n)\ \ r_1(n)\ \ r_2(n)\ \ ....\ \ r_{(N-1)}(n)]^T \tag{5.35.d}$$

$$xr_i(n) = x_i(n) - r_i(n) \text{ para } \begin{matrix} 0 \le \mathrm{n} \\ \mathrm{e} \\ 0 \le \mathrm{i} \le N-1 \end{matrix} \tag{5.36.a}$$

que representa uma amostra da seqüência xr_i no instante "n". Logo,

$$\underline{\mathbf{x\,r}}(n) = [xr_0(n)\ \ xr_1(n)\ \ xr_2(n)\ \ ....\ \ xr_{(N-1)}(n)]^T \tag{5.36.b}$$

e portanto, o erro de estimação é dado pela Eq.(5.30). Utilizando-se uma estimativa instantânea para a Eq.(5.24) e adotando-se o valor de μ variável com a potência do sinal relativa a cada componente do vetor $\underline{\mathbf{u}}(n)$, então tem-se a seguinte equação de atualização do vetor peso $\underline{\boldsymbol{\theta}}(n)$:

$$\theta_i(n+1) = \theta_i(n) + \mu_i(n)e(n)u_i(n), \qquad i = 0,1,\cdots,2N-1 \tag{5.37}$$

onde , $\mu_i(n)$ é um passo de degrau variável dado por:

$$\mu_i(n) = \frac{\mu}{\hat{\sigma}^2_{u_i}(n)} \qquad i = 0,1,\cdots,2N-1 \tag{5.38}$$

$$\hat{\sigma}^2_{u_i}(n) = \beta\hat{\sigma}^2_{u_i}(n-1) + (1-\beta)u_i^2(n) \qquad i = 0,1,\cdots,2N-1 \tag{5.39}$$

onde $0 < \beta < 1$ e β é uma constante escolhida com valor próximo de 1.

Pela Eq.(5.38) tem-se uma divisão dada por $\mu \big/ \hat{\sigma}^2_{u_i}(n)$, então, para prevenir uma amplificação excessiva (instabilidade do LMS) em momentos que a $\hat{\sigma}^2_{u_i}(n)$ tende à zero, adiciona-se um pequeno valor positivo δ ao denominador.

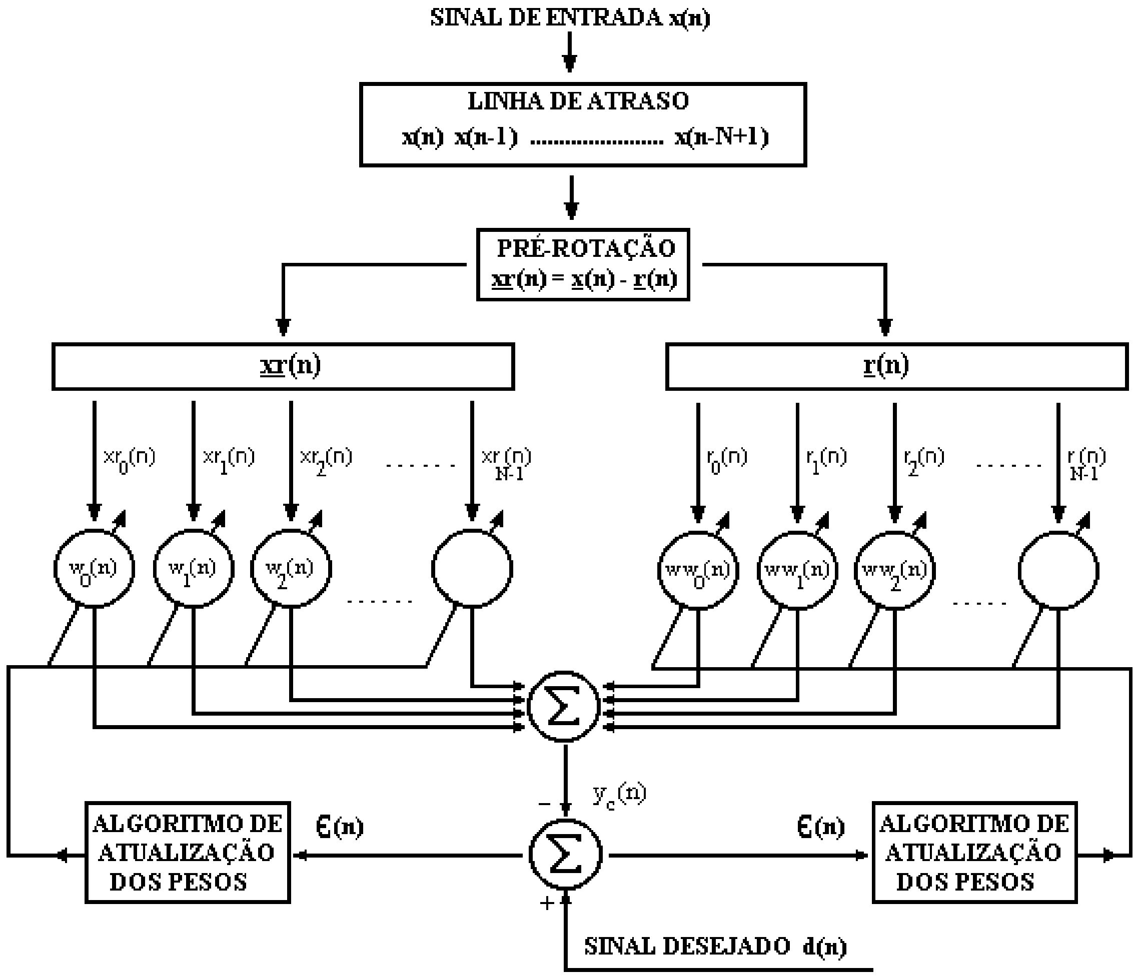


Fig.5.3. Diagrama em blocos do filtro adaptativo D- LMS.

O diagrama em blocos do algoritmo D-LMS apresentado na Fig.(5.3), baseou-se nas Eqs.(5.25.a), (5.25.b), (5.35.a), (5.35.d) e (5.36.b). No entanto, utilizando-se a Eq.(5.27.a) que agrupa os vetores peso **w**(n) e **ww**(n) em um só vetor e a Eq.(5.29.a) que agrupa os vetores **xr**(n) e **r**(n) no vetor **u**(n), pode-se simplificar o diagrama em blocos do algoritmo D-LMS, conforme apresenta a Fig.(5.4).

A Fig.(5.5) apresenta o algoritmo DSTr-LMS, que é uma simples variação do algoritmo D-LMS. Ele é obtido aplicando-se a Transformada Discreta Seno do tipo I (DST-I) dada pela Eq.(5.40) [57, 58] (vide Apêndice A), sobre as amostras do vetor **xr**(n) para gerar o vetor transformado $\underline{\mathbf{xr}}_{DST}(n)$. Da mesma forma, aplicando-se essa transformada sobre $\underline{\mathbf{r}}(n)$, gera-se o vetor transformado $\underline{\mathbf{r}}_{DST}(n)$.

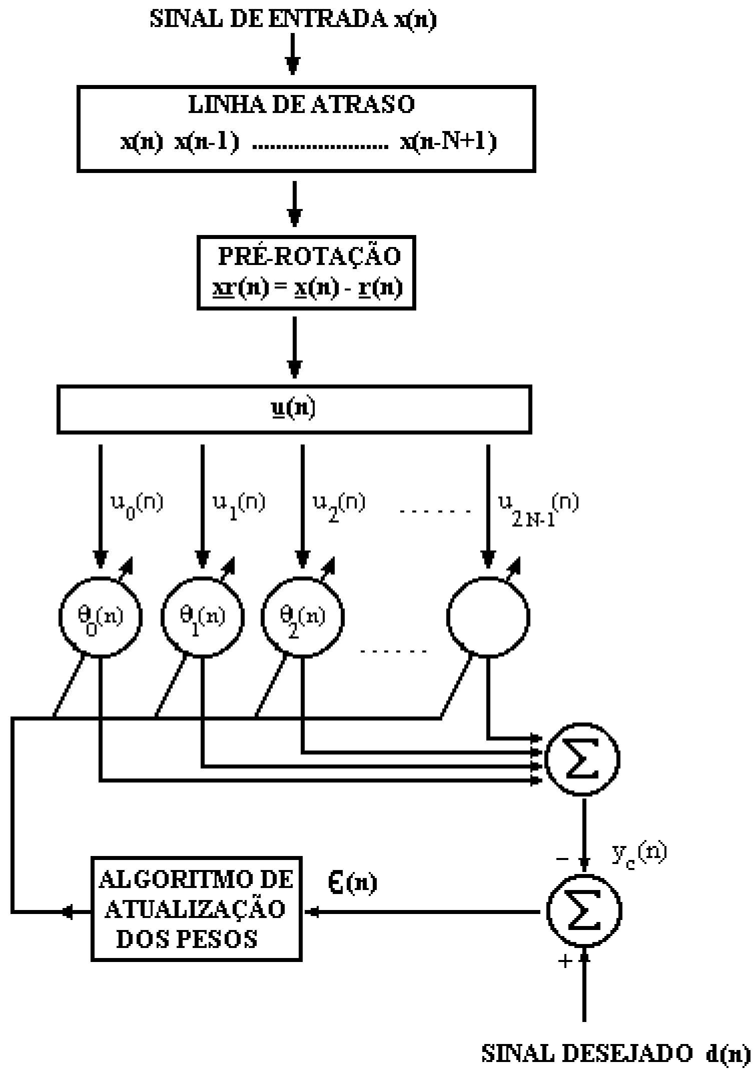


Fig.5.4. Simplificação do diagrama em blocos do filtro adaptativo D- LMS.

$$S_N^I(m,n) = \sqrt{\frac{2}{N+1}}\,\mathbf{sin}\left(\frac{(m+1)(n+1)}{N+1}\mathrm{pi}\right), \qquad m,\ n = 0,1,\cdots,N-1 \tag{5.40}$$

onde $S_N^I(m,n)$ representa cada um dos elementos da matriz de transformação da DST-I de dimensão N x N, denotada por $\underline{\mathbf{S}}_N^I$. Portanto, os elementos $xr_{DST,m}(n)$ do vetor transformado $\underline{\mathbf{xr}}_{DST}(n)$ serão calculados pela Eq.(5.41.a).

$$xr_{DST,m}(n) = \sum_{t=0}^{N-1} S_N^I(m,t) xr_t(n), \qquad m = 0,1,\cdots,N-1 \tag{5.41.a}$$

$$\underline{\mathbf{xr}}_{DST}(n) = \left[xr_{DST,0}(n) \quad xr_{DST,1}(n) \quad \cdots \quad xr_{DST,(N-1)}(n) \right]^T \tag{5.41.b}$$

Tem-se ainda que os coeficientes do vetor transformado $\underline{\mathbf{r}}_{DST}(n)$ são fornecidos por:

$$r_{DST,m}(n) = \sum_{t=0}^{N-1} S_N^I(m,t) r_t(n), \qquad m = 0,1,\cdots,N-1 \tag{5.42.a}$$

$$\underline{\mathbf{r}}_{DST}(n) = \left[r_{DST,0}(n) \quad r_{DST,1}(n) \quad \cdots \quad r_{DST,(N-1)}(n) \right]^T \tag{5.42.b}$$

Observando-se a Fig.(5.5), define-se $\underline{\mathbf{b}}(n)$ e $\underline{\mathbf{bb}}(n)$ como vetores peso dados por:

$$\underline{\mathbf{b}}(n) = \left[b_0(n) \quad b_1(n) \quad \cdots \quad b_{N-1}(n) \right]^T \tag{5.43}$$

$$\underline{\mathbf{bb}}(n) = \left[bb_0(n) \quad bb_1(n) \quad \cdots \quad bb_{N-1}(n) \right]^T \tag{5.44}$$

que podem ser agrupados em um só vetor $\underline{\phi}(n)$, onde,

$$\underline{\phi}(n) = \begin{bmatrix} \underline{\mathbf{b}}(n) \\ \cdots\cdots \\ \underline{\mathbf{bb}}(n) \end{bmatrix} \tag{5.45.a}$$

$$\underline{\phi}(n) = [b_0(n) \; b_1(n) \;\; \ldots \;\; b_{N\text{-}1}(n) \; bb_0(n) \; bb_1(n) \;\; \ldots \;\; bb_{N\text{-}1}(n)]^T \tag{5.45.b}$$

$$\underline{\phi}(n) = [\phi_0(n) \quad \phi_1(n) \quad \cdots \quad \phi_{2N-1}(n)]^T \tag{5.45.c}$$

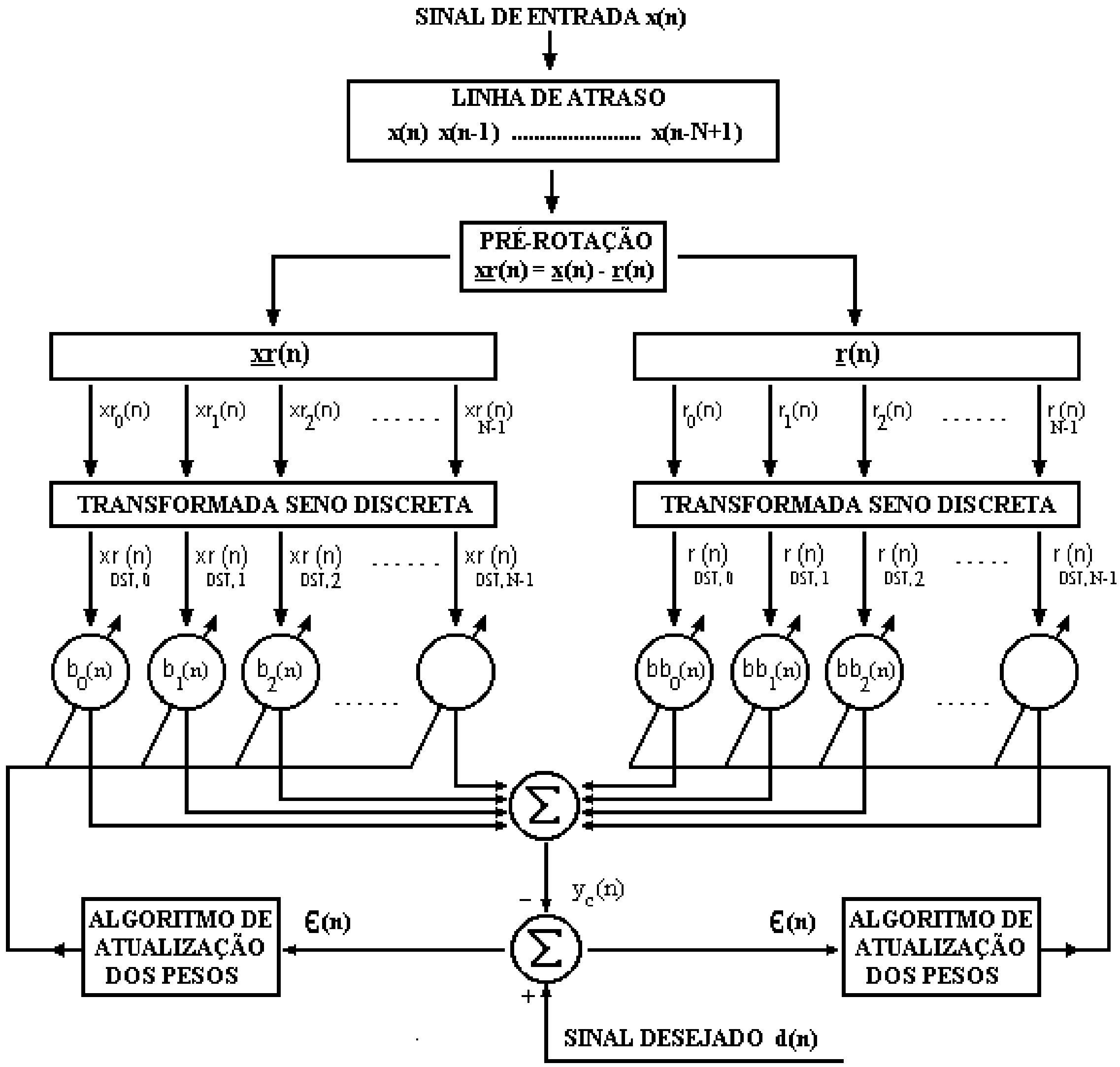


Fig.5.5. Diagrama em blocos do filtro adaptativo DSTr- LMS.

Agrupando-se também as componentes pertencentes às Eqs.(5.41.b) e (5.42.b), tem-se que:

$$\underline{\mathbf{u}}_{\mathrm{DST}}(n) = \begin{bmatrix} \underline{\mathbf{xr}}_{\mathrm{DST}}(n) \\ \cdots\cdots \\ \underline{\mathbf{r}}_{\mathrm{DST}}(n) \end{bmatrix} \tag{5.46.a}$$

$$\underline{\mathbf{u}}_{\mathrm{DST}}(n) = [xr_{DST,0}(n) \;\; xr_{DST,1}(n) \;\; \mathbf{....} \;\; xr_{DST,(N-1)}(n) \;\; r_{DST,0}(n) \;\; r_{DST,1}(n) \;\; \mathbf{....} \;\; r_{DST,(N-1)}(n)]^T \tag{5.46.b}$$

$$\underline{\mathbf{u}}_{\mathrm{DST}}(n) = [u_{DST,0}(n) \;\; u_{DST,1}(n) \;\; \cdots \;\; u_{DST,2N-1}(n)]^T \tag{5.46.c}$$

Logo, pode-se expressar o erro de estimação $e(n)$ como:

$$e(n) = d(n) - \underline{\phi}^T(n)\underline{\mathbf{u}}_{\mathrm{DST}}(n) \tag{5.47}$$

De forma análoga a Eq.(5.37), tem-se a equação de atualização do vetor peso $\underline{\phi}(n)$ expressa por:

$$\phi_i(n+1) = \phi_i(n) + \mu_i(n)e(n)u_{\mathrm{DST},\,i}(n), \qquad i = 0, 1, \cdots, 2N-1 \tag{5.48}$$

onde , $\mu_i(n)$ é um passo de degrau variável dado por:

$$\mu_i(n) = \frac{\mu}{\hat{\sigma}^2_{u_{\mathrm{DST},\,i}}(n)} \qquad i = 0, 1, \cdots, 2N-1 \tag{5.49}$$

$$\hat{\sigma}^2_{u_{\mathrm{DST},\,i}}(n) = \beta\hat{\sigma}^2_{u_{\mathrm{DST},\,i}}(n-1) + (1-\beta)u^2_{\mathrm{DST},\,i}(n) \qquad i = 0, 1, \cdots, 2N-1 \tag{5.50}$$

onde $0 < \beta < 1$ e β é uma constante escolhida com valor próximo de 1.

Pela Eq.(5.49), tem-se uma divisão dada por $\mu \big/ \hat{\sigma}^2_{u_{DST,\,i}}(n)$, então, para prevenir uma amplificação excessiva (instabilidade do LMS) em momentos que a $\hat{\sigma}^2_{u_{DST,\,i}}(n)$ tende a zero, adiciona-se um pequeno valor positivo δ ao denominador.

CAPÍTULO 6

Simulações

6.1 Introdução

Propõe-se neste capítulo, apresentar simulações que comparam os resultados provenientes do sistema de equações D com os advindos do algoritmo D-LMS.

Além disso, geram-se uma série de "Curvas de Aprendizagem" e os parâmetros extraídos das mesmas são utilizados na comparação do desempenho dos algoritmos D-LMS e DSTr-LMS com outros algoritmos da mesma classe, tais como NLMS, DST-LMS e DCT-LMS, em situações que envolvam a predição linear de processos AR(2) e de sinais eletromiográficos.

6.2 Grupo de Simulações que Compara os Resultados Obtidos pelo Sistema de Equações D com os Advindos do Algoritmo D-LMS

Cada simulação desse grupo consiste na comparação dos valores dos coeficientes obtidos por um dado sistema de equações D, visto no Capítulo 4, e a solução encontrada através do algoritmo D-LMS. Em outras palavras, tem-se a predição linear de um processo AR(1) e espera-se que os valores dos pesos obtidos para um determinado sistema de equações D, sejam muito próximos dos valores dos pesos obtidos pelo algoritmo D-LMS, conforme será visto a seguir.

6.2.1 <u>Simulação-1</u>: Comparação entre os Valores dos Pesos ww_0, ww_2 e w_1 Obtidos Via Eq.(4.27.a) e os Obtidos Via D-LMS.

O primeiro sistema avaliado no Capítulo 4 foi aquele descrito pela Eq.(4.27.a) que desconsidera a utilização da entrada decomposta $r_1(n)$ e por conseguinte são calculados apenas os pesos ww_0, ww_2 e w_1. Para efeito de comparação, adotou-se valores médios para esses coeficientes, onde os mesmos estão representados na Fig.(4.7.a). Com base nessa figura e considerando-se NA=4010 (i=401), tem-se que, $\overline{ww_0} = 0.992$, $\overline{ww_2} = -0.002$, $\overline{w_1} = 0.004$.

Seja o algoritmo D-LMS utilizado em um esquema de predição linear dado pela Fig.(6.1). Considera-se ainda, que o sinal aleatório na entrada desse preditor seja fornecido por uma função amostra pertencente a um processo AR(1), dado pela Eq.(4.18.a) e reescrita a seguir:

$$x(n) = 0.99x(n-1) + \upsilon(n) \tag{6.1}$$

onde $a_1 = -0.99$

A Fig.(6.2) retrata o comportamento dos valores dos pesos ww_0, ww_2 e w_1 obtidos através do algoritmo D-LMS empregado no esquema de predição linear apresentado pela Fig.(6.1), quando o mesmo está sujeito sob as seguintes condições:

-Predição de uma amostra à frente, isto é, $d(n)=x(n\text{-}1)$ ou $\Delta=1$.

-Número de amostras igual a 160000.

-$\mu=0.0002$; a escolha desse valor se deu através da observância da saturação dos coeficientes "ww_0" e "ww_2" a partir de cem mil amostras processadas conforme apresenta a Fig.(6.2). Diminuindo-se o valor de μ, o processo tende a ficar mais lento, e portanto, um maior número de amostras serão necessários para que os coeficientes "ww_0" e "ww_2" atinjam os seu valores de regime. De forma geral essa é uma regra que baliza a escolha desse parâmetro.

-$\beta=0.9$; a escolha desse valor obedece a faixa compreendida entre 0 e 1, de preferência próximo de 1, conforme expresso por [19, 39].

-$\delta=0$; esse valor foi adotado por não terem sido observadas diferenças significativas quando utilizado um valor pequeno, porém próximo de zero, da ordem de 10^{-4} segundo [39]. Essa é uma regra a ser observada em simulações futuras.

-Iniciam-se os vetores pesos relativos a qualquer algoritmo adaptativo presente nessa tese, através do vetor nulo.

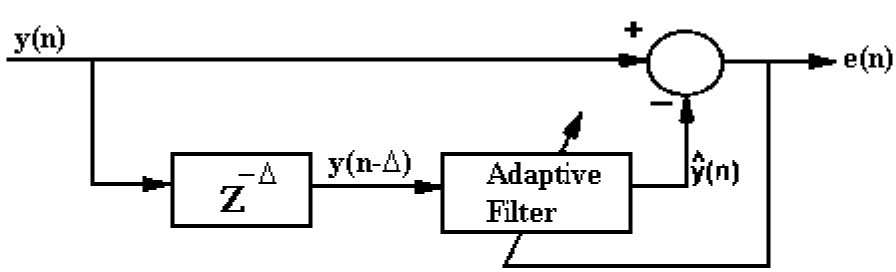


Fig.6.1. Diagrama em blocos de um preditor linear adaptativo de Δ amostras a frente.

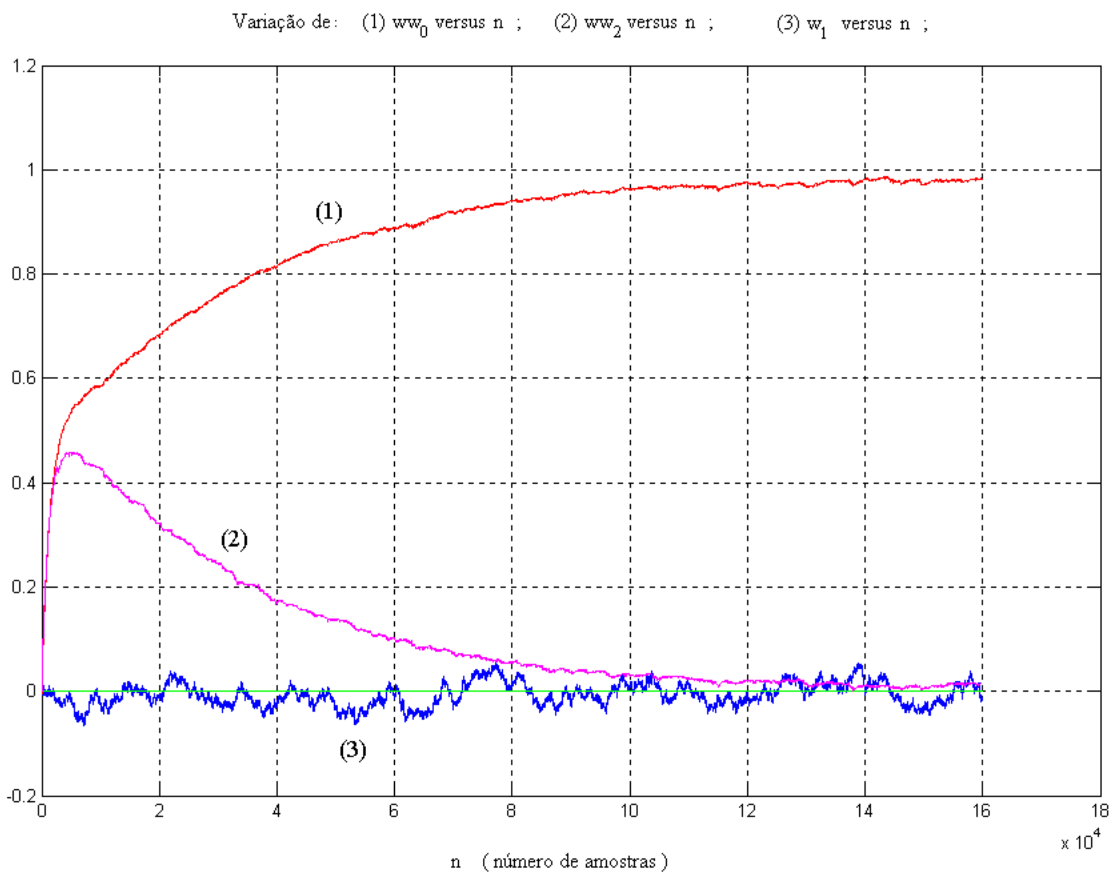


Fig.6.2. Variação dos pesos ww_0, ww_2 e w_1 em função do número de amostras n do processo de entrada $x(n)$, considerando-se que μ=0.0002 e β=0.9.

6.2.2 Simulação-2: Comparação entre os Valores dos Pesos ww_1, ww_2 e w_1 Obtidos Via Eq.(4.29.a) e os Obtidos Via D-LMS.

Essa simulação consiste na comparação entre os valores dos coeficientes obtidos pelo sistema de equações D descrito pela Eq.(4.29.a) e a solução encontrada através do algoritmo D-LMS. Nesse segundo sistema, avaliado no Capítulo 4, desconsidera-se a utilização da entrada decomposta $r_0(n)$ e por conseguinte são calculados apenas os pesos ww_1, ww_2 e w_1. Para efeito de comparação, adotou-se valores médios para esses coeficientes, onde os mesmos estão representados na Fig.(4.8.a). Com base nessa figura, e considerando-se NA=4010 (i=401), tem-se que, $\overline{ww_1} = 1.984$, $\overline{ww_2} = -0.994$, $\overline{w_1} = 0.004$. A Fig.(6.3) retrata o comportamento dos valores dos pesos ww_1, ww_2 e w_1 obtidos através do algoritmo D-LMS empregado no esquema de predição linear apresentado pela Fig.(6.1), quando o mesmo está sujeito sob condições idênticas presentes no experimento 1, exceto que μ=0.0008.

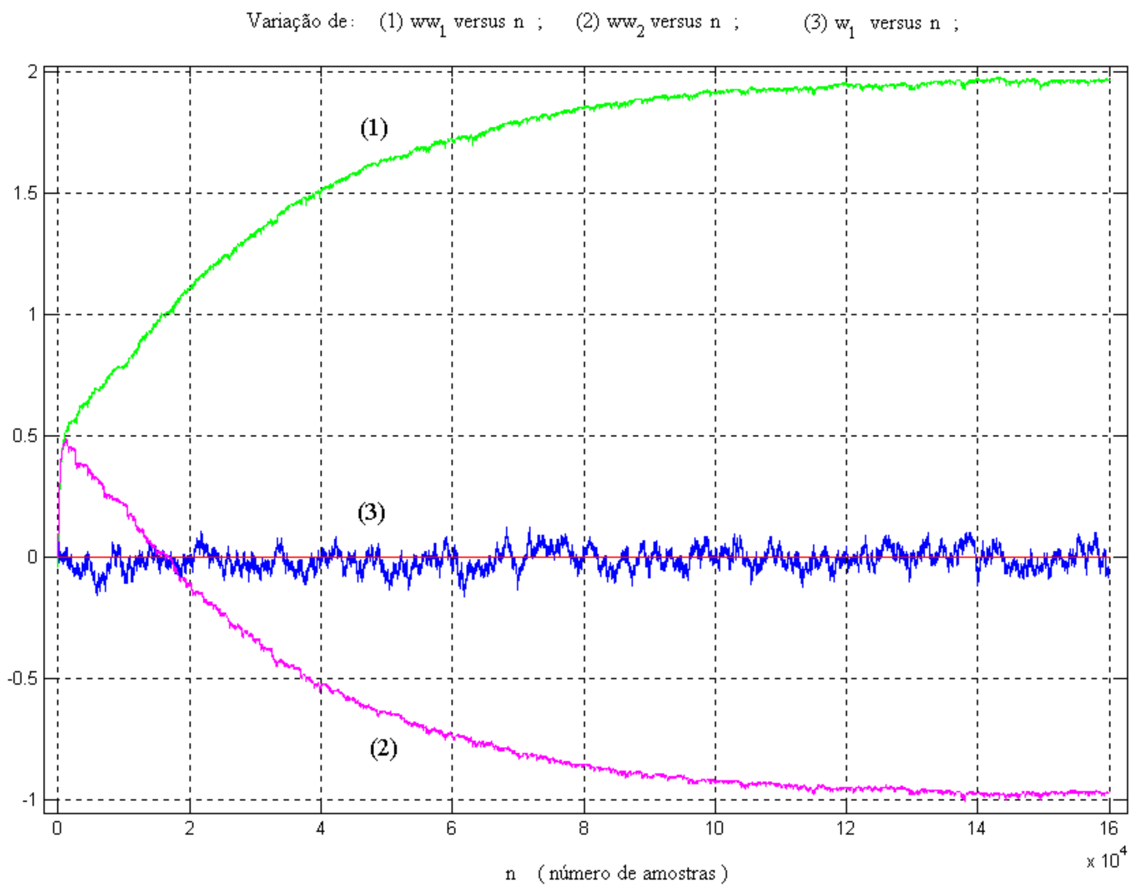


Fig.6.3. Variação dos pesos ww_1, ww_2 e w_1 em função do número de amostras n do processo de entrada $x(n)$, considerando-se que μ=0.0008 e β=0.9.

6.2.3 <u>Simulação-3</u>: Comparação entre os Valores dos Pesos ww_0, ww_1 e w_1 Obtidos Via Eq.(4.30.a) e os Obtidos Via D-LMS.

Essa simulação consiste na comparação entre os valores dos coeficientes obtidos pelo sistema de equações D descrito pela Eq.(4.30.a) e a solução encontrada através do algoritmo D-LMS. Nesse terceiro sistema, avaliado no Capítulo 4, desconsidera-se a utilização da entrada decomposta $r_2(n)$ e por conseguinte, são calculados apenas os pesos ww_0, ww_1 e w_1. Para efeito de comparação, adotou-se valores médios para esses coeficientes, onde os mesmos estão representados na Fig.(4.9.a). Com base nessa figura, e considerando-se *NA*=4010 (*i*=401), tem-se que, $\overline{ww_0} = 0.994$, $\overline{ww_1} = -0.004$, $\overline{w_1} = 0.004$.

A Fig.(6.4) retrata o comportamento dos valores dos pesos ww_0, ww_1 e w_1 obtidos através do algoritmo D-LMS empregado no esquema de predição linear apresentado pela Fig.(6.1), quando o mesmo está sujeito sob condições idênticas presentes no experimento 1, exceto que μ=0.0008.

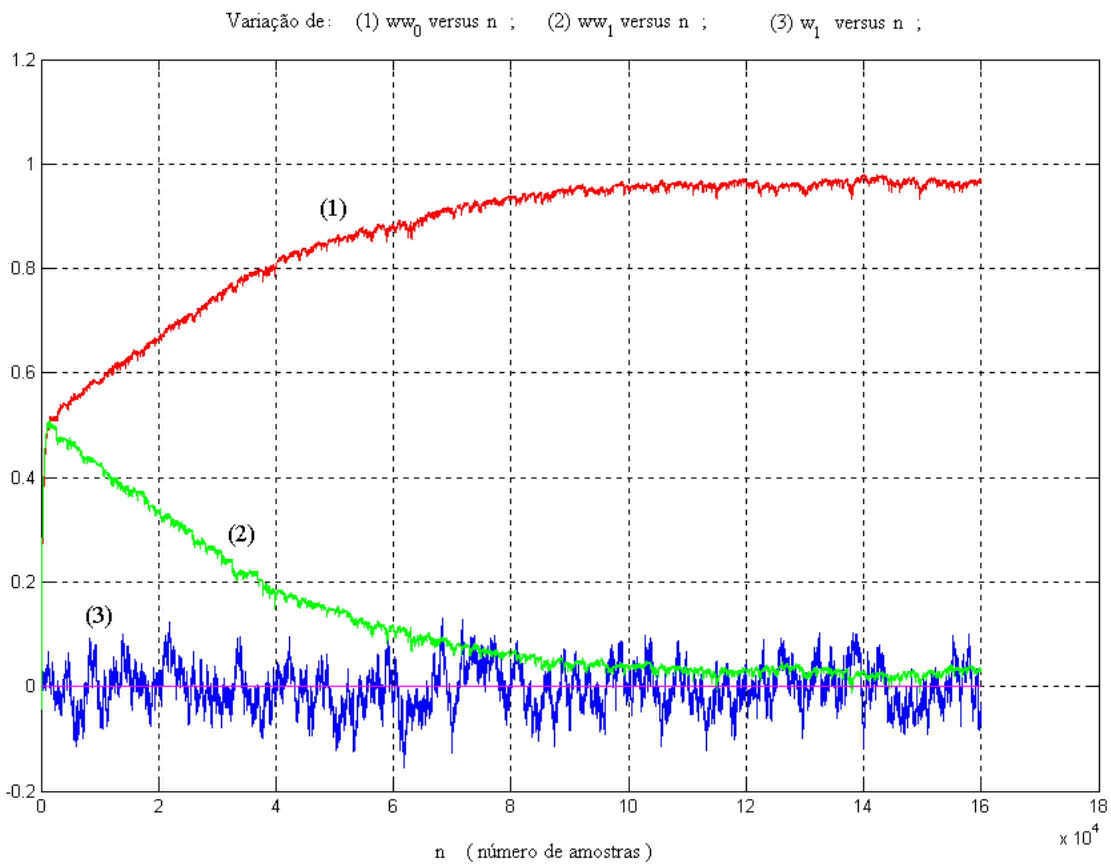


Fig.6.4. Variação dos pesos ww_0, ww_1 e w_1 em função do número de amostras n do processo de entrada $x(n)$, considerando-se que μ=0.0008 e β=0.90.

6.2.4 <u>Simulação-4</u>: Comparação entre os Valores dos Pesos ww_0, ww_2 e w_1 Obtidos Via Eq.(4.27.a) e os Obtidos Via D-LMS.

Análoga a simulação 1, diferindo-se fundamentalmente na adoção do sinal aleatório na entrada do preditor dado pela Fig.(6.1), constituir-se de um processo AR(1) com $a_1 = -0.2$. Essa simulação consiste na comparação entre os valores dos coeficientes obtidos pelo sistema de equações D descrito pela Eq.(4.27.a) com a solução encontrada através do algoritmo D-LMS. Para efeito de comparação, adotou-se valores médios para os coeficientes ww_0, ww_2 e w_1, representados na Fig.(4.10.a). Com base nessa figura e considerando-se NA=4010 (i=401), tem-se que, $\overline{ww_0} = 0.203$, $\overline{ww_2} = 0.001$, $\overline{w_1} = 0.004$.

A Fig.(6.5) retrata o comportamento dos valores dos pesos ww_0, ww_2 e w_1 obtidos através do algoritmo D-LMS empregado no esquema de predição linear apresentado pela Fig.(6.1), quando o mesmo está sujeito sob condições idênticas presentes na simulação 1, exceto que μ=4×10^{-5} .

6.2.5 <u>Simulação-5</u>: Comparação entre os Valores dos Pesos ww_1, ww_2 e w_1 Obtidos Via Eq.(4.29.a) e os Obtidos Via D-LMS.

Essa simulação é análoga a simulação 2, diferindo apenas na adoção do sinal aleatório na entrada do preditor, ser um processo AR(1) com $a_1 = -0.2$. Consiste na comparação entre os valores dos coeficientes obtidos pelo sistema de equações D descrito pela Eq.(4.29.a) com a solução encontrada através do algoritmo D-LMS. Para efeito de comparação, adotou-se valores médios para os coeficientes ww_1, ww_2, w_1, representados na Fig.(4.11.a). Com base nessa figura, e considerando-se $NA = 4010$ ($i = 401$), tem-se que, $\overline{ww_1} = 0.406$, $\overline{ww_2} = -0.201$, $\overline{w_1} = 0.004$.

A Fig.(6.6) retrata o comportamento dos valores dos pesos ww_1, ww_2 e w_1 obtidos através do algoritmo D-LMS, empregado no esquema de predição linear apresentado pela Fig.(6.1), quando o mesmo está sujeito sob condições idênticas presentes na simulação 2, exceto que $\mu = 6 \times 10^{-5}$.

6.2.6 <u>Simulação-6</u>: Comparação entre os Valores dos Pesos ww_0, ww_1 e w_1 Obtidos Via Eq.(4.30.a) e os Obtidos Via D-LMS.

Essa simulação é análoga a simulação 3, diferindo apenas na adoção do sinal aleatório na entrada do preditor, ser um processo AR(1) com $a_1 = -0.2$. Consiste na comparação entre os valores dos coeficientes obtidos pelo sistema de equações D descrito pela Eq.(4.30.a) com a solução encontrada através do algoritmo D-LMS. Para efeito de comparação, adotou-se valores médios para os coeficientes ww_1, ww_2, w_1, representados na Fig.(4.12.a). Com base nessa figura e considerando $NA = 4010$ ($i = 401$), tem-se que, $\overline{ww_0} = 0.201$, $\overline{ww_1} = 0.002$, $\overline{w_1} = 0.004$.

A Fig.(6.7) retrata o comportamento dos valores dos pesos ww_0, ww_1 e w_1 obtidos através do algoritmo D-LMS empregado no esquema de predição linear, quando o mesmo está sujeito sob condições idênticas presentes na simulação 3, inclusive $\mu = 8 \times 10^{-5}$.

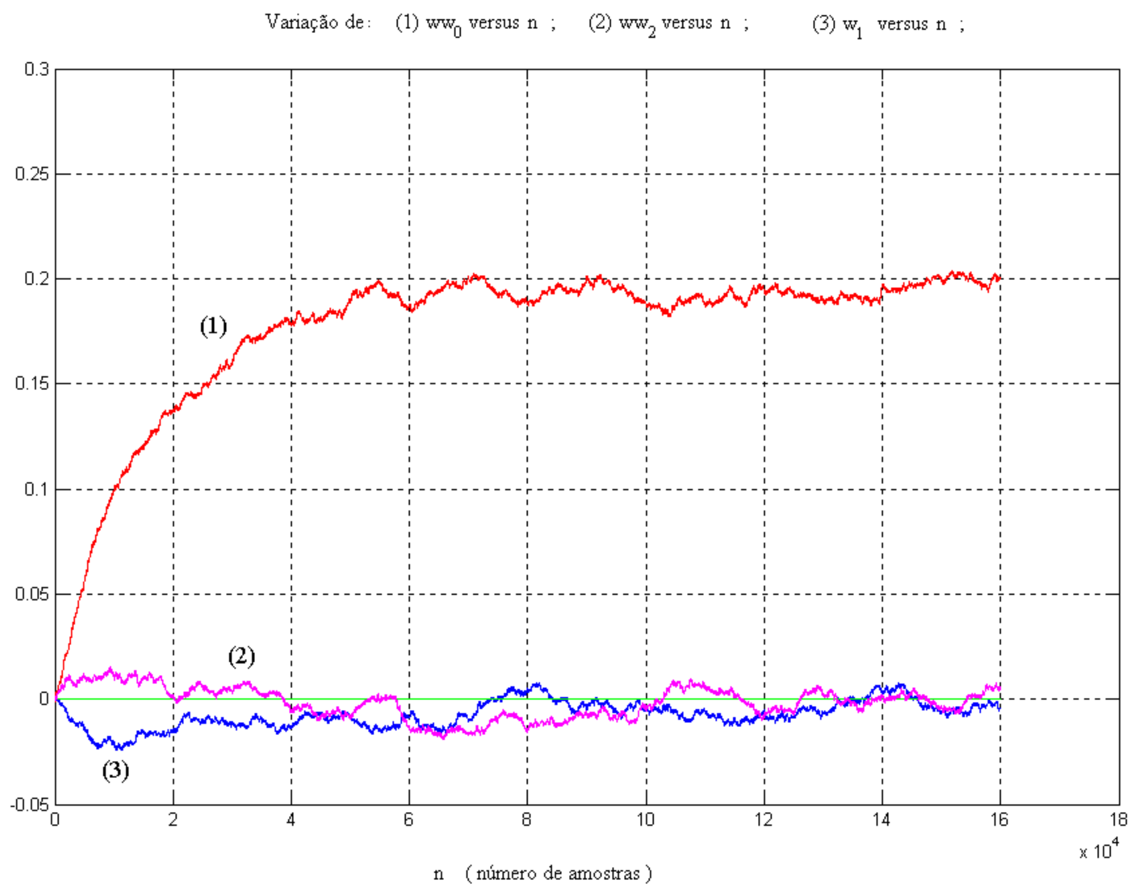


Fig.6.5. Variação dos pesos ww_0, ww_2 e w_1 em função do número de amostras n do processo de entrada $x(n)$, considerando-se que $\mu = 4 \times 10^{-5}$ e $\beta = 0.9$.

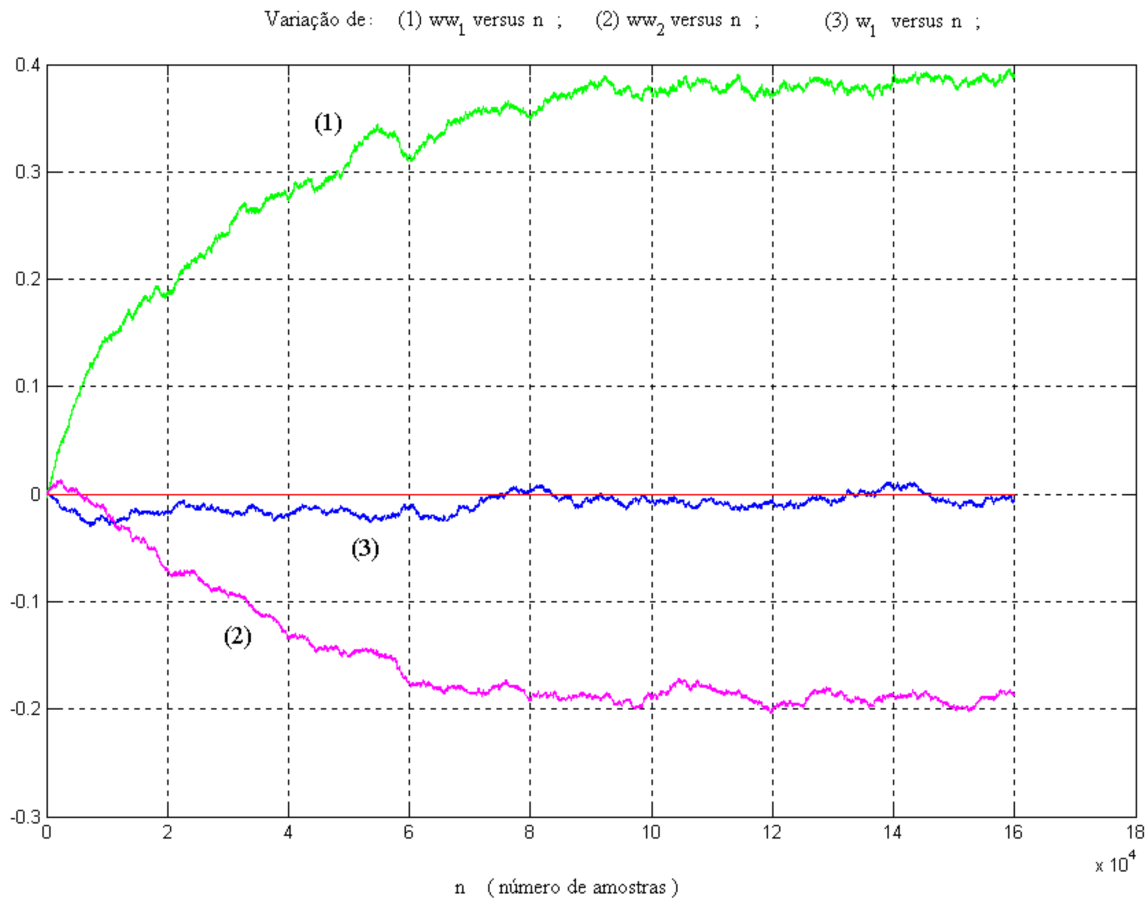


Fig.6.6. Variação dos pesos ww_1, ww_2 e w_1 em função do número de amostras n do processo de entrada $x(n)$, considerando-se que $\mu = 6 \times 10^{-5}$ e $\beta = 0.9$.

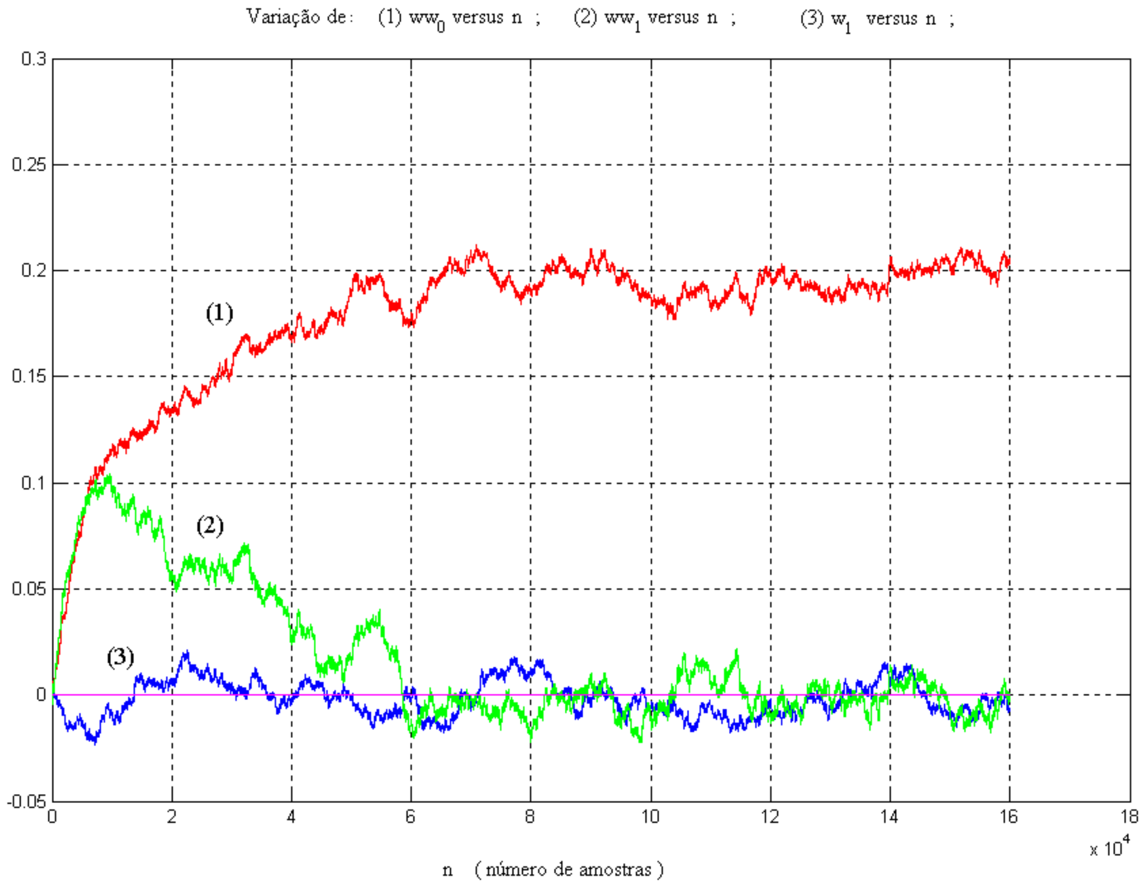


Fig.6.7. Variação dos pesos ww_0, ww_1 e w_1 em função do número de amostras n do processo de entrada $x(n)$, considerando-se que $\mu = 8 \times 10^{-5}$ e $\beta = 0.9$.

6.3 Grupo de Simulações que Analisa e Compara Entre Si o Desempenho dos Algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS

Cada simulação desse grupo objetiva analisar individualmente e comparar entre si o desempenho dos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS, empregados no esquema de predição linear dado pela Fig.(6.1), com $\Delta = 1$, quando submetidos a três processos autoregressivos de entrada com diferentes valores para o número de condição (*nc*).

Tanto a análise individual quanto a comparação entre os algoritmos, são feitas baseadas nas curvas de aprendizagem associadas a um dado processo e em parâmetros extraídos das mesmas, para cada um dos referidos algoritmos. Esses parâmetros que qualificam o desempenho dos algoritmos, relativos a determinados aspectos, são os seguintes:

- Velocidade ou taxa de convergência;
- Erro de regime permanente $\hat{J}(\infty)$;
- Variação dos parâmetros anteriores com o número de condição (nc) da matriz de autocorrelação do processo de entrada.

Sendo uma curva de aprendizagem um gráfico dado por $J(n) = \mathbf{E}[|e(n)|^2]$ *versus n*, então, pode-se aproximar essa curva [39] por $\hat{J}(n) = \hat{\mathbf{E}}[|e(n)|^2]$ *versus n* tomando-se a média sobre “S” eventos da curva $|e(n)|^2$ *versus n*, segundo mostra a equação a seguir:

$$\hat{J}(n) = \hat{\mathbf{E}}\left[|e(n)|^2\right] = \frac{1}{S}\sum_{s=1}^{S}|e_s(n)|^2 \qquad (6.2)$$

onde, $|e_s(n)|^2$ denota o erro quadrático no instante “n” referente ao s-ésimo evento.

Define-se ainda o erro quadrático médio em regime permanente $J(\infty)$ [19, 39], como sendo dado por:

$$J(\infty) = J_{Min} + J_{excesso}(\infty) \qquad (6.3)$$

onde, J_{Min} é o erro quadrático médio mínimo e $J_{excesso}(\infty)$ é o erro quadrático médio adicional ou em excesso. Pode-se estimar o erro quadrático médio em regime permanente a partir de uma média sobre os valores da curva $\hat{J}(n)=\hat{\mathbf{E}}[|e(n)|^2]$ *versus* n, situados na região de regime permanente conforme denota a Eq.(6.4):

$$\hat{J}(\infty) = \hat{J}(Intervalo = N1:N2) = \frac{1}{\Delta N}\sum_{n=N1}^{N2}\hat{J}(n) \tag{6.4}$$

$$\Delta N = N2 - N1 + 1 \quad amostras \; (\; na \; região \; de \; regime \; permanente \;).$$

Denomina-se $y(n)=s_i(n)$ com i=1, 2 e 3, aos diferentes sinais utilizados na entrada do preditor apresentado na Fig.(6.1), sendo os mesmos gerados por processos AR(2) denominados respectivamente por AR1, AR2 e AR3. Esses processos são descritos pelas seguintes equações:

AR1: $s_1(n) = 1.9114 * s_1(n-1) - 0.95 * s_1(n-2) + \nu(n)$, logo $nc = 100.04$

AR2: $s_2(n) = 1.73 * s_2(n-1) - 0.81 * s_2(n-2) + \nu(n)$, logo $nc = 44.25$

AR3: $s_3(n) = 1.4 * s_3(n-1) - 0.81 * s_3(n-2) + \nu(n)$, logo $nc = 7.83$

onde $\nu(n)$ é um ruído gaussiano branco com variância unitária. Com base na notação utilizada nos diagramas dos algoritmos adaptativos apresentados pelas Figs.(5.1), (5.2), (5.3) e (5.4), têm-se que o sinal desejado ou primário é fornecido por $d(n)=s_i(n)$ e o sinal de entrada ou de referência é dado por $x(n)=s_i(n-1)$.

Cada curva de aprendizagem exibida nessa simulação é obtida pela média de 100 eventos independentes, ou seja, S=100 na Eq.(6.2), referentes a um dado processo de entrada constituído de 8000 amostras. Uma vez estimada a curva de aprendizagem, pode-se também estimar o erro quadrático médio de regime permanente $\hat{J}(\infty)$ sobre um intervalo de amostras através da Eq.(6.4). Para o intervalo denominado I_1, considerou-se os seguintes valores para os parâmetros presentes na Eq.(6.4): $N1$=7800, $N2$=7999 e ΔN=200. Para o intervalo denominado I_2, considerou-se os seguintes valores: $N1$=7000, $N2$=7199 e ΔN=200. Desde que esses intervalos estejam situados na região de regime permanente, tem-se que os $\hat{J}(\infty)$ obtidos para cada intervalo, devem ser coincidentes ou próximos, isto é, a diferença entre eles deve ser inferior a 3%. Dessa forma, a título de referência, os valores adotados para o erro de regime permanente $\hat{J}(\infty)$ são aqueles produzidos no intervalo I_1, isto é, $\hat{J}(\infty)=\hat{J}(I_1)$. Caso a diferença entre os $\hat{J}(\infty)$ seja superior a 3%, deve-se estender os valores de N1 e N2 em cada intervalo até se encontrar uma diferença inferior a 3%. Nesse caso, utilizam-se os seguintes valores estendidos: $N1=11800$, $N2=11999$ e $\Delta N=200$ para definir o intervalo I_{1e} e $N1=11000$, $N2=11199$ e $\Delta N=200$ para definir o intervalo I_{2e}.

Portanto, a fim de obedecer o critério dos 3%, os valores adotados para o erro de regime permanente $\hat{J}(\infty)$ podem ser aqueles produzidos no intervalo I_1, isto é, $\hat{J}(\infty)=\hat{J}(I_1)$ ou aqueles produzidos no intervalo I_{1e}, isto é, $\hat{J}(\infty)=\hat{J}(I_{1e})$.

Pode-se também extrair informações da curva de aprendizagem relativas a taxa de convergência. Nesse sentido, o procedimento utilizado é obter da curva de aprendizagem o valor de pico e adota-lo como referência. Em seguida, define-se o intervalo I_3=(limite inferior : limite superior) ao primeiro intervalo de 100 amostras posterior a localização do valor de referência, de modo que a média dos valores de $\hat{J}(n)$ relativos a esse intervalo seja igual ou menor ao erro de regime permanente acrescido de 10%, isto é, I_3 é tal que $\hat{J}(I_3) \leq 1.1\times\hat{J}(\infty)$. Utiliza-se o intervalo I_3 como parâmetro de comparação entre os algoritmos, avaliando a rapidez da convergência para um valor igual ou imediatamente inferior ao erro de regime permanente acrescido 10%. Em outras palavras, pode-se comparar a taxa de convergência entre os algoritmos verificando-se qual curva de aprendizagem possui o menor limite inferior do intervalo I_3, isto é, qual curva atinge mais rapidamente a primeira região em que se constata $\hat{J}(I_3) \leq 1.1\times\hat{J}(\infty)$.

Para cada um dos algoritmos adaptativos citados, define-se a curva de aprendizagem relacionada ao processo AR1 de CA_AR1, ao processo AR2 de CA_AR2 e ao processo AR3 de CA_AR3. As Tabelas 1, 2 e 3, que serão detalhadas a seguir, apresentam diversos valores extraídos dessas curvas de aprendizagem. A Tabela 1 apresenta os resultados relacionados a curva CA_AR1, a Tabela 2 refere-se a curva CA_AR2 e a Tabela 3 refere-se a curva CA_AR3.

Nessas tabelas, tem-se que a primeira coluna representa os valores de referência ou de pico, a segunda coluna o intervalo I_3 e os respectivos valores de $\hat{J}(I_3)$, a terceira coluna os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(\infty)$ e também os valores de $\hat{J}(I_{1e})$ e $\hat{J}(I_{2e})$ quando necessários e finalmente na última coluna localiza-se $\%\Delta=[\hat{J}(I_{2(e)})/\hat{J}(\infty) -1]\times100$, que informa em termos percentuais a diferença entre os $\hat{J}(\infty)$ e portanto a necessidade de se utilizar ou não o intervalo estendido.

Nas subseções seguintes, estipulam-se os parâmetros relacionados a cada um dos algoritmos citados anteriormente e apresentam-se os resultados obtidos de diversas simulações.

6.3.1 <u>Simulação-7</u>: Análise do Desempenho do Algoritmo NLMS Quando Submetido a Predição Linear dos Processos AR1, AR2 e AR3

Essa análise possui os seguintes objetivos:

- Verificar a variação do erro de regime permanente, simbolizado por $\hat{J}(\infty)$, em função da variação do número de condição (*nc*).
- Avaliar a taxa de convergência do algoritmo NLMS em função da variação do *nc*.

Observando-se a Eq.(5.14) e adotando-se os valores N=4, α=0.02 e δ=10^{-4}, estimou-se as curvas de aprendizagem CA_AR1, CA_AR2 e CA_AR3 apresentadas na Fig.(6.8.a), referentes somente ao algoritmo NLMS. Diversos valores extraídos dessas curvas de aprendizagem são apresentados na primeira linha de cada uma das Tabelas 1, 2 e 3. Através da curva CA_AR1 contida nessa figura e utilizando-se o intervalo I_1, obteve-se um erro quadrático médio igual a $\hat{J}(I_1)$=1.381, ver Tabela 1; para a CA_AR2 obteve-se $\hat{J}(I_1)$=1.085, ver Tabela 2 e para a CA_AR3 obteve-se $\hat{J}(I_1)$=1.030, ver Tabela 3. Recalculando-se esses valores utilizando-se o intervalo I_2, obteve-se em relação aos valores de $\hat{J}(I_1)$ um aumento de 3.6%, isto é, $\hat{J}(I_2)$=1.431, ver Tabela 1 para CA_AR1; um aumento de 0.55%, isto é, $\hat{J}(I_2)$=1.091, ver Tabela 2 para a CA_AR2 e um aumento de 0.68%, isto é, $\hat{J}(I_2)$=1.037, ver Tabela 3 para a CA_AR3.

Com exceção da curva CA_AR1, verificou-se que os $\hat{J}(\infty)$ ficaram praticamente inalterados, isto é, os valores de $\hat{J}(I_2)$ obtidos são menos do que 1%, superiores aos respectivos valores de $\hat{J}(I_1)$. Isso indica que os intervalos I_1 e I_2 utilizados no cálculo de $\hat{J}(\infty)$, situam-se realmente na região de regime permanente. Portanto, os valores de erro de regime permanente $\hat{J}(\infty)$ adotados nos cálculos são aqueles produzidos no intervalo I_1, isto é, $\hat{J}(\infty)$=$\hat{J}(I_1)$.

Para a curva CA_AR1, define-se o erro de regime permanente $\hat{J}(\infty)$ em termos do intervalo estendido, pois ocorreu uma diferença de 3.6% entre os valores de $\hat{J}(I_1)$ e $\hat{J}(I_2)$. Para a curva CA_AR1, obteve-se um erro quadrático médio igual a $\hat{J}(I_{1e})$=1.258 e para o intervalo I_{2e} obteve-se um acréscimo de 2.86%, ou seja, $\hat{J}(I_{2e})$=1.294, logo, $\hat{J}(\infty)$=$\hat{J}(I_{1e})$=1.258.

Torna-se necessário enfatizar no comportamento da curva de aprendizagem, a influência do *nc* correspondente a cada processo de entrada, seja em termos do valor de $\hat{J}(\infty)$, seja em termos de sua velocidade de convergência. Pelos dados anteriores, verificou-se que com o aumento do valor de nc=7.83 relacionado ao processo AR3, para nc=100.04 relacionado ao processo AR1, ocorreu um correspondente aumento no valor do erro quadrático médio de regime permanente de $\hat{J}(\infty)$=1.030 para $\hat{J}(\infty)$=1.258, o que eqüivale a uma acréscimo em torno de 22%.

Os detalhes das curvas CA_AR1, CA_AR2 e CA_AR3 apresentados na Fig.(6.8.b) e os dados das Tabelas 1, 2 e 3 auxiliam na avaliação da taxa de convergência do algoritmo NLMS submetido aos processos AR1, AR2 e AR3. Considerando-se a curva CA_AR1 dessa figura tem-se que o seu valor de pico $\hat{J}(10)$=97 é adotado como valor de referência na determinação de $\hat{J}(I_3)$=1.38$\cong$ 1.1×$\hat{J}(\infty)$, onde I_3=7049:7149 denota o intervalo constituído da 7049ª até a 7149ª amostra. Esse é o primeiro intervalo cuja média obtida dos valores de $\hat{J}(n)$ relativos ao mesmo é igual ou imediatamente inferior a 1.1×$\hat{J}(\infty)$. Por último, o erro quadrático médio de regime permanente é $\hat{J}(\infty)$=1.258, conforme apresenta a primeira linha da Tabela 1. Considerando-se a curva CA_AR2, tem-se que o seu valor de pico $\hat{J}(7)$=27 é adotado como valor de referência na determinação de $\hat{J}(I_3)$=1.19$\cong$ 1.1×$\hat{J}(\infty)$, onde I_3=2515:2615, sendo o erro quadrático médio de regime permanente $\hat{J}(\infty)$=1.085, conforme apresenta a primeira linha da Tabela 2. Considerando-se a curva CA_AR3, tem-se que o seu valor de pico $\hat{J}(22)$ =7 é adotado como valor de referência na determinação de $\hat{J}(I_3)$=1.13$\cong$ 1.1×$\hat{J}(\infty)$, onde I_3=1131:1231, sendo o erro quadrático médio de regime permanente $\hat{J}(\infty)$=1.030, conforme apresenta a primeira linha da Tabela 3.

Pelos dados anteriores, verificou-se a influência do nc na taxa de convergência do algoritmo NLMS. Conclui-se que a curva CA_AR3 relacionada ao menor valor nc convergiu mais rapidamente, pois possui o menor limite inferior para o intervalo I_3. Além disso, ela também registrou o menor valor do erro de regime permanente, isto é, $\hat{J}(\infty)$=1.030.

6.3.2 Simulação-8: Análise do Desempenho do Algoritmo DST-LMS Quando Submetido a Predição Linear dos Processos AR1, AR2 e AR3

Essa análise possui os seguintes objetivos:

- Verificar a variação do erro de regime permanente em função da variação do número de condição (nc) e comparar este resultado com o obtido pelo algoritmo NLMS.
- Avaliar a taxa de convergência do algoritmo DST-LMS em função da variação do nc e comparar este resultado com o obtido pelo NLMS.

- Determinar entre os algoritmos NLMS e DST-LMS, qual deles obteve o menor erro de regime permanente e a melhor taxa de convergência, para cada um dos processos.

Observando-se as Eqs.(5.15), (5.22) e (5.23) referentes aos algoritmos DST-LMS e DCT-LMS, adotou-se N=4 ; μ=α=0.02 ; β=0.90 e δ=10^{-4}. Além disso, empregou-se a transformada DST dada pela Eq.(5.40) e no algoritmo DCT-LMS empregou-se a transformada DCT fornecida pela Eq.(6.5) [57, 58].

$$C_N^{II}(m,n) = \sqrt{\frac{2}{N}} \cdot k_m \cdot \cos\left(\frac{m \cdot (n + \frac{1}{2})}{N} \cdot \pi\right), \qquad m, n = 0, 1, \cdots, N-1 \tag{6.5}$$

$$\text{onde } k_m = \begin{cases} \frac{1}{\sqrt{2}} & ,\text{se } m = 0 \\ 1 & ,\text{se } m = 1 \end{cases}$$

onde, $C_N^{II}(m,n)$ representa cada um dos elementos da matriz de transformação da DCT-II de dimensão N x N, denotada por $\underline{\mathbf{C}}_N^{II}$. Portanto, dado $\underline{\mathbf{x}}(n) = \left[x_0(n) \quad x_1(n) \quad ... \quad x_{(N-1)}(n) \right]^T$ um vetor qualquer de N elementos, então, tem-se que os elementos $x_{DCT,m}(n)$ do vetor transformado $\underline{\mathbf{x}}_{DCT}(n)$ serão calculados pela Eq.(6.6.a).

$$x_{DCT,m}(n) = \sum_{t=0}^{N-1} C_N^{II}(m,t) x_t(n), \qquad m = 0, 1, ..., N-1 \tag{6.6.a}$$

$$\underline{\mathbf{x}}_{DCT}(n) = \left[x_{DCT,0}(n) \quad x_{DCT,1}(n) \quad .... \quad x_{DCT,(N-1)}(n) \right]^T \tag{6.6.b}$$

De forma análoga a figura anterior, apresenta-se na Fig.(6.9.a) as estimativas das curvas de aprendizagem CA_AR1, CA_AR2 e CA_AR3, referentes ao algoritmo DST-LMS. Diversos valores extraídos dessas curvas de aprendizagem são apresentados na segunda linha de cada uma das Tabelas 1, 2 e 3. Através da curva CA_AR1 contida nessa figura e utilizando-se o intervalo I_1, obteve-se um erro quadrático médio de regime permanente $\hat{J}(\infty)$=$\hat{J}(I_1)$=1.077 ver Tabela 1; para a CA_AR2 obteve-se $\hat{J}(\infty)$=$\hat{J}(I_1)$=1.060, ver Tabela 2 e para a CA_AR3 obteve-se

$\hat{J}(\infty)$=$\hat{J}(I_1)$=1.051, ver Tabela 3. Recalculando-se os valores de cada um dos $\hat{J}(\infty)$, só que utilizando-se o intervalo I_2, obteve-se em relação aos valores de $\hat{J}(\infty) = \hat{J}(I_1)$ um aumento de 1,1%, isto é, $\hat{J}(\infty)$=$\hat{J}(I_2)$=1.089, ver Tabela 1 para CA_AR1; uma diminuição de 0.19%, isto é, $\hat{J}(\infty)$=$\hat{J}(I_2)$=1.058, ver Tabela 2 para a CA_AR2 e um aumento de 0.28%, isto é, $\hat{J}(\infty)$=$\hat{J}(I_2)$=1.054, ver Tabela 3 para a CA_AR3. Verifica-se que os $\hat{J}(\infty)$ calculados para cada um dos intervalos ficaram praticamente inalterados, pois os valores obtidos de $\hat{J}(I_2)$ ficaram 1% acima ou abaixo dos valores de $\hat{J}(I_1)$. Conclui-se então, que os intervalos I_1 e I_2 utilizados no cálculo de $\hat{J}(\infty)$ situam-se realmente na região de regime permanente. Dessa forma, a título de referência, os valores de erro de regime permanente $\hat{J}(\infty)$ adotados nos cálculos relacionados a esse algoritmo são aqueles produzidos no intervalo I_1, isto é, $\hat{J}(\infty)$=$\hat{J}(I_1)$.

Pelos dados anteriores, verificou-se que com o aumento do valor de nc=7.83, relacionado ao processo AR3, para nc=100.04, relacionado ao processo AR1, ocorreu um correspondente aumento no valor do erro quadrático médio de regime permanente, de $\hat{J}(\infty)$=1.051 para $\hat{J}(\infty)$=1.077, o que equivale a uma acréscimo de 2.5%. Isso demonstra uma elevada insensibilidade desse algoritmo com relação a variação dos valores de nc, pois os $\hat{J}(\infty)$ mantiveram-se praticamente inalterados com a variação dos "*nc*" relacionados a cada um dos processos de entrada. Nesse aspecto, verifica-se um desempenho superior do algoritmo DST-LMS sobre o NLMS, pois o primeiro resultou numa variação do $\hat{J}(\infty)$ de apenas 2.5%, enquanto o segundo registrou uma variação de 22.1%.

Os detalhes das curvas CA_AR1, CA_AR2 e CA_AR3 apresentados na Fig.(6.9.b) e os dados das Tabelas 1,2 e 3 auxiliam na avaliação da taxa de convergência do algoritmo DST-LMS, submetido aos processos AR1, AR2 e AR3. Considerando-se a curva CA_AR1 dessa figura, tem-se que o seu valor de pico $\hat{J}(14)$=62 é adotado como valor de referência na determinação de $\hat{J}(I_3)$=1.18 $\cong$ 1.1×$\hat{J}(\infty)$, onde I_3=3220:3320. Por último, o erro quadrático médio de regime permanente é $\hat{J}(\infty)$=1.077, conforme apresenta a segunda linha da Tabela 1. Considerando-se a curva CA_AR2, tem-se que o valor de pico dessa curva $\hat{J}(12)$=46 é adotado como valor de referência na determinação de $\hat{J}(I_3)$=1.17 $\cong$ 1.1×$\hat{J}(\infty)$, onde I_3=762:862, sendo o erro quadrático médio de regime permanente $\hat{J}(\infty)$=1.06, conforme apresenta a segunda linha da Tabela 2. Considerando-se a curva CA_AR3, tem-se que o valor de pico dessa curva $\hat{J}(9)$=22 é adotado como valor de referência na determinação de $\hat{J}(I_3)$=1.15 $\cong$ 1.1×$\hat{J}(\infty)$, onde I_3=194:294 e o erro quadrático médio de regime permanente $\hat{J}(\infty)$=1.051, conforme apresenta a segunda linha da Tabela 3. Através desses dados, verificou-se a influência do *nc* na taxa de convergência do algoritmo DST-LMS. Conclui-se que a curva CA_AR3 relacionada ao menor valor *nc*, convergiu mais rapidamente, pois obteve o menor limite inferior para o intervalo I_3. Além disso, ela também registrou o menor valor do erro de regime permanente, isto é, $\hat{J}(\infty)$=1.051.

Compara-se a taxa de convergência dos algoritmos NLMS e DST-LMS verificando-se qual das curvas de aprendizagem desses algoritmos possui em seu intervalo I_3 o menor limite inferior. Considerando-se a curva CA_AR1, tem-se um I_3=7049:7149 para o NLMS e um I_3=3220:3320 para o DST-LMS. Logo, o primeiro algoritmo é mais lento que o segundo, pois o mesmo obteve um I_3 com um limite inferior bem maior do que o do DST-LMS. Naturalmente, necessita-se examinar o valor do $\hat{J}(\infty)$ para ambos os casos. Verifica-se que $\hat{J}(\infty)$=1.258 para o NLMS e que $\hat{J}(\infty)$=1.077 para o DST-LMS. Logo, além de mais lento o NLMS registrou um $\hat{J}(\infty)$ 16.8% superior em relação ao obtido pelo DST-LMS.

Considerando-se a curva CA_AR2, tem-se um I_3=2515:2615 para o NLMS e um I_3=762:862 para o DST-LMS. Logo, o primeiro algoritmo continua bem mais lento que o segundo. Verificando-se o erro de regime permanente, tem-se que $\hat{J}(\infty)$=1.085 para o NLMS e que $\hat{J}(\infty)$=1.060 para o DST-LMS. Novamente, além de mais lento, o NLMS registrou um $\hat{J}(\infty)$ 2.3% superior em relação ao do DST-LMS. Considerando-se a curva CA_AR3, tem-se um I_3=1131:1231 para o NLMS e um I_3=194:294 para o DST-LMS. Logo, o primeiro algoritmo é bem mais lento que o segundo. Verificando-se o erro de regime permanente, tem-se que $\hat{J}(\infty)$=1.030 para o NLMS e que $\hat{J}(\infty)$=1.051 para o DST-LMS. Logo, apesar de bem mais rápido, o DST-LMS registrou um $\hat{J}(\infty)$ 2% superior em relação ao NLMS.

6.3.3 Simulação-9: Análise do Desempenho do Algoritmo DCT-LMS Quando Submetido a Predição Linear dos Processos AR1, AR2 e AR3

Essa análise possui os seguintes objetivos:

- Verificar a variação do erro de regime permanente em função da variação do número de condição (*nc*) e comparar este resultado com os obtidos pelos algoritmos NLMS e DST-LMS.
- Avaliar a taxa de convergência do algoritmo DCT-LMS em função da variação do *nc* e comparar este resultado com os obtidos pelos algoritmos NLMS e DST-LMS.

- Determinar entre os algoritmos NLMS, DST-LMS e DCT-LMS, qual deles obteve o menor erro de regime permanente e a melhor taxa de convergência, para cada um dos processos.

Apresenta-se na Fig.(6.10.a) as curvas de aprendizagem CA_AR1, CA_AR2 e CA_AR3, referentes ao algoritmo DCT-LMS. Os valores extraídos dessas curvas de aprendizagem são apresentados na terceira linha das Tabelas 1, 2 e 3. Foram calculados para cada uma das curvas os valores de $\hat{J}(I_1)$ e $\hat{J}(I_2)$ e pela mesma justificativa dos casos anteriores adotou-se o valor de $\hat{J}(I_1)$ como sendo o erro de regime permanente $\hat{J}(\infty)$ utilizado nos cálculos relacionados a esse algoritmo. Pelos valores dessas tabelas verificou-se que com a mudança do valor de nc=7.83 relacionado ao processo AR3, para nc=44.25 relacionado ao processo AR2, ocorreu um leve aumento no valor do erro quadrático médio de regime permanente, de $\hat{J}(\infty)$=1.052 para $\hat{J}(\infty)$=1.059, o que equivale a uma acréscimo de 0.66%. Com a mudança de nc=7.83 para nc=100.04 relacionado ao processo AR1, ocorreu um aumento de 0.47% no valor de $\hat{J}(\infty)$. Isso demonstra uma elevada insensibilidade desse algoritmo pois os valores de $\hat{J}(\infty)$ mantiveram-se praticamente inalterados com a variação dos "*nc*".

Com a mudança dos extremos nc=7.83 para nc=100.04 verificou-se um desempenho superior do DCT-LMS sobre os algoritmos DST-LMS e NLMS, pois para o primeiro houve uma variação de apenas 0.47% sobre o valor do $\hat{J}(\infty)$, para o segundo de 2.5% e para o terceiro uma variação de 22.1%.

Os detalhes das curvas CA_AR1, CA_AR2 e CA_AR3 apresentados na Fig.(6.10.b) e os dados das Tabelas 1,2 e 3 auxiliam na avaliação da taxa de convergência do algoritmo DCT-LMS, submetido aos processos AR1, AR2 e AR3. Verifica-se nesses detalhes que a curva CA_AR1 possuiu a convergência mais lenta e por conseguinte o I_3 com maior limite inferior, ou seja, I_3=394:494. A curva CA_AR2 obteve a convergência mais rápida, no entanto, muito próxima da convergência registrada pela curva CA_AR3 cujos valores são respectivamente, I_3=237:337 e I_3=282:382. O menor erro quadrático médio de regime permanente foi obtido pela curva CA_AR3, ou seja, $\hat{J}(\infty)$=1.052.

Comparou-se a taxa de convergência entre os algoritmos NLMS, DST-LMS e DCT-LMS e verificou-se que para a curva CA_AR1 o algoritmo DCT-LMS foi mais rápido pois obteve o I_3 com menor limite inferior, ou seja, I_3=394:494 ; para a curva CA_AR2, o mais rápido foi o

DCT-LMS com I_3=237:337 e para a curva CA_AR3, o mais rápido foi o DST-LMS com I_3=194:294.

O menor erro de regime permanente foi obtido pelo DCT-LMS para CA_AR1, ou seja, $\hat{J}(\infty)$=1.057 ; pelo DCT-LMS para CA_AR2, ou seja, $\hat{J}(\infty)$=1.059 e pelo NLMS para CA_AR3, ou seja, $\hat{J}(\infty)$=1.030.

6.3.4 <u>Simulação-10</u>: Análise do Desempenho do Algoritmo D-LMS Quando Submetido a Predição Linear dos Processos AR1, AR2 e AR3

Essa análise possui os seguintes objetivos:

- Verificar a variação do erro de regime permanente em função da variação do número de condição (*nc*) e comparar este resultado com os obtidos pelos algoritmos NLMS, DST-LMS e DCT-LMS.
- Avaliar a taxa de convergência do algoritmo D-LMS em função da variação do *nc* e comparar este resultado com os obtidos pelos algoritmos NLMS, DST-LMS e DCT-LMS.
- Determinar entre os algoritmos NLMS, DST-LMS, DCT-LMS e D-LMS, qual deles obteve o menor erro de regime permanente e a melhor taxa de convergência, para cada um dos processos.

Observando-se as Eqs.(5.37), (5.38) e (5.39) referentes ao algoritmo D-LMS, adotou-se N=4; $\mu=\alpha=0.02$; $\beta=0.90$ e $\delta=10^{-4}$. Apresenta-se na Fig.(6.11.a) as curvas de aprendizagem CA_AR1, CA_AR2 e CA_AR3, referentes ao algoritmo D-LMS. Os valores extraídos dessas curvas de aprendizagem são apresentados na quarta linha de cada uma das três tabelas mencionadas. Foram calculados para cada uma das curvas, os valores de $\hat{J}(I_1)$ e $\hat{J}(I_2)$ e pela mesma justificativa dos casos anteriores, adotou-se o valor de $\hat{J}(I_1)$ como sendo o erro de regime permanente $\hat{J}(\infty)$ utilizado nos cálculos relacionados a esse algoritmo. Pelos valores dessas tabelas, verificou-se que com a mudança do valor de *nc*=7.83 relacionado ao processo AR3, para *nc*=44.25 relacionado ao processo AR2, ocorreu um aumento de 0.8% no $\hat{J}(\infty)$. Com a mudança de *nc*=7.83 para *nc*=100.04 relacionado ao processo AR1, não foi alterado o valor de $\hat{J}(\infty)$. Isso demonstra uma elevada insensibilidade desse algoritmo, pois o $\hat{J}(\infty)$ manteve-se praticamente inalterado com a variação dos "*nc*".

Com a mudança dos extremos de *nc*=7.83 para *nc*=100.04 verificou-se um desempenho superior do D-LMS sobre os algoritmos DCT-LMS, DST-LMS e NLMS, pois para o primeiro não houve variação, para o segundo houve uma variação do $\hat{J}(\infty)$ de apenas 0.47%, para o terceiro de 2.5% e para o quarto houve uma variação de 22.1%.

Os detalhes das curvas CA_AR1, CA_AR2 e CA_AR3 apresentados na Fig.(6.11.b) juntamente com os valores exibidos nas três tabelas, auxiliam na avaliação da taxa de convergência do algoritmo D-LMS submetido aos processos AR1, AR2 e AR3. Verifica-se nesses detalhes que a curva CA_AR1 possuiu a convergência mais lenta e por conseguinte o I_3 com maior limite inferior, ou seja, I_3=995:1095. A curva CA_AR3 obteve a convergência mais rápida com I_3=425:525. O menor erro quadrático médio de regime permanente $\hat{J}(\infty)$=1.076 foi obtido pela curva CA_AR1 e pela curva CA_AR3, no entanto, o erro de regime permanente registrado pela curva CA_AR2 ficou muito próximo desse valor, com uma diferença de apenas 0.84%, ou seja, $\hat{J}(\infty)$=1.085.

Comparou-se a taxa de convergência entre os algoritmos NLMS, DST-LMS, DCT-LMS e D-LMS e verificou-se que para a curva CA_AR1 o algoritmo DCT-LMS foi mais rápido, pois obteve o I_3 com menor limite inferior, ou seja, I_3=394:494, em seguida veio o algoritmo D-LMS com I_3=995:1095. Para a curva CA_AR2, o mais rápido foi o DCT-LMS com I_3=237:337, em seguida veio o D-LMS com I_3=457:557. Para a curva CA_AR3, o mais rápido foi o DST-LMS com I_3=194:294, em seguida vieram o DCT-LMS, o D-LMS e por último o NLMS.

O menor erro de regime permanente foi obtido pelo DCT-LMS para CA_AR1, ou seja, $\hat{J}(\infty)$=1.057, em seguida veio o D-LMS com uma diferença de 1.8%, ou seja, $\hat{J}(\infty)$=1.076. Para a curva CA_AR2, o menor erro foi obtido pelo DCT-LMS com $\hat{J}(\infty)$=1.059 e para a curva CA_AR3, o menor erro foi obtido pelo NLMS com $\hat{J}(\infty)$=1.030.

6.3.5 Simulação-11: Análise do Desempenho do Algoritmo DSTr-LMS Quando Submetido a Predição Linear dos Processos AR1, AR2 e AR3

Essa análise possui os seguintes objetivos:

- Verificar a variação do erro de regime permanente em função da variação do número de condição (*nc*) e comparar este resultado com os obtidos pelos algoritmos NLMS, DST-LMS, DCT-LMS e D-LMS.

- Avaliar a taxa de convergência do algoritmo DSTr-LMS em função da variação do *nc* e comparar este resultado com os obtidos pelos algoritmos NLMS, DST-LMS, DCT-LMS e D-LMS.
- Determinar entre os algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS, qual deles obteve o menor erro de regime permanente e a melhor taxa de convergência, para cada um dos processos.

Observando-se as Eqs.(5.48), (5.49) e (5.50) referentes ao algoritmo DSTr-LMS, adotou-se N=4; $\mu=\alpha=0.02$; $\beta=0.90$ e $\delta=10^{-4}$. Apresenta-se na Fig.(6.12.a) as curvas de aprendizagem CA_AR1, CA_AR2 e CA_AR3, referentes ao algoritmo DSTr-LMS. Os valores extraídos dessas curvas de aprendizagem são apresentados na quinta linha de cada uma das três tabelas mencionadas. Foram calculados para cada uma das curvas os valores de $\hat{J}(I_1)$ e $\hat{J}(I_2)$ e pela mesma justificativa dos casos anteriores adotou-se o valor de $\hat{J}(I_1)$ como sendo o erro de regime permanente $\hat{J}(\infty)$ utilizado nos cálculos relacionados a esse algoritmo.

Pelos valores dessas tabelas, verificou-se que com a mudança do valor de nc=7.83 relacionado ao processo AR3, para nc=44.25 relacionado ao processo AR2, ocorreu um aumento de 0.9% no $\hat{J}(\infty)$. Com a mudança de nc=7.83 para nc=100.04 relacionado ao processo AR1, ocorreu um aumento de 0.2% no valor de $\hat{J}(\infty)$. Isso demonstra uma elevada insensibilidade desse algoritmo, pois os $\hat{J}(\infty)$ mantiveram-se praticamente inalterados com a variação dos "*nc*".

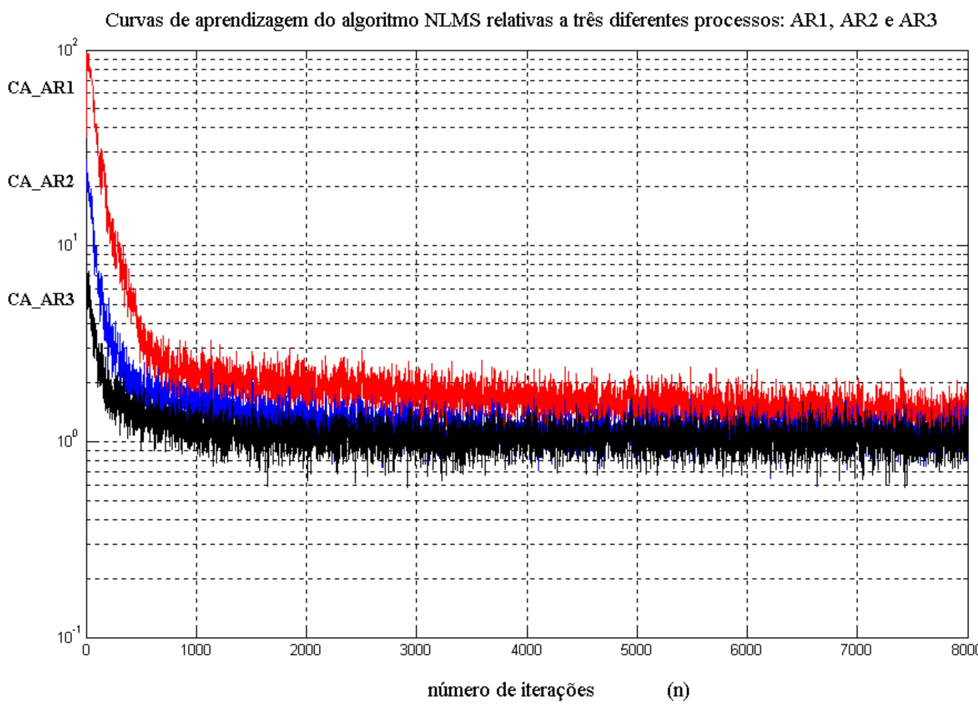
Com a mudança dos extremos nc=7.83 para nc=100.04 verificou-se um desempenho superior do D-LMS e do DSTr-LMS sobre os algoritmos DCT-LMS, DST-LMS e NLMS, pois para o primeiro não houve uma variação do $\hat{J}(\infty)$, para o segundo houve uma variação de apenas 0.2% , para o terceiro houve uma variação de 0.47%, para o quarto de 2.5% e para o quinto uma variação de 22.1%.

Os detalhes das curvas CA_AR1, CA_AR2 e CA_AR3 apresentados na Fig.(6.12.b) juntamente com os valores exibidos nas três tabelas, auxiliam na avaliação da taxa de convergência do algoritmo DSTr-LMS submetido aos processos AR1, AR2 e AR3. Verifica-se nesses detalhes que a curva CA_AR1 obteve a convergência mais lenta e por conseguinte o I_3 com maior limite inferior, ou seja, I_3=134:234. A curva CA_AR3 obteve a convergência mais rápida com I_3=75:175. O menor erro quadrático médio de regime permanente $\hat{J}(\infty)$=1.107, foi obtido pela curva CA_AR3, no entanto, os erros de regime permanente registrados pelas curvas

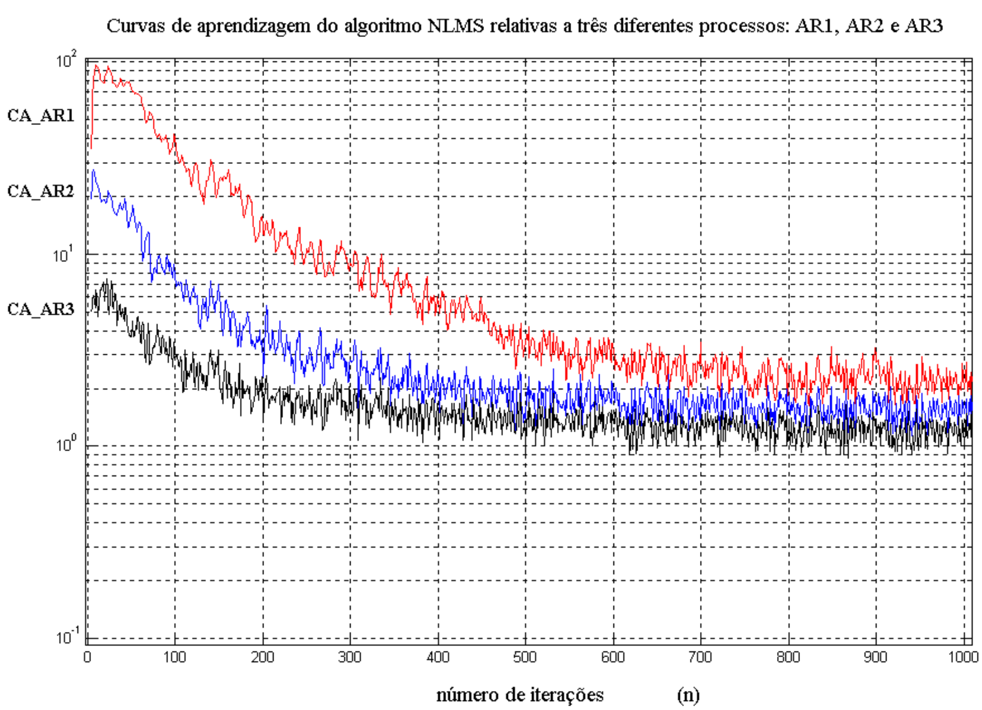
CA_AR2 e CA_AR1 ficaram muito próximos desse valor, com um aumento de apenas 0.2% para a CA_AR1 e um aumento de 0.9% para a CA_AR2.

Comparou-se a taxa de convergência entre os algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS e verificou-se que para a curva CA_AR1, o algoritmo DSTr-LMS foi mais rápido pois obteve o I_3 com menor limite inferior, ou seja, I_3=134:234, em seguida veio o algoritmo DCT-LMS com I_3=394:494. Para a curva CA_AR2, o mais rápido foi o DSTr-LMS com I_3=88:188, em seguida veio o DCT-LMS com I_3=237:337. Para a curva CA_AR3, o mais rápido foi novamente o DSTr-LMS com I_3=75:175, em seguida vieram o DST-LMS com I_3=194:294, o DCT-LMS, o D-LMS e por último o NLMS.

Para a CA_AR1, o erro de regime permanente obtido pelo DSTr-LMS é 4.9% superior em relação ao menor erro obtido para essa curva, pelo DCT-LMS. Para a CA_AR2, o erro de regime permanente obtido pelo DSTr-LMS é 5.5% superior em relação ao menor erro obtido para essa curva, pelo DCT-LMS. Para a CA_AR3, o erro de regime permanente obtido pelo DSTr-LMS é 7.5% superior em relação ao menor erro obtido para essa curva, pelo NLMS e 5.3% superior em relação ao erro obtido pelo DCT-LMS.

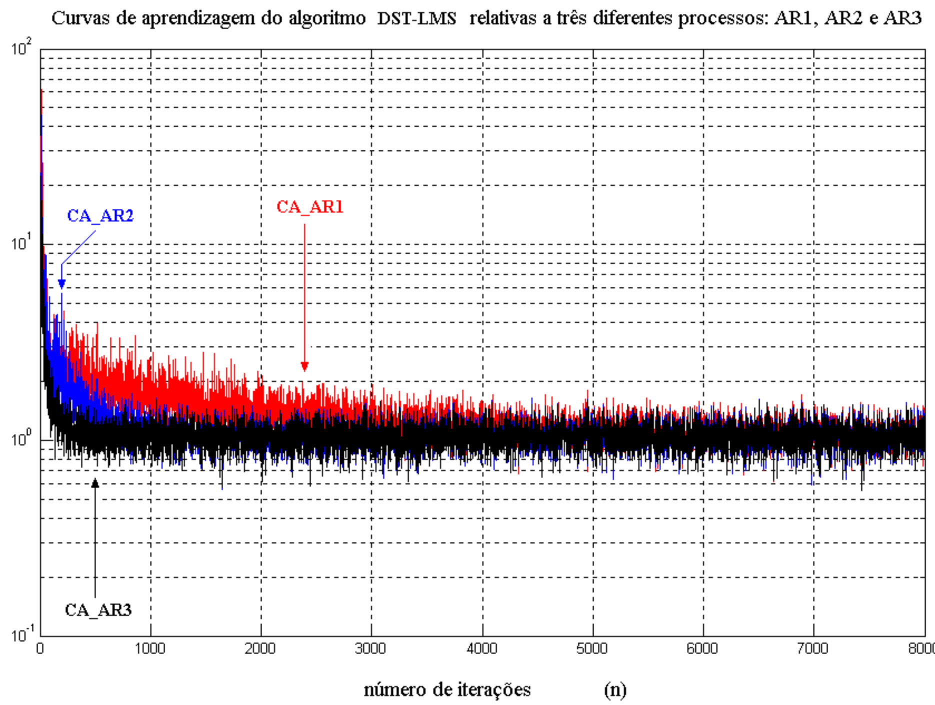


a)

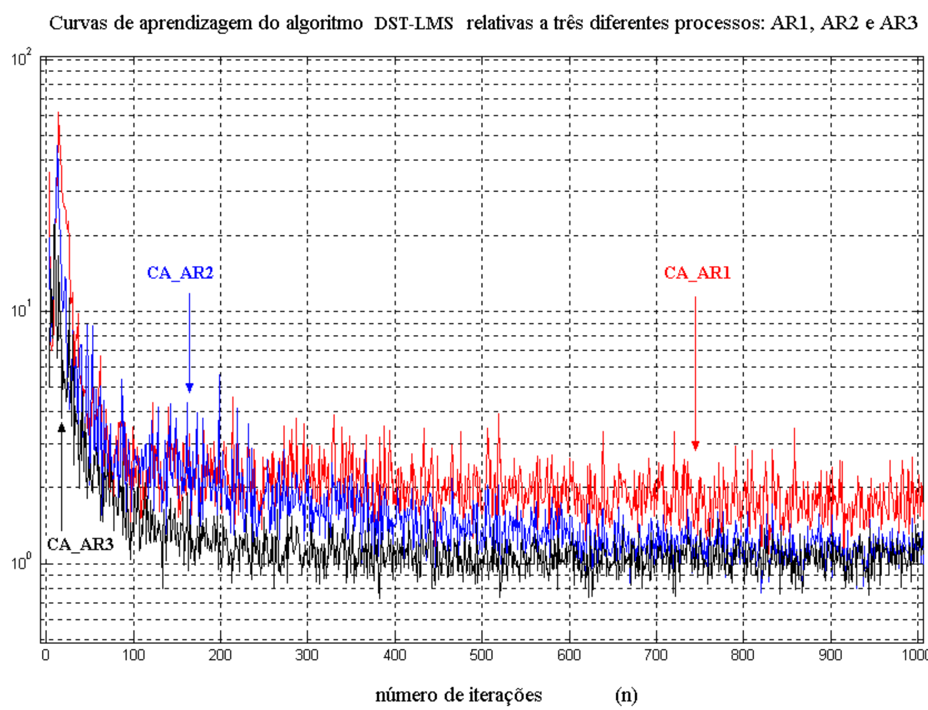


b)

Fig.6.8. a)Curvas de aprendizagem para o algoritmo NLMS usado em predição linear, considerando-se que N=4 ; α=0.02 ; δ= 10^{-4}. b)Detalhes dessa curva.

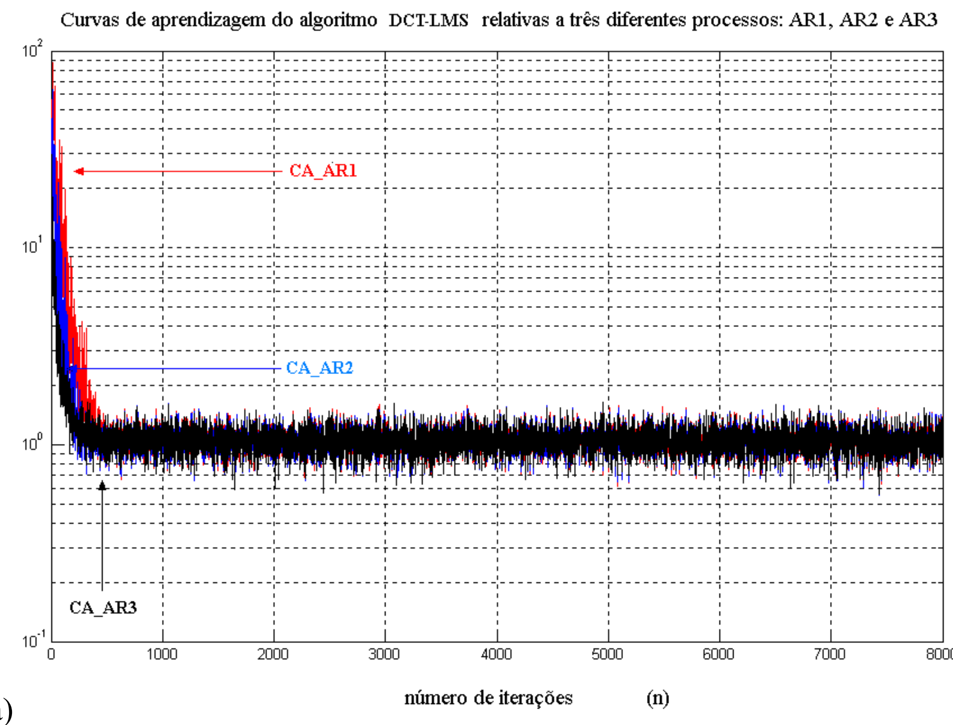


a)

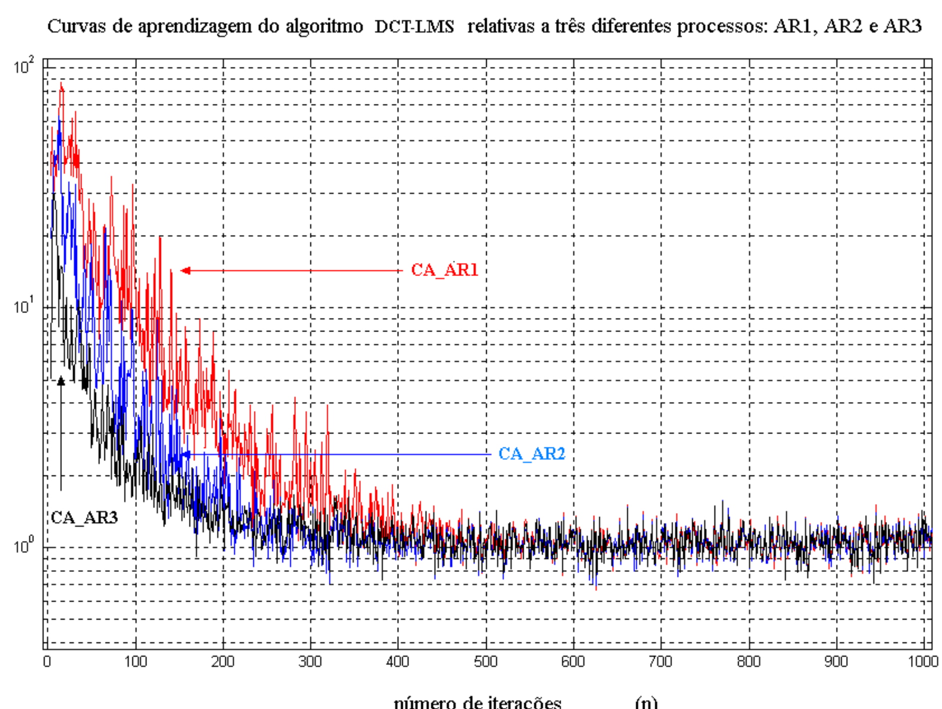


b)

Fig.6.9. a)Curvas de aprendizagem para o algoritmo DST-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

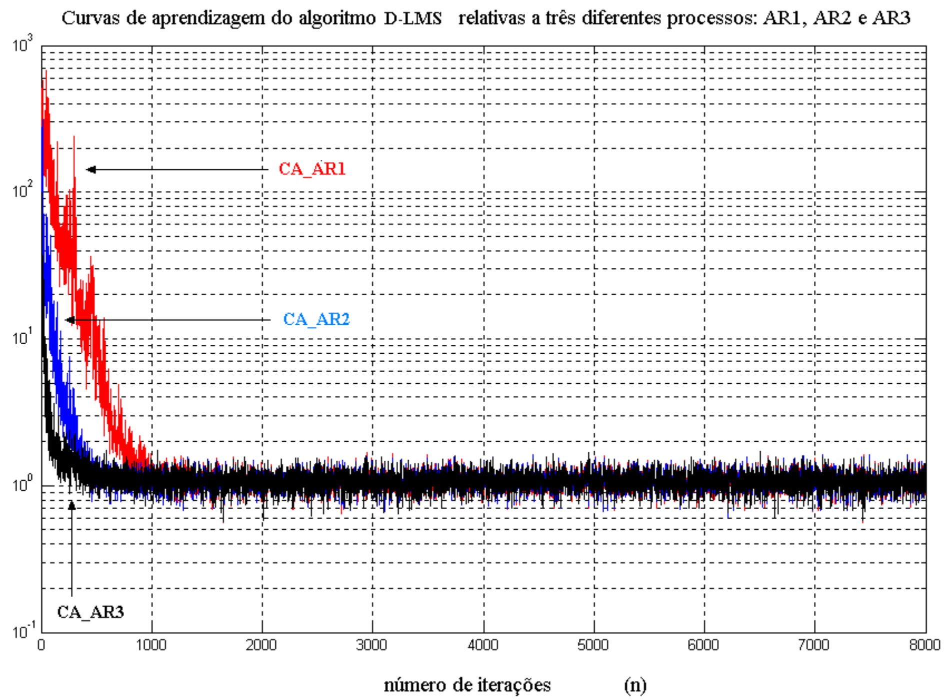


a)

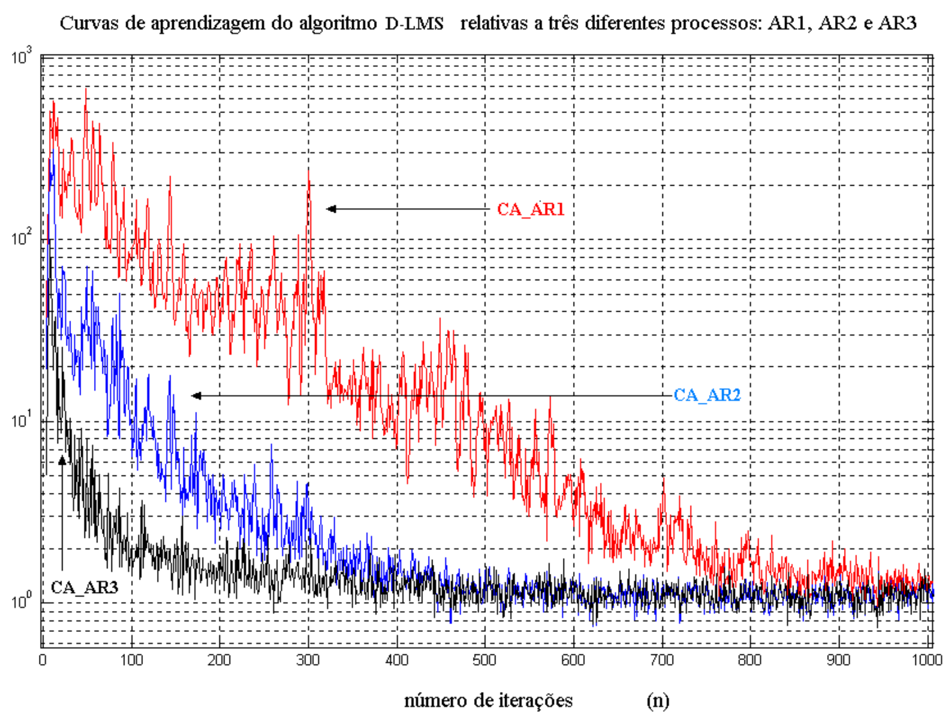


b)

Fig.6.10. a)Curvas de aprendizagem para o algoritmo DCT-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

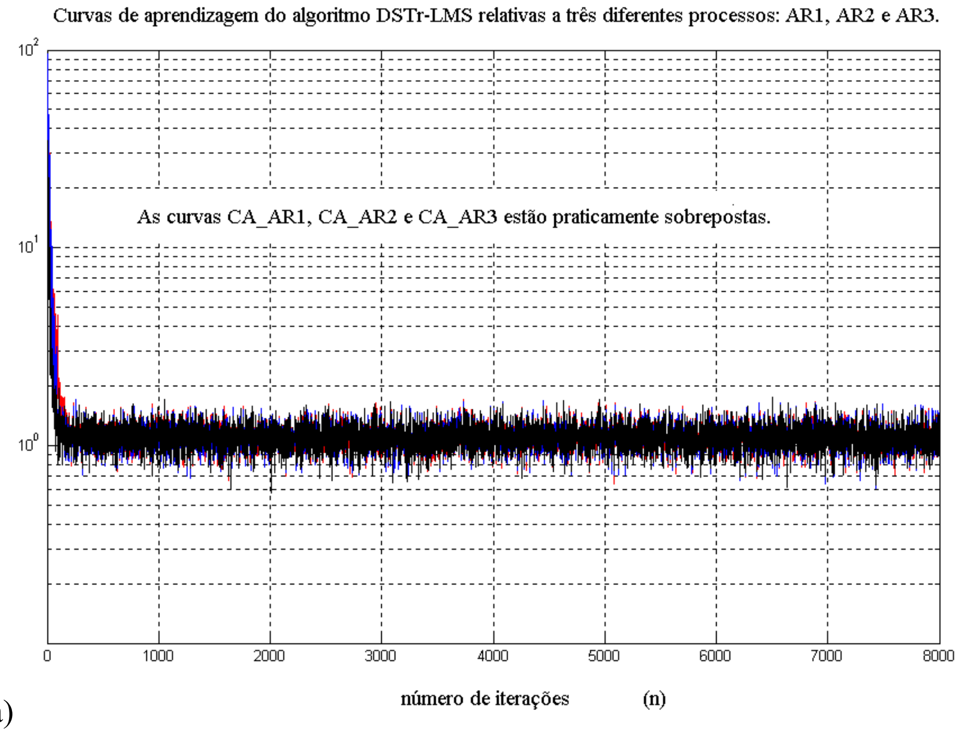


a)

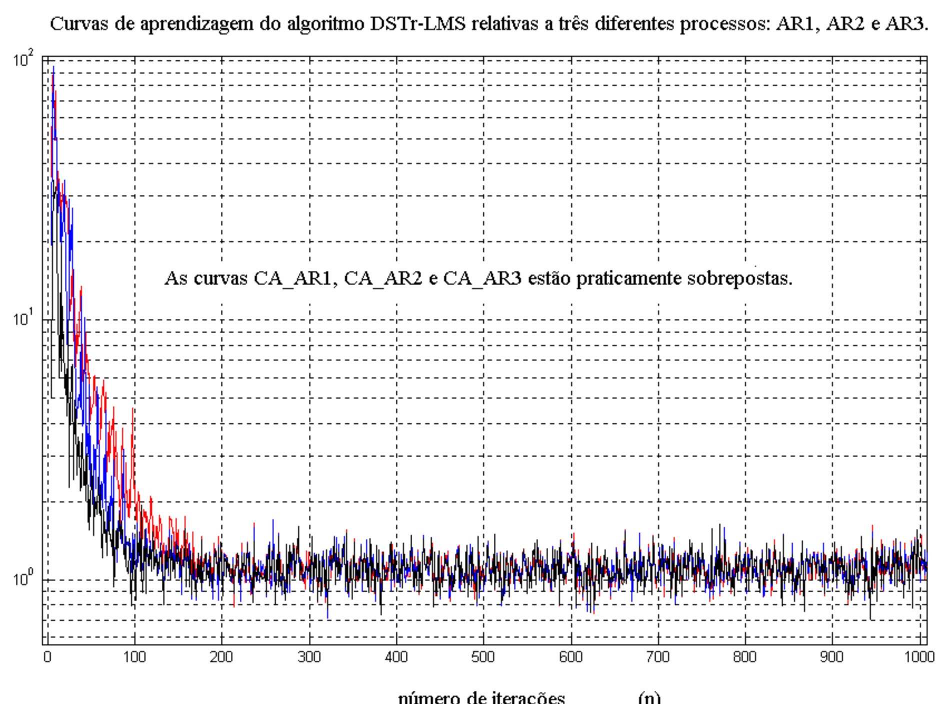


b)

Fig.6.11. a)Curvas de aprendizagem para o algoritmo D-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.



a)



b)

Fig.6.12. a)Curvas de aprendizagem para o algoritmo DSTr-LMS usado em predição linear, considerando-se que $N=4$; $\mu=0.02$; $\beta=0.90$; $\delta= 10^{-4}$. b)Detalhes dessa curva.

	Processo AR1 ; $\alpha = \mu = 0.02$, $\beta = 0.90$ e $\delta = 10^{-4}$ (Dados obtidos da Curva CA_AR1, relacionada ao processo AR1)			
	Valor de pico	Valores Médios $\hat{J}(I_3) \leq 1.1 \times \hat{J}(\infty)$ (Intervalo I_3 é variável)	Erro de regime permanente $\hat{J}(\infty)$ $\hat{J}(\infty) = \hat{J}(I_1) \cong \hat{J}(I_2)$ ou $\hat{J}(\infty) = \hat{J}(I_{1e}) \cong \hat{J}(I_{2e})$ $I_1 = 7800{:}7999$; $I_2 = 7000{:}7199$ $I_{1e} = 11800{:}11999$; $I_{2e} = 11000{:}11199$	$D = \hat{J}(I_{2(e)})/\hat{J}(\infty)$; $\%\Delta = (D-1)\times 100$
NLMS	-	-	$\hat{J}(I_1) = 1.381$; $\hat{J}(I_2) = 1.431$	3.6 %
	$\hat{J}(10) = 97$	$\hat{J}(7049{:}7149) = 1.38$	$\hat{J}(\infty) = \hat{J}(I_{1e}) = 1.258$; $\hat{J}(I_{2e}) = 1.294$	2.86 %
DST-LMS	$\hat{J}(14) = 62$	$\hat{J}(3220{:}3320) = 1.18$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.077$; $\hat{J}(I_2) = 1.089$	1.11%
DCT-LMS	$\hat{J}(15) = 87$	$\hat{J}(394{:}494) = 1.16$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.057$; $\hat{J}(I_2) = 1.057$	0
D-LMS	$\hat{J}(48) = 678$	$\hat{J}(995{:}1095) = 1.18$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.076$; $\hat{J}(I_2) = 1.079$	0.27%
DSTr-LMS	$\hat{J}(5) = 88$	$\hat{J}(134{:}234) = 1.22$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.109$; $\hat{J}(I_2) = 1.112$	0.27%

Tabela 1. Apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_{1e})$, $\hat{J}(I_{2e})$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e $\%\Delta$ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS, sujeitos aos parâmetros $\alpha = \mu = 0.02$, $\beta = 0.90$, $\delta = 10^{-4}$ e processo AR1.

	Processo AR2 ; $\alpha = \mu = 0.02$, $\beta = 0.90$ e $\delta = 10^{-4}$ (Dados obtidos da Curva CA_AR2, relacionada ao processo AR2)			
	Valor de pico	Valores Médios $\hat{J}(I_3) \leq 1.1 \times \hat{J}(\infty)$ (Intervalo I_3 é variável)	Erro de regime permanente $\hat{J}(\infty)$ $\hat{J}(\infty) = \hat{J}(I_1) \cong \hat{J}(I_2)$ $I_1 = 7800{:}7999$; $I_2 = 7000{:}7199$	$D = \hat{J}(I_2)/\hat{J}(\infty)$; $\%\Delta = (D-1)\times 100$
NLMS	$\hat{J}(7) = 27$	$\hat{J}(2515{:}2615) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.085$; $\hat{J}(I_2) = 1.091$	0.55 %
DST-LMS	$\hat{J}(12) = 46$	$\hat{J}(762{:}862) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.060$; $\hat{J}(I_2) = 1.058$	-0.19%
DCT-LMS	$\hat{J}(12) = 63$	$\hat{J}(237{:}337) = 1.16$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.059$; $\hat{J}(I_2) = 1.054$	-0.47%
D-LMS	$\hat{J}(12) = 312$	$\hat{J}(457{:}557) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.085$; $\hat{J}(I_2) = 1.083$	-0.18%
DSTr-LMS	$\hat{J}(6) = 95$	$\hat{J}(88{:}188) = 1.21$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.117$; $\hat{J}(I_2) = 1.110$	-0.63%

Tabela 2. Apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e $\%\Delta$ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS, sujeitos aos parâmetros $\alpha = \mu = 0.02$, $\beta = 0.90$, $\delta = 10^{-4}$ e processo AR2.

	Processo AR3 ; $\alpha = \mu = 0.02$, $\beta = 0.90$ e $\delta = 10^{-4}$ (Dados obtidos da Curva CA_AR3, relacionada ao processo AR3)			
	Valor de pico	Valores Médios $\hat{J}(I_3) \leq 1.1 \times \hat{J}(\infty)$ (Intervalo I_3 é variável)	Erro de regime permanente $\hat{J}(\infty)$ $\hat{J}(\infty) = \hat{J}(I_1) \cong \hat{J}(I_2)$ $I_1 = 7800{:}7999$; $I_2 = 7000{:}7199$	$D = \hat{J}(I_2)/\hat{J}(\infty)$; $\%\Delta = (D-1)\times 100$
NLMS	$\hat{J}(22) = 7$	$\hat{J}(1131{:}1231) = 1.13$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.030$; $\hat{J}(I_2) = 1.037$	0.68 %
DST-LMS	$\hat{J}(9) = 22$	$\hat{J}(194{:}294) = 1.15$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.051$; $\hat{J}(I_2) = 1.054$	0.28%
DCT-LMS	$\hat{J}(6) = 30$	$\hat{J}(282{:}382) = 1.15$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.052$; $\hat{J}(I_2) = 1.053$	0.01%
D-LMS	$\hat{J}(8) = 99$	$\hat{J}(425{:}525) = 1.18$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.076$; $\hat{J}(I_2) = 1.082$	0.56%
DSTr-LMS	$\hat{J}(6) = 35$	$\hat{J}(75{:}175) = 1.22$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.107$; $\hat{J}(I_2) = 1.110$	0.27%

Tabela 3. Apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e $\%\Delta$ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS, sujeitos aos parâmetros $\alpha = \mu = 0.02$, $\beta = 0.90$, $\delta = 10^{-4}$ e processo AR3.

6.4 Grupo de Simulações que Analisa o Desempenho dos Algoritmos Dr_i-LMS e Dr_{ij}-LMS

Denomina-se Dr_0-LMS ao algoritmo D-LMS que exclui a entrada decomposta $r_0(n)$, isto é, não a utiliza. Essa entrada é uma das componentes do vetor **u**(n) definido pela Eq.(5.29.a), o qual está inserido na expressão de atualização dos pesos denotada pela Eq.(5.37). Caso duas entradas sejam excluídas, por exemplo $r_0(n)$ e $r_1(n)$, então, o algoritmo denominar-se-á de $Dr_{0\,1}$-LMS.

Cada simulação desse grupo objetiva analisar o desempenho dos algoritmos Dr_i-LMS e Dr_{ij}-LMS com $0 \leq i, j \leq N-1$ (vide Tabelas 4, 5 e 6), inclusive comparando-o com o dos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS.

Esses algoritmos são empregados no esquema de predição linear dado pela Fig.(6.1), com $\Delta = 1$, quando submetidos a três processos autoregressivos de entrada com diferentes valores para o número de condição (*nc*). Tanto a análise individual quanto a comparação entre os algoritmos são baseadas nos mesmos critérios de desempenho relatados anteriormente.

Para cada variação do algoritmo D-LMS, isto é, Dr_i-LMS ou $Dr_{i\,j}$-LMS, definem-se as curvas de aprendizagem desses algoritmos relacionadas ao processo AR1 de CA_AR1, ao processo AR2 de CA_AR2 e ao processo AR3 de CA_AR3. As Tabelas 4, 5 e 6, que serão detalhadas a seguir, apresentam diversos valores extraídos dessas curvas de aprendizagem. A Tabela 4 apresenta os resultados relacionados a curva CA_AR1, a Tabela 5 refere-se a curva CA_AR2 e a Tabela 6 refere-se a curva CA_AR3.

Atribuiu-se aos parâmetros das Eqs.(5.37), (5.38) e (5.39), relacionadas ao algoritmo D-LMS e suas variações, os valores $N=4$; $\mu=0.02$; $\beta=0.90$ e $\delta=10^{-4}$, necessários para geração das curvas de aprendizagem e tabelas mencionadas. A seguir apresentam-se os resultados obtidos de algumas simulações.

6.4.1 <u>Simulação-12</u>: Análise do Desempenho do Algoritmo Dr_0-LMS Quando Submetido a Predição Linear dos Processos AR1, AR2 e AR3

Essa análise possui os seguintes objetivos:

- Verificar a variação do erro de regime permanente em função da variação do número de condição (*nc*) e comparar este resultado com os obtidos pelos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS.
- Avaliar a taxa de convergência do algoritmo Dr_0-LMS em função da variação do *nc* e comparar este resultado com os obtidos pelos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS.
- Determinar entre os algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS, DSTr-LMS e Dr_0-LMS, qual deles obteve o menor erro de regime permanente e a melhor taxa de convergência, para cada um dos processos.

Apresenta-se na Fig.(6.13.a) as curvas de aprendizagem CA_AR1, CA_AR2 e CA_AR3, referentes ao algoritmo Dr_0-LMS. Os valores extraídos dessas curvas de aprendizagem são apresentados na primeira linha das Tabelas 4, 5 e 6. Foram calculados para cada uma das curvas, os valores de $\hat{J}(I_1)$ e $\hat{J}(I_2)$ e pela mesma justificativa dos casos anteriores adotou-se o valor de $\hat{J}(I_1)$ como sendo o erro de regime permanente $\hat{J}(\infty)$, utilizado nos cálculos relacionados a esse algoritmo. Pelos valores dessas tabelas, verificou-se que com a mudança do valor de $nc = 7.83$ relacionado ao processo AR3 para $nc = 44.25$ relacionado ao processo AR2, ocorreu um aumento de 0.85% no $\hat{J}(\infty)$. Com a mudança de $nc = 7.83$ para $nc = 100.04$ relacionado ao processo AR1, ocorreu um aumento insignificante de 0.09% no $\hat{J}(\infty)$. Isso demonstra uma elevada insensibilidade desse algoritmo, pois os valores de $\hat{J}(\infty)$ mantiveram-se praticamente inalterados com a variação dos "*nc*".

Com a mudança dos extremos de $nc = 7.83$ para $nc = 100.04$ verificou-se um desempenho superior dos algoritmos D-LMS, Dr_0-LMS e DSTr-LMS sobre os algoritmos DCT-LMS, DST-LMS e NLMS, pois para o primeiro não houve variação, para o segundo houve uma variação do $\hat{J}(\infty)$ de apenas 0.09%, para o terceiro houve uma variação do $\hat{J}(\infty)$ de 0.18%, para o quarto houve uma variação de 0.47%, para o quinto de 2.5% e para o sexto houve uma variação de 22.1%.

Os detalhes das curvas CA_AR1, CA_AR2 e CA_AR3 apresentados na Fig.(6.13.b) juntamente com os valores exibidos nas Tabelas 4, 5 e 6 auxiliam na avaliação da taxa de convergência do algoritmo Dr_0-LMS submetido aos processos AR1, AR2 e AR3. Verifica-se nesses detalhes que a curva CA_AR1 possuiu a convergência mais lenta e por conseguinte o I_3 com maior limite inferior, ou seja, I_3=2099:2199. A curva CA_AR3 obteve a convergência mais rápida com I_3=438:538.

O menor erro quadrático médio de regime permanente $\hat{J}(\infty) = 1.063$ foi obtido pela curva CA_AR3, no entanto, o erro de regime permanente registrado pela curva CA_AR2 ficou muito próximo desse valor, com uma diferença de apenas 0.85%, ou seja, $\hat{J}(\infty) = 1.072$ e para a curva CA_AR1 a diferença foi de apenas 0.09%, ou seja, 1.064.

Comparou-se a taxa de convergência entre os algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS, DSTr-LMS e Dr_0-LMS, verificou-se que para a curva CA_AR1 o algoritmo DSTr-LMS foi o mais rápido, pois obteve o I_3 com o menor limite inferior, ou seja, I_3=134:234. O algoritmo Dr_0-LMS foi mais lento que o D-LMS, pois o primeiro obteve o I_3 com menor limite inferior, ou seja, I_3=2099:2199 e para o segundo foi de I_3=995:1095. No entanto, o Dr_0-LMS foi mais rápido que o DST-LMS e o NLMS.

Para a curva CA_AR2, o algoritmo DSTr-LMS foi o mais rápido, o Dr_0-LMS foi mais lento que o D-LMS pois registrou o I_3 com menor limite inferior, ou seja, I_3=796:896 e para o segundo foi de I_3=457:557. No entanto, o Dr_0-LMS foi um pouco mais lento que o DST-LMS e bem mais rápido que o NLMS.

Para a curva CA_AR3, o algoritmo DSTr-LMS foi o mais rápido, o Dr_0-LMS foi um pouco mais lento que o D-LMS pois registrou o I_3 com menor limite inferior, ou seja, I_3=438:538 e para o segundo foi de I_3=425:525. No entanto, o Dr_0-LMS foi bem mais rápido que o NLMS.

O menor erro de regime permanente foi obtido pelo DCT-LMS para CA_AR1, ou seja, $\hat{J}(\infty)$=1.057, em seguida veio o Dr_0-LMS com uma diferença de 0.7% e o D-LMS com uma diferença de 1.8%, ou seja, $\hat{J}(\infty)$=1.076. Para a curva CA_AR2, o menor erro foi obtido pelo DCT-LMS com $\hat{J}(\infty)$=1.059 e para a curva CA_AR3 o menor erro foi obtido pelo NLMS com $\hat{J}(\infty)$=1.030.

6.4.2 <u>Simulação-13</u>: Análise do Erro de Regime Permanente para os Algoritmos Dr_i-LMS e Dr_{ij}-LMS, Quando Submetidos a Predição Linear dos Processos AR1, AR2 e AR3

Essa análise possui os seguintes objetivos:

- Verificar a variação do erro de regime permanente para os algoritmos Dr_i-LMS e Dr_{ij}-LMS em relação ao erro do algoritmo D-LMS, ou seja, com a eliminação de uma ou mais entradas $r_i(n)$ será observado, em diversas simulações, uma diminuição do $\hat{J}(\infty)$.
- Determinar entre os algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS, DSTr-LMS, Dr_i-LMS e Dr_{ij}-LMS, qual deles obteve o menor erro de regime permanente, em relação a um dado processo.

Observando-se as Tabelas 4, 5 e 6 e as Figs.(6.13.a) , (6.13.b) , (6.14.a) , (6.14.b) , (6.15.a), (6.15.b), (6.16.a), (6.16.b), (6.17.a), (6.17.b), (6.18.a), (6.18.b), (6.19.a), (6.19.b), (6.20.a), (6.20.b), (6.21.a), (6.21.b), (6.22.a) e (6.22.b), verifica-se que a eliminação de algumas entradas, de forma geral, levou a uma diminuição do erro de regime permanente em relação ao D-LMS, pois para a curva CA_AR1 o valor desse parâmetro obtido pelo D-LMS foi $\hat{J}(\infty)$=1.076, para o Dr_0-LMS obteve-se $\hat{J}(\infty)$=1.064, para o Dr_1-LMS obteve-se $\hat{J}(\infty)$=1.065, para o Dr_2-LMS obteve-se $\hat{J}(\infty)$=1.065, para o Dr_3-LMS obteve-se $\hat{J}(\infty)$=1.064, para o Dr_{01}-LMS obteve-se $\hat{J}(\infty)$=1.063, para o Dr_{02}-LMS obteve-se $\hat{J}(\infty)$=1.053, etc. Ainda para essa curva, o menor erro de regime permanente entre todos os algoritmos foi $\hat{J}(\infty)$=1.053, obtido pelos algoritmos Dr_{02}-LMS e Dr_{13}-LMS, vindo em seguida o algoritmo DCT-LMS com um valor muito próximo, isto é, $\hat{J}(\infty)$=1.057.

Para a curva CA_AR2 também constatou-se que a eliminação de algumas entradas proporcionou uma diminuição do erro de regime permanente em relação ao D-LMS, pois o valor desse parâmetro obtido pelo D-LMS foi $\hat{J}(\infty)$=1.085, para o Dr_0-LMS obteve-se $\hat{J}(\infty)$=1.072, para o Dr_1-LMS obteve-se $\hat{J}(\infty)$=1.072, para o Dr_2-LMS obteve-se $\hat{J}(\infty)$=1.072, para o Dr_3-LMS obteve-se $\hat{J}(\infty)$=1.072, para o Dr_{01}-LMS obteve-se $\hat{J}(\infty)$=1.057, para o Dr_{02}-LMS obteve-se $\hat{J}(\infty)$=1.059, etc. Ainda para essa curva, o menor erro de regime permanente entre todos os algoritmos foi $\hat{J}(\infty)$=1.057 obtido pelo algoritmo Dr_{01}-LMS. Próximo desse valor têm-se ainda os seguintes resultados: $\hat{J}(\infty)$=1.058 obtido pelo Dr_{03}-LMS, $\hat{J}(\infty)$=1.059 obtido pelos algoritmos Dr_{02}-LMS, Dr_{12}-LMS, Dr_{13}-LMS e DCT-LMS e para o Dr_{23}-LMS obteve-se $\hat{J}(\infty)$=1.060.

Essa diminuição do erro de regime permanente é constatada também para a curva CA_AR3, onde o menor valor $\hat{J}(\infty)$=1.030 foi obtido pelo NLMS, acompanhado pelos seguintes resultados: $\hat{J}(\infty)$=1.050 obtido pelos algoritmos Dr_{03}-LMS, Dr_{12}-LMS, Dr_{13}-LMS, $\hat{J}(\infty)$=1.051 obtido pelos algoritmos Dr_{02}-LMS e DST-LMS, $\hat{J}(\infty)$=1.052 obtido pelos algoritmos Dr_{01}-LMS e DCT-LMS, $\hat{J}(\infty)$=1.053 obtido pelo Dr_{23}-LMS, etc.

6.4.3 Simulação-14: Análise da Taxa de Convergência para os Algoritmos Dr_i-LMS e Dr_{ij}-LMS, Quando Submetidos a Predição Linear dos Processos AR1, AR2 e AR3

Essa análise possui o seguinte objetivo:

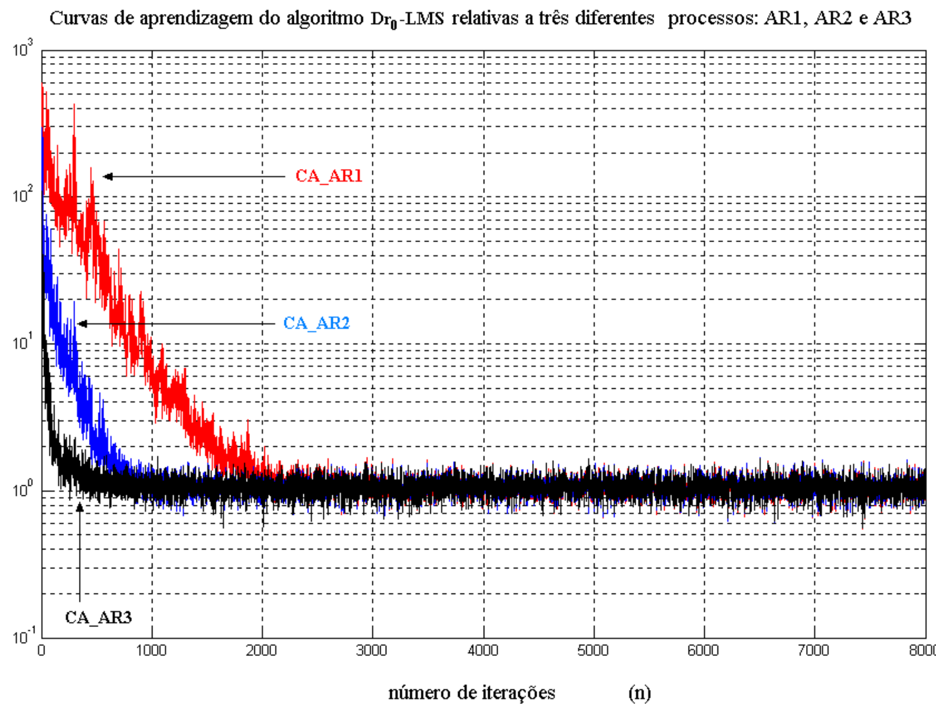
- Verificar a variação da taxa de convergência para os algoritmos Dr_i-LMS e Dr_{ij}-LMS em relação a taxa de convergência D-LMS, ou seja, com a eliminação de uma ou mais entradas $r_i(n)$ será observado, em diversas simulações, uma diminuição ou aumento da taxa de convergência.

Com relação ao aspecto taxa de convergência, analisou-se as Tabelas 4, 5 e 6 e as Figs.(6.13.a) , (6.13.b) , (6.14.a) , (6.14.b) , (6.15.a), (6.15.b), (6.16.a), (6.16.b), (6.17.a), (6.17.b), (6.18.a), (6.18.b), (6.19.a), (6.19.b), (6.20.a), (6.20.b), (6.21.a), (6.21.b), (6.22.a) e (6.22.b), verificou-se que em certos casos, a eliminação de algumas entradas conduziu a uma pequena variação entre a taxa de convergência obtida por esse procedimento e a obtida através do algoritmo D-LMS, com a vantagem de gerar uma diminuição do erro de regime permanente em relação ao D-LMS, conforme visto na simulação anterior.

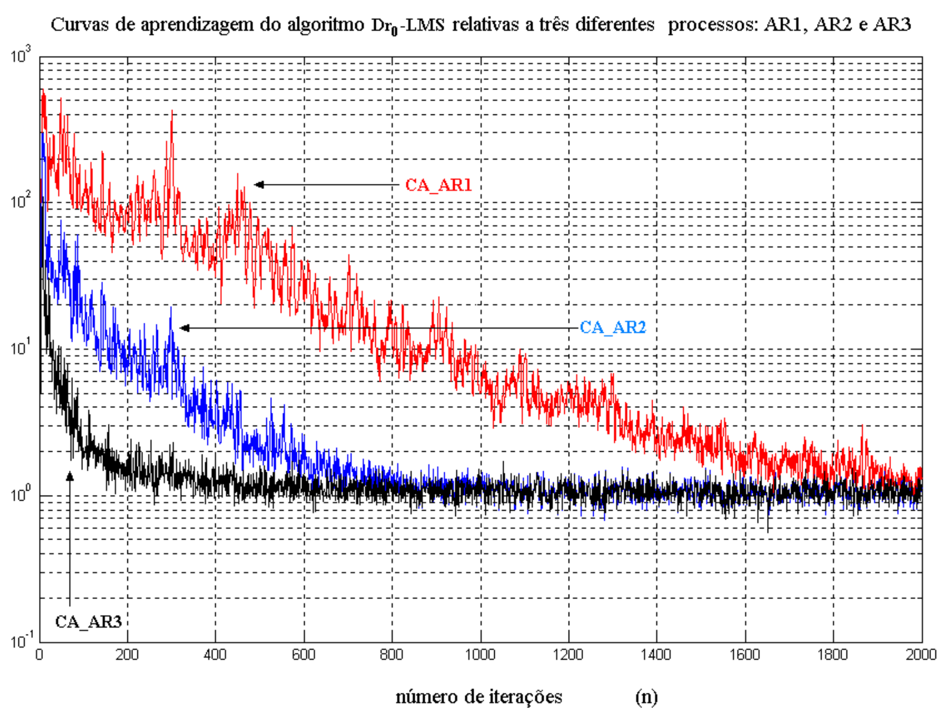
Para a curva CA_AR1, obteve-se I_3=995:1095 para o D-LMS, obteve-se I_3=1073:1173 para o Dr_1-LMS, obteve-se I_3=1122:1222 para o Dr_2-LMS e I_3=1177:1277 para o Dr_{12}-LMS.

Para a curva CA_AR2, obteve-se I_3=457:557 para o D-LMS, obteve-se I_3=507:607 para o Dr_1-LMS, obteve-se I_3=526:626 para o Dr_2-LMS e I_3=528:628 para o Dr_{12}-LMS.

Para a curva CA_AR3, obteve-se I_3=425:525 para o D-LMS, obteve-se I_3=419:519 para o Dr_2-LMS, obteve-se I_3=422:522 para o Dr_{12}-LMS, obteve-se I_3=526:626 para o Dr_2-LMS, obteve-se I_3=429:529 para o Dr_3-LMS, etc.

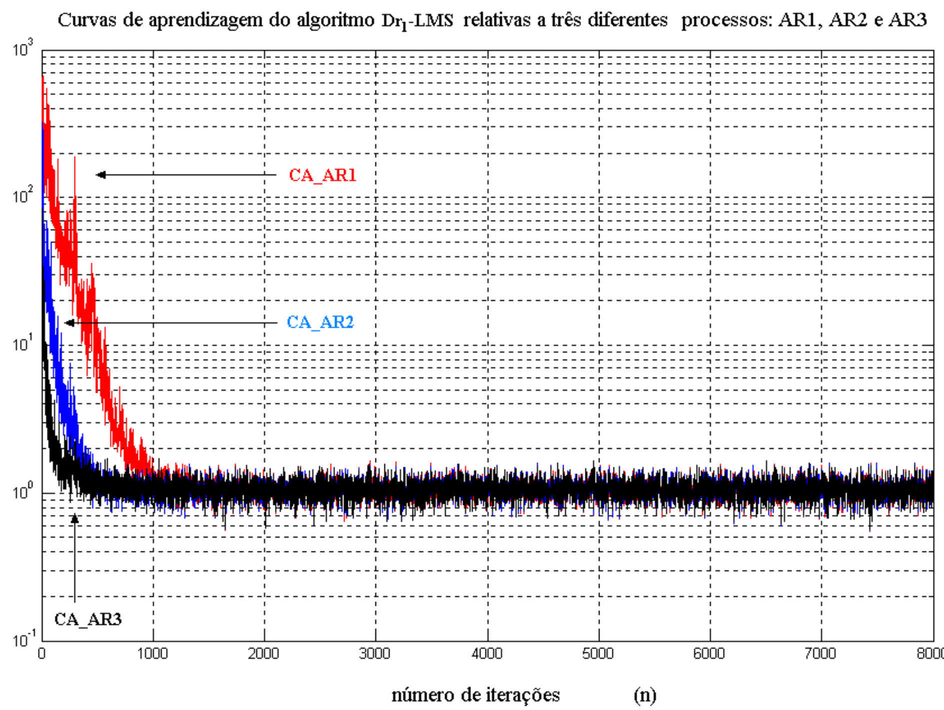


a)

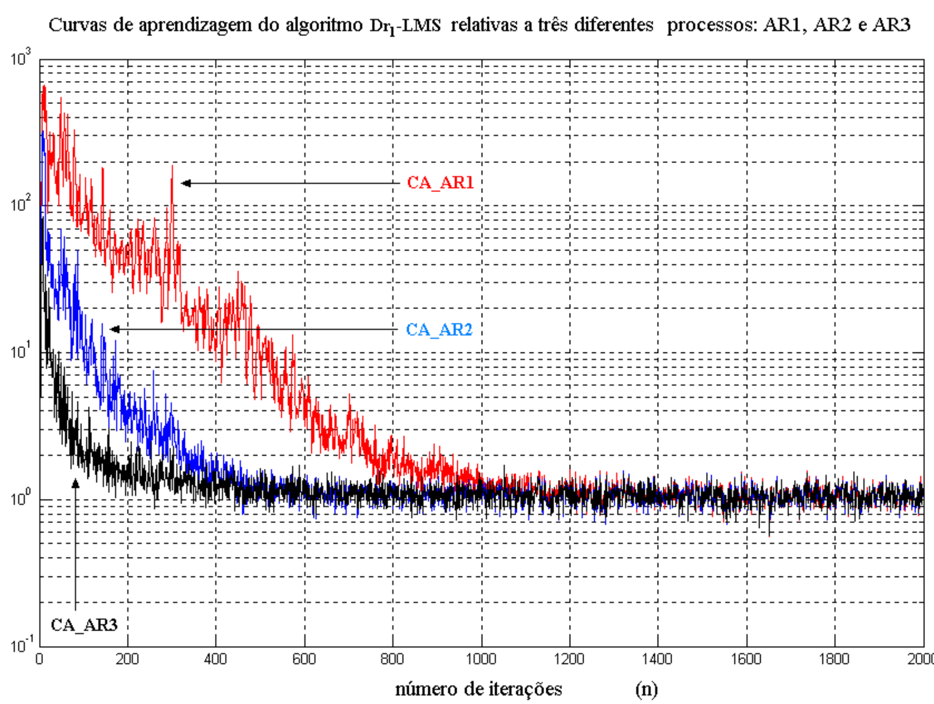


b)

Fig.6.13. a)Curvas de aprendizagem para o algoritmo Dr_0-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

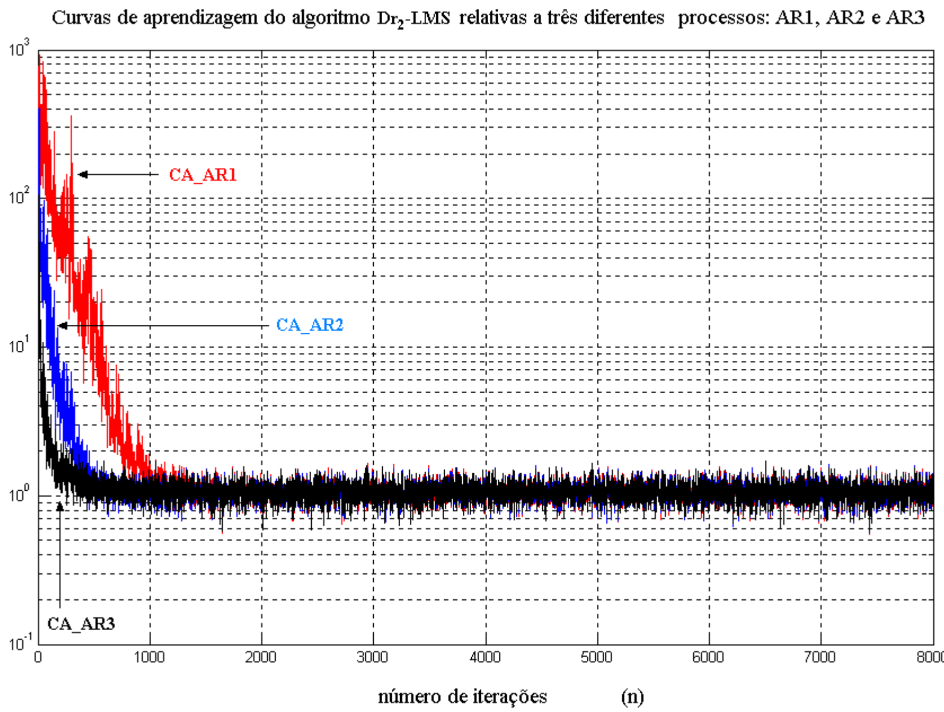


a)

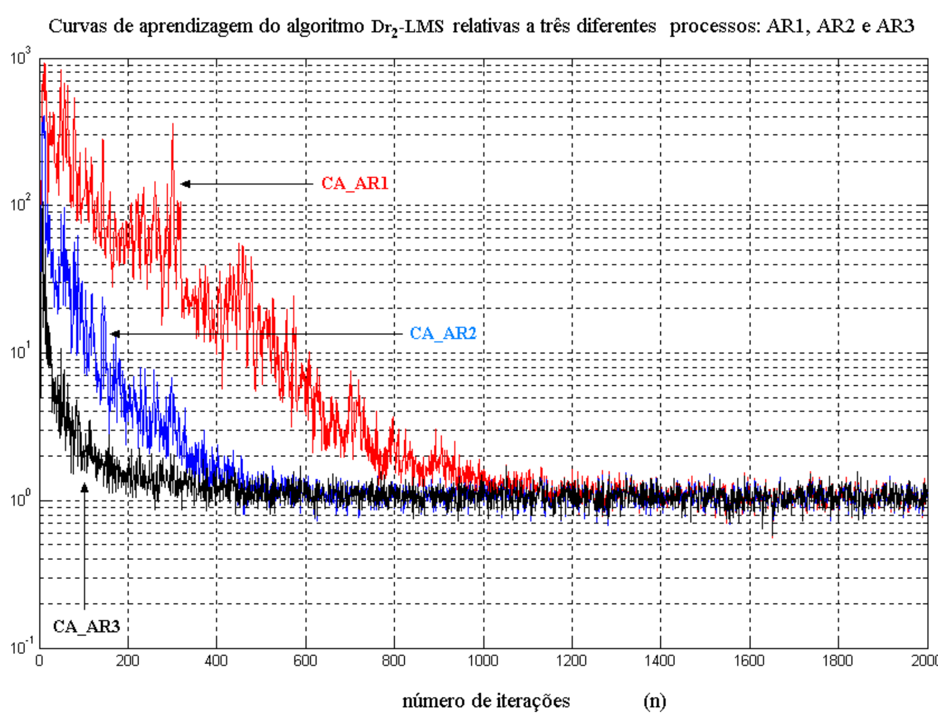


b)

Fig.6.14. a)Curvas de aprendizagem para o algoritmo Dr_1-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

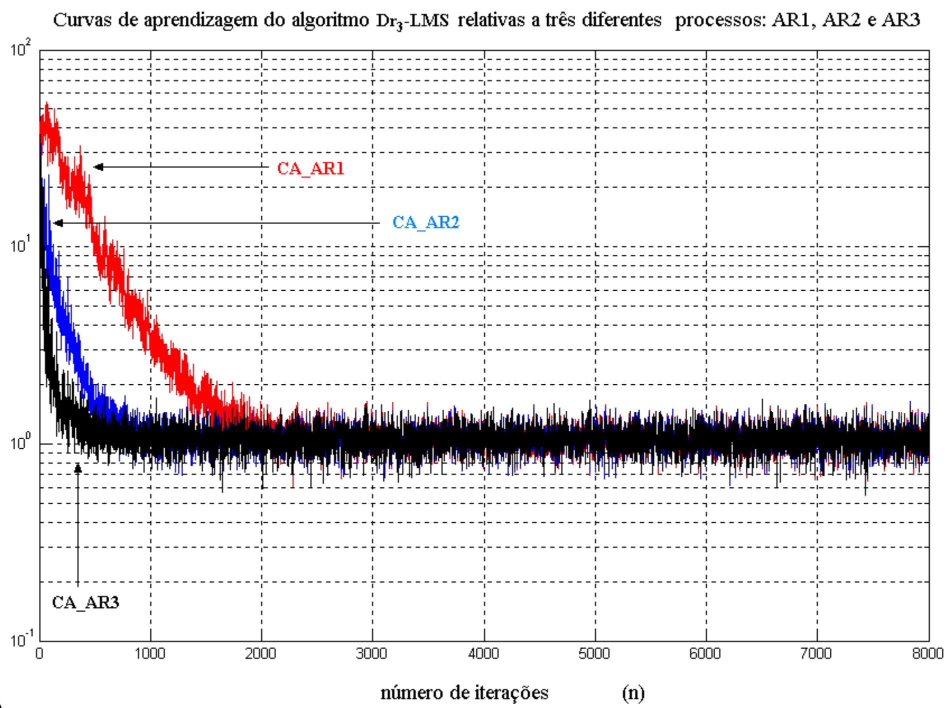


a)

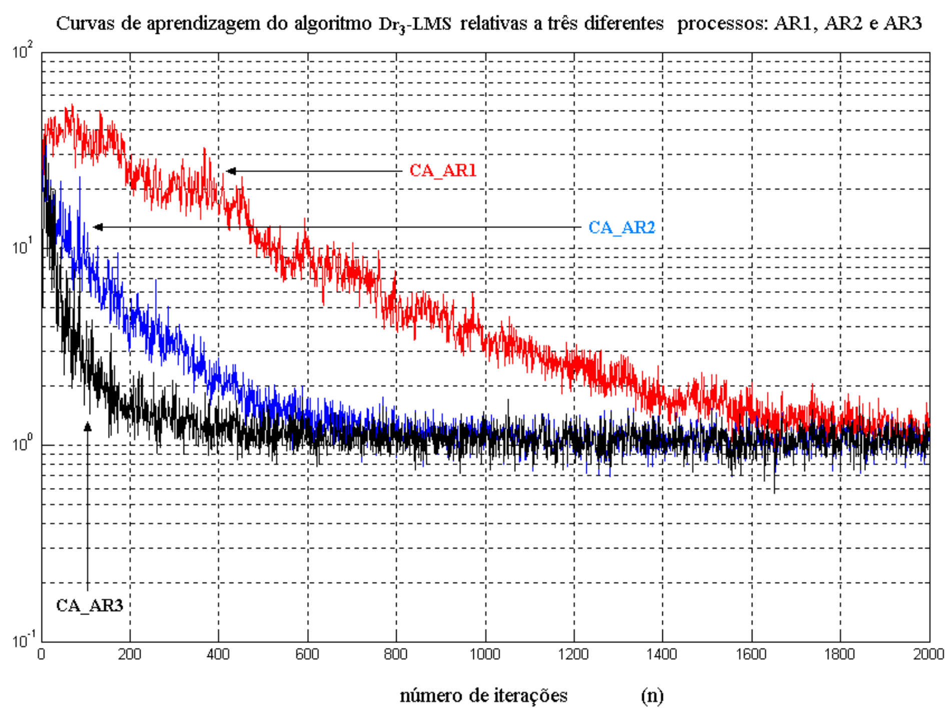


b)

Fig.6.15. a)Curvas de aprendizagem para o algoritmo Dr_2-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

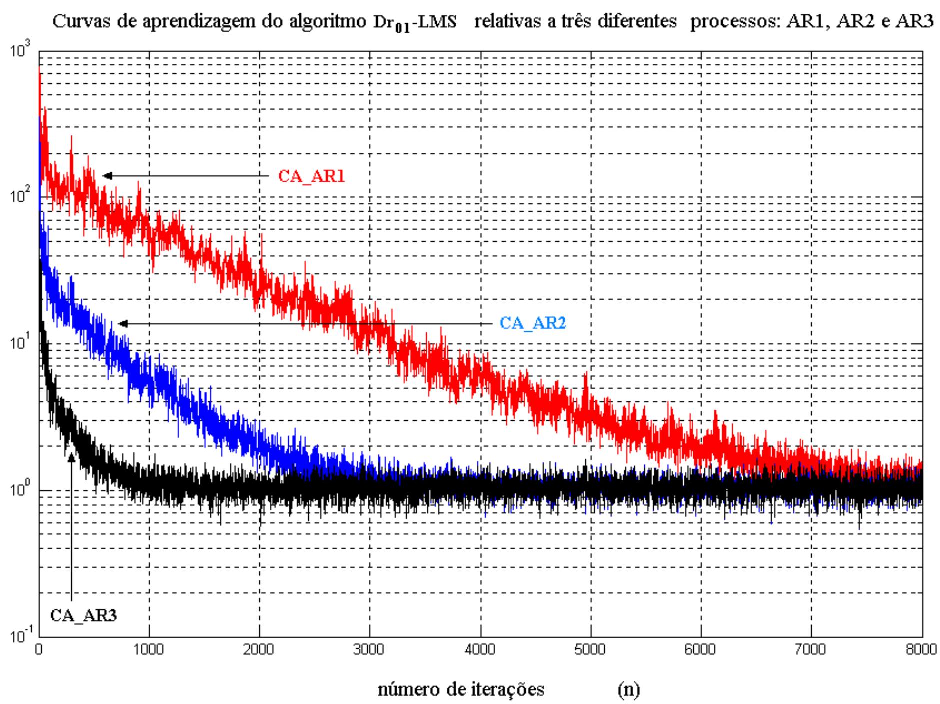


a)

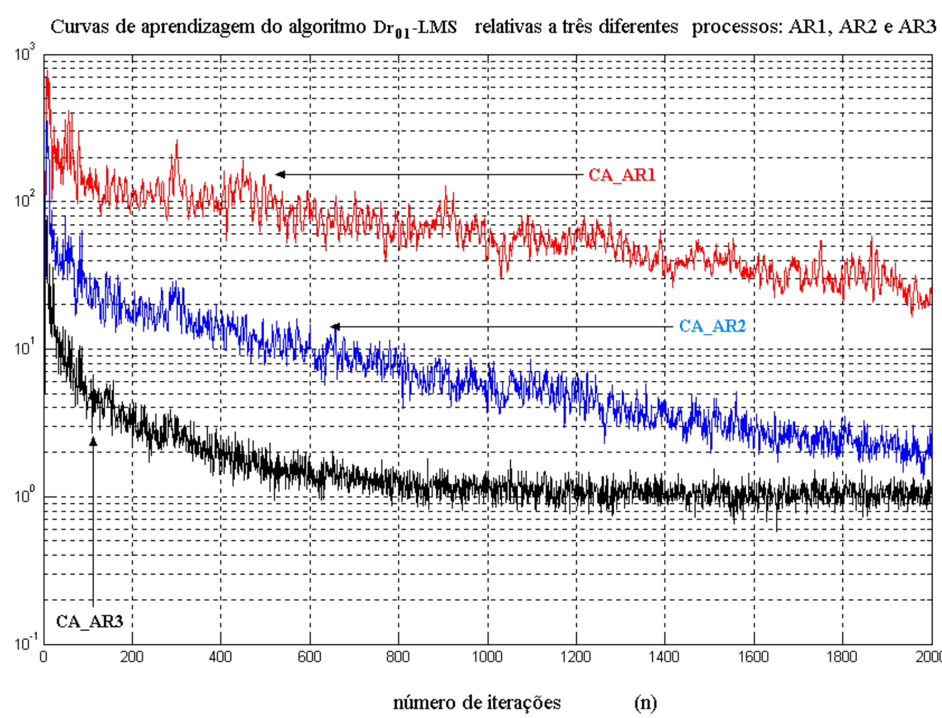


b)

Fig.6.16. a)Curvas de aprendizagem para o algoritmo Dr_3-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

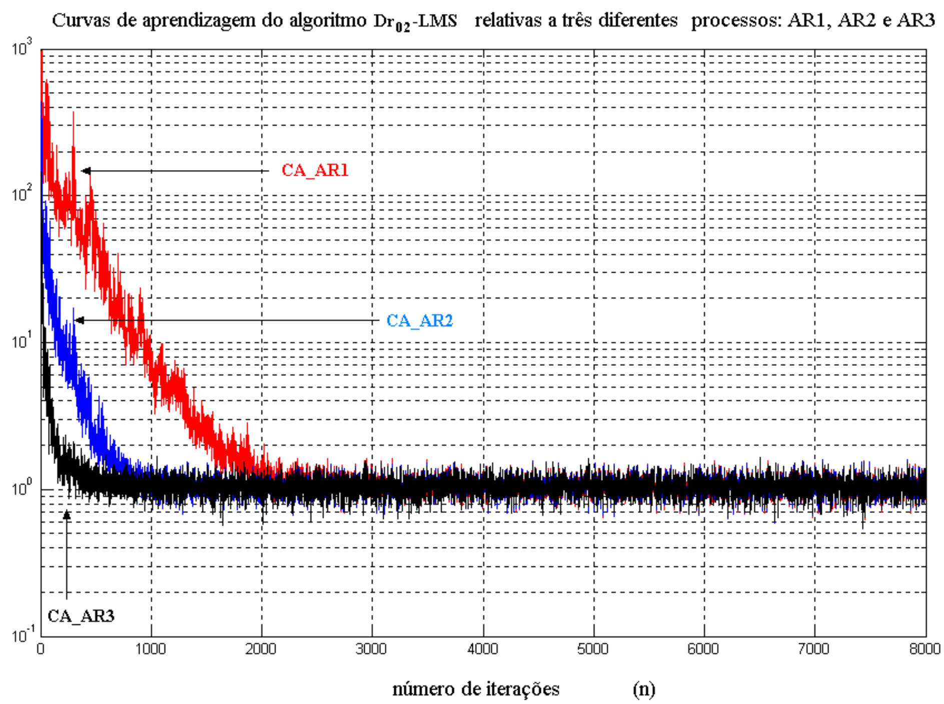


a)

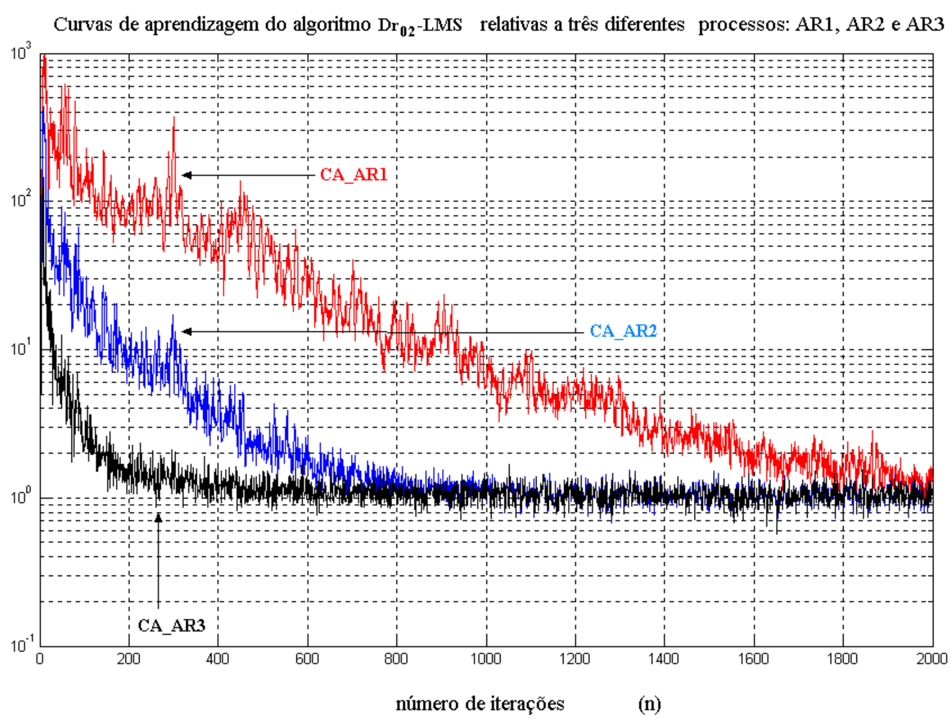


b)

Fig.6.17. a)Curvas de aprendizagem para o algoritmo Dr_{01}-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

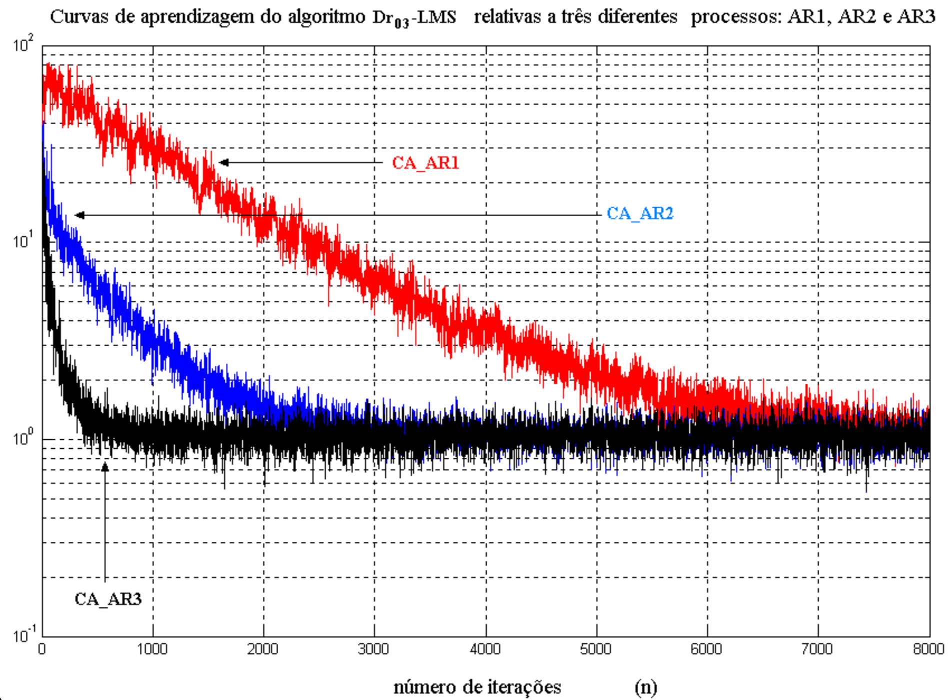


a)

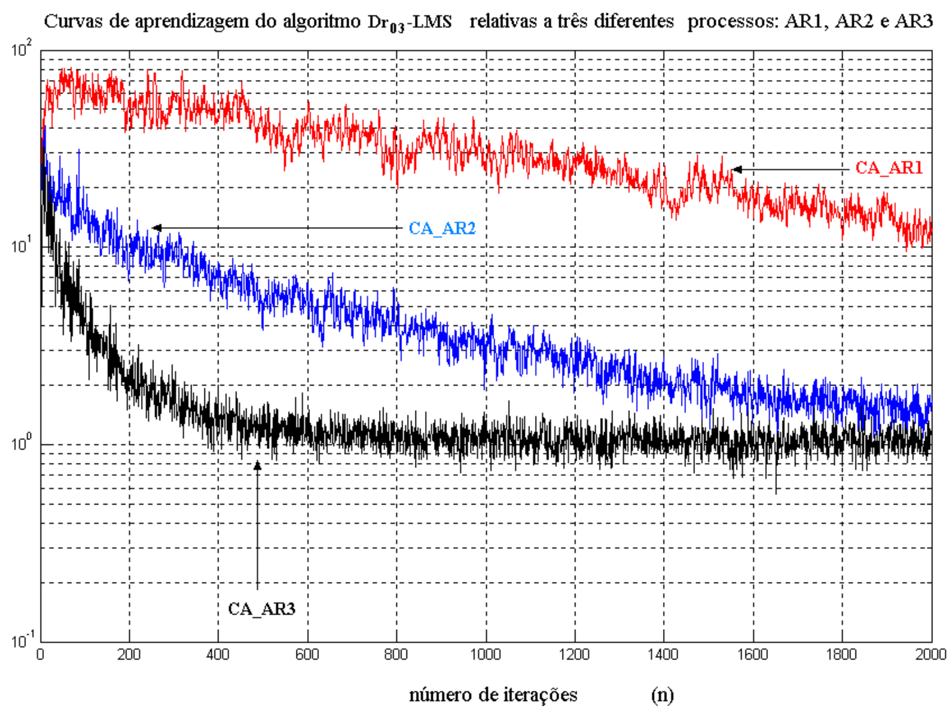


b)

Fig.6.18. a)Curvas de aprendizagem para o algoritmo Dr_{02}-LMS usado em predição linear, considerando-se que N=4; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

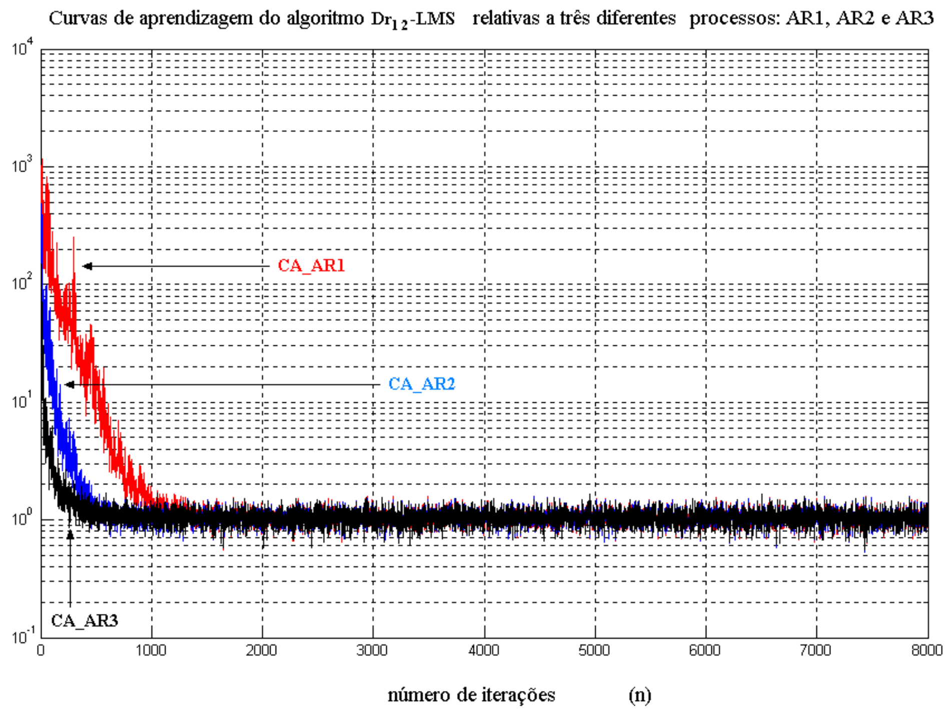


a)

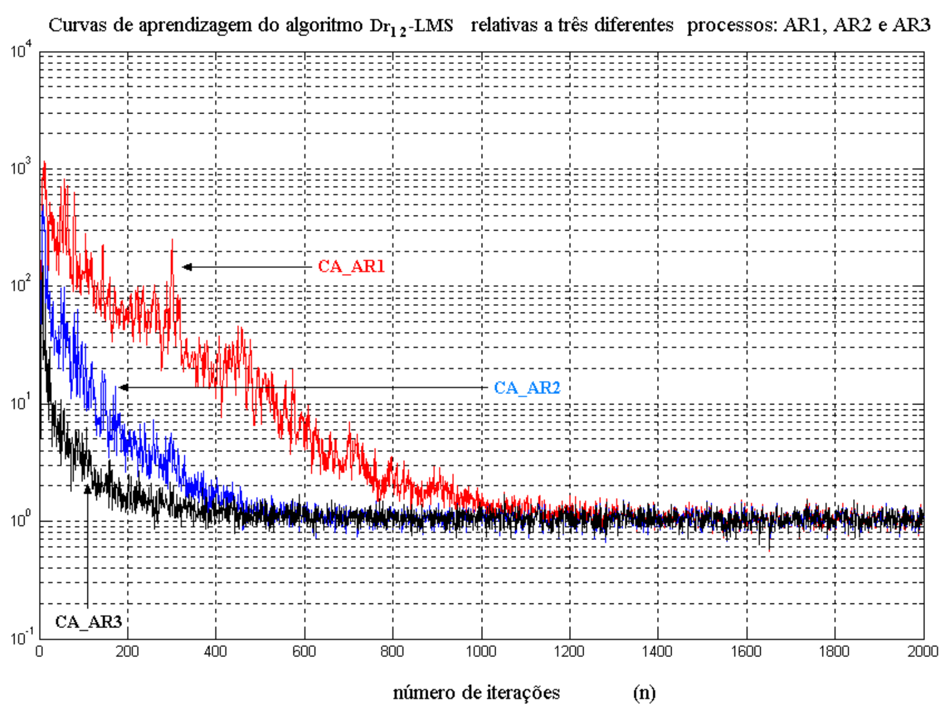


b)

Fig.6.19. a)Curvas de aprendizagem para o algoritmo Dr_{03}-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

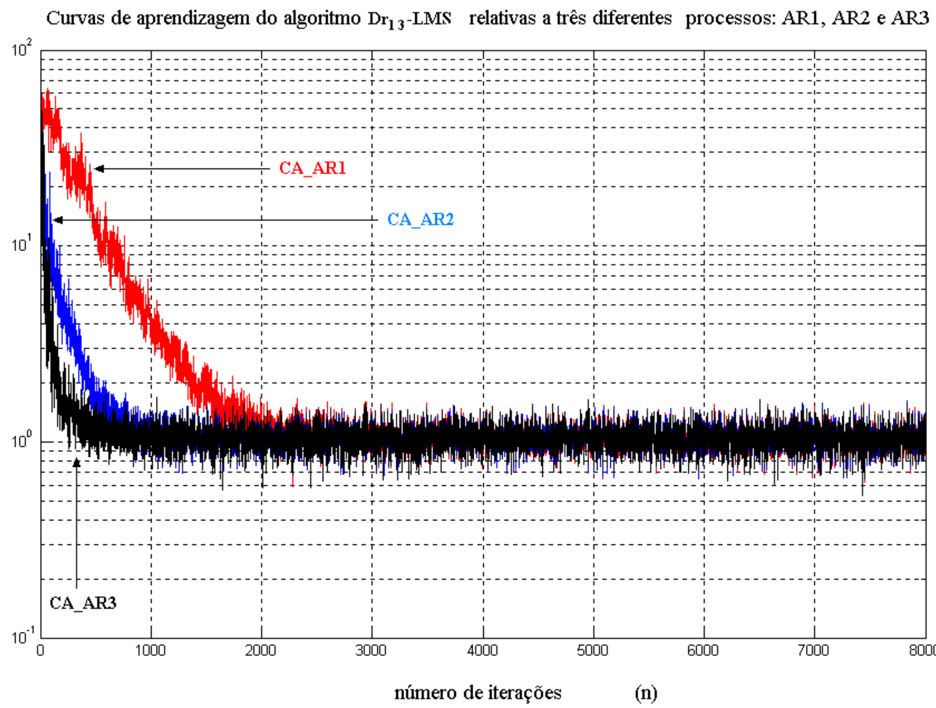


a)

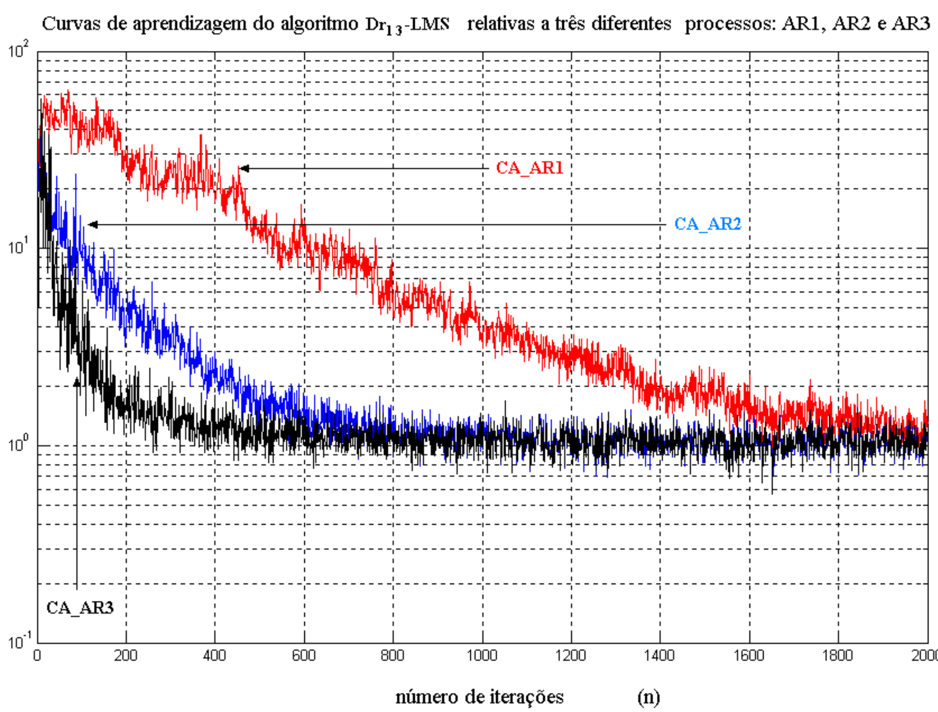


b)

Fig.6.20. a)Curvas de aprendizagem para o algoritmo Dr_{12}-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

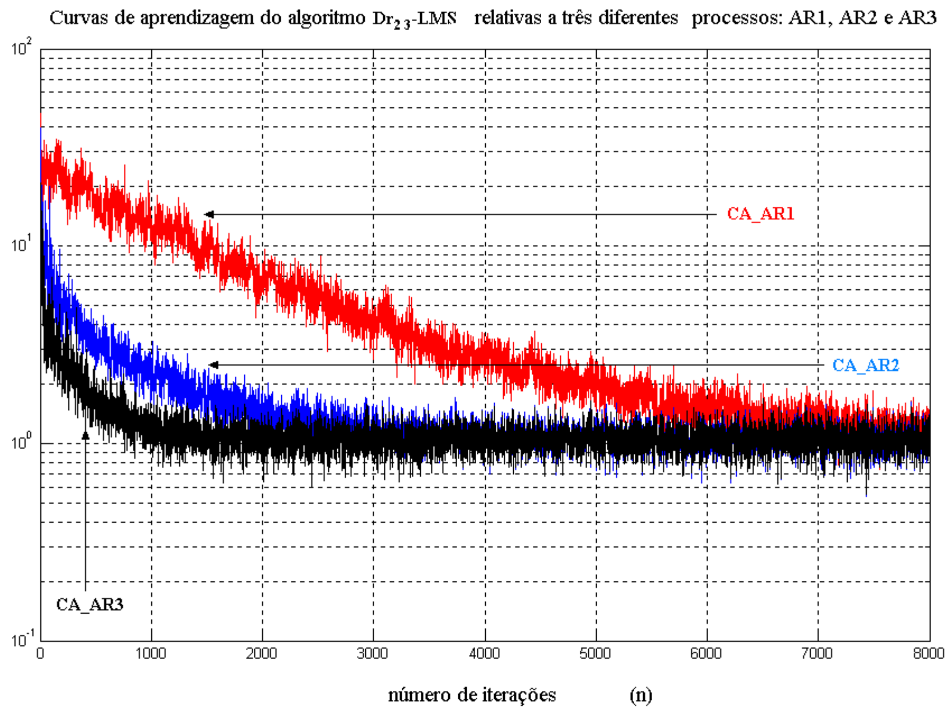


a)

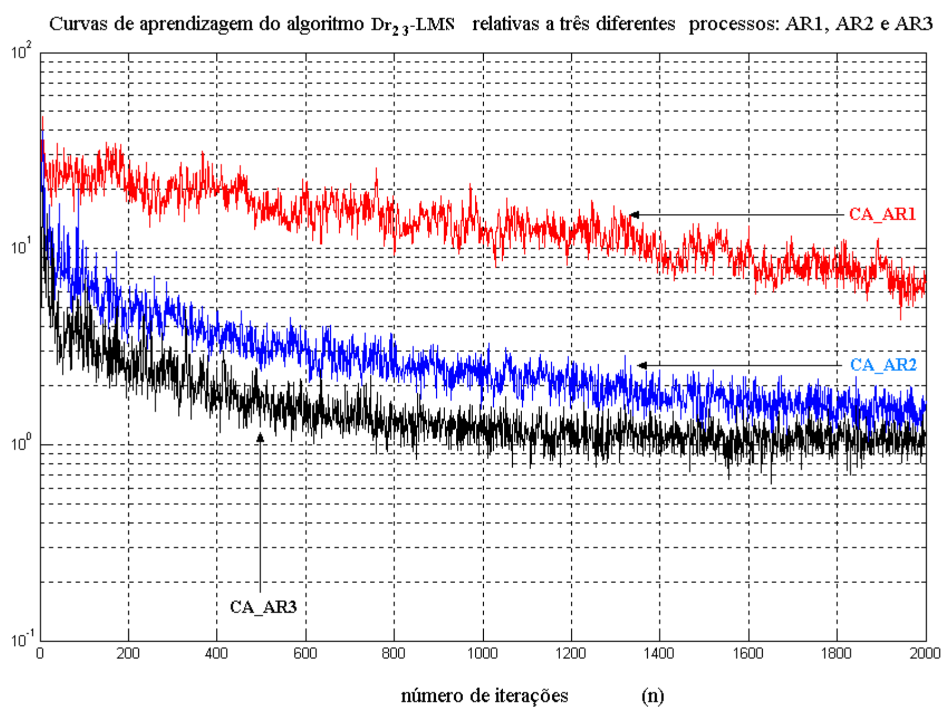


b)

Fig.6.21. a)Curvas de aprendizagem para o algoritmo Dr_{13}-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.



a)



b)

Fig.6.22. a)Curvas de aprendizagem para o algoritmo Dr_{23}-LMS usado em predição linear, considerando-se que N=4 ; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

	Processo AR1 ; $\mu = 0.02$, $\beta = 0.90$ e $\delta = 10^{-4}$ (Dados obtidos da Curva CA_AR1, relacionada ao processo AR1)			
	Valor de pico	Valores Médios $\hat{J}(I_3) \leq 1.1 \times \hat{J}(\infty)$ (Intervalo I_3 é variável)	Erro de regime permanente $\hat{J}(\infty)$ $\hat{J}(\infty) = \hat{J}(I_1) \cong \hat{J}(I_2)$ ou $\hat{J}(\infty) = \hat{J}(I_{1e}) \cong \hat{J}(I_{2e})$ $I_1 = 7800{:}7999$; $I_2 = 7000{:}7199$ $I_{1e} = 11800{:}11999$; $I_{2e} = 11000{:}11199$	$D = \hat{J}(I_2)/\hat{J}(\infty)$; $\%\Delta = (D-1)\times 100$
Dr_0-LMS	$\hat{J}(8) = 602$	$\hat{J}(2099{:}2199) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.064$; $\hat{J}(I_2) = 1.068$	0.38%
Dr_1-LMS	$\hat{J}(12) = 663$	$\hat{J}(1073{:}1173) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.065$; $\hat{J}(I_2) = 1.068$	0.28%
Dr_2-LMS	$\hat{J}(12) = 930$	$\hat{J}(1122{:}1222) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.065$; $\hat{J}(I_2) = 1.068$	0.28%
Dr_3-LMS	$\hat{J}(69) = 54$	$\hat{J}(1930{:}2030) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.064$; $\hat{J}(I_2) = 1.068$	0.38%
Dr_{01}LMS	$\hat{J}(8) = 781$	-	$\hat{J}(I_1) = 1.234$; $\hat{J}(I_2) = 1.388$	12.5%
		$\hat{J}(8114{:}8214) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_{1e}) = 1.063$; $\hat{J}(I_{2e}) = 1.082$	1.79%
Dr_{02}-LMS	$\hat{J}(12) = 976$	$\hat{J}(2168{:}2268) = 1.15$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.053$; $\hat{J}(I_2) = 1.056$	0.28%
Dr_{03}-LMS	$\hat{J}(69) = 82$	-	$\hat{J}(I_1) = 1.155$; $\hat{J}(I_2) = 1.251$	8.30%
		$\hat{J}(7290{:}7390) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_{1e}) = 1.063$; $\hat{J}(I_{2e}) = 1.068$	0.47%
Dr_{12}-LMS	$\hat{J}(12) = 1175$	$\hat{J}(1177{:}1277) = 1.16$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.054$; $\hat{J}(I_2) = 1.056$	0.19%
Dr_{13}-LMS	$\hat{J}(69) = 64$	$\hat{J}(2066{:}2166) = 1.16$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.053$; $\hat{J}(I_2) = 1.057$	0.38%
Dr_{23}-LMS	$\hat{J}(6) = 47$	-	$\hat{J}(I_1) = 1.197$; $\hat{J}(I_2) = 1.289$	7.70%
		$\hat{J}(7370{:}7470) = 1.18$	$\hat{J}(\infty) = \hat{J}(I_{1e}) = 1.072$; $\hat{J}(I_{2e}) = 1.092$	1.87%

Tabela 4. Apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_{1e})$, $\hat{J}(I_{2e})$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e $\%\Delta$ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos Dr_0-LMS, Dr_1-LMS, Dr_2-LMS, Dr_3-LMS, Dr_{01}LMS, Dr_{02}-LMS, Dr_{03}-LMS, Dr_{12}-LMS, Dr_{13}-LMS e Dr_{23}-LMS, sujeitos aos parâmetros $\mu = 0.02$, $\beta = 0.90,$ $\delta = 10^{-4}$ e processo AR1.

	Processo AR2 ; $\mu = 0.02$, $\beta = 0.90$ e $\delta = 10^{-4}$ (Dados obtidos da Curva CA_AR2, relacionada ao processo AR2)			
	Valor de pico	Valores Médios $\hat{J}(I_3) \leq 1.1 \times \hat{J}(\infty)$ (Intervalo I_3 é variável)	Erro de regime permanente $\hat{J}(\infty)$ $\hat{J}(\infty) = \hat{J}(I_1) \cong \hat{J}(I_2)$ $I_1 = 7800{:}7999$; $I_2 = 7000{:}7199$	D = $\hat{J}(I_2)/\hat{J}(\infty)$; $\%\Delta = (D\text{-}1)\times 100$
Dr_0-LMS	$\hat{J}(8) = 296$	$\hat{J}(796{:}896) = 1.18$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.072$; $\hat{J}(I_2) = 1.070$	-0.19 %
Dr_1-LMS	$\hat{J}(8) = 325$	$\hat{J}(507{:}607) = 1.18$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.072$; $\hat{J}(I_2) = 1.070$	-0.19 %
Dr_2-LMS	$\hat{J}(12) = 407$	$\hat{J}(526{:}626) = 1.18$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.072$; $\hat{J}(I_2) = 1.070$	-0.19 %
Dr_3-LMS	$\hat{J}(11) = 34$	$\hat{J}(747{:}847) = 1.18$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.072$; $\hat{J}(I_2) = 1.070$	-0.19 %
Dr_{01}LMS	$\hat{J}(8) = 354$	$\hat{J}(3139{:}3239) = 1.16$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.057$; $\hat{J}(I_2) = 1.056$	-0.09 %
Dr_{02}-LMS	$\hat{J}(8) = 438$	$\hat{J}(814{:}914) = 1.16$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.059$; $\hat{J}(I_2) = 1.057$	-0.19 %
Dr_{03}-LMS	$\hat{J}(11) = 41$	$\hat{J}(2699{:}2799) = 1.16$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.058$; $\hat{J}(I_2) = 1.056$	-0.19 %
Dr_{12}-LMS	$\hat{J}(8) = 483$	$\hat{J}(528{:}628) = 1.16$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.059$; $\hat{J}(I_2) = 1.056$	-0.28 %
Dr_{13}-LMS	$\hat{J}(7) = 37$	$\hat{J}(794{:}894) = 1.16$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.059$; $\hat{J}(I_2) = 1.057$	-0.19 %
Dr_{23}-LMS	$\hat{J}(7) = 40$	$\hat{J}(3090{:}3190) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.060$; $\hat{J}(I_2) = 1.059$	-0.09 %

Tabela 5. Apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e %Δ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos Dr_0-LMS, Dr_1-LMS, Dr_2-LMS, Dr_3-LMS, Dr_{01}LMS, Dr_{02}-LMS, Dr_{03}-LMS, Dr_{12}-LMS, Dr_{13}-LMS e Dr_{23}-LMS, sujeitos aos parâmetros $\mu = 0.02$, $\beta = 0.90$, $\delta = 10^{-4}$ e processo AR2.

	Processo AR3 ; $\mu = 0.02$, $\beta = 0.90$ e $\delta = 10^{-4}$ (Dados obtidos da Curva CA_AR3, relacionada ao processo AR3)			
	Valor de pico	Valores Médios $\hat{J}(I_3) \leq 1.1 \times \hat{J}(\infty)$ (Intervalo I_3 é variável)	Erro de regime permanente $\hat{J}(\infty)$ $\hat{J}(\infty) = \hat{J}(I_1) \cong \hat{J}(I_2)$ $I_1 = 7800{:}7999$; $I_2 = 7000{:}7199$	$D = \hat{J}(I_2)/\hat{J}(\infty)$; $\%\Delta = (D-1)\times 100$
Dr_0-LMS	$\hat{J}(8) = 109$	$\hat{J}(438{:}538) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.063$; $\hat{J}(I_2) = 1.069$	0.56 %
Dr_1-LMS	$\hat{J}(8) = 84$	$\hat{J}(437{:}537) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.062$; $\hat{J}(I_2) = 1.068$	0.56 %
Dr_2-LMS	$\hat{J}(8) = 106$	$\hat{J}(419{:}519) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.064$; $\hat{J}(I_2) = 1.068$	0.38 %
Dr_3-LMS	$\hat{J}(10) = 37$	$\hat{J}(429{:}529) = 1.17$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.063$; $\hat{J}(I_2) = 1.067$	0.38 %
Dr_{01}LMS	$\hat{J}(8) = 80$	$\hat{J}(911{:}1011) = 1.15$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.052$; $\hat{J}(I_2) = 1.056$	0.38 %
Dr_{02}-LMS	$\hat{J}(8) = 146$	$\hat{J}(423{:}523) = 1.16$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.051$; $\hat{J}(I_2) = 1.055$	0.38 %
Dr_{03}-LMS	$\hat{J}(10) = 32$	$\hat{J}(566{:}666) = 1.15$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.050$; $\hat{J}(I_2) = 1.054$	0.38 %
Dr_{12}-LMS	$\hat{J}(8) = 151$	$\hat{J}(422{:}522) = 1.15$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.050$; $\hat{J}(I_2) = 1.054$	0.38%
Dr_{13}-LMS	$\hat{J}(10) = 57$	$\hat{J}(442{:}542) = 1.15$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.050$; $\hat{J}(I_2) = 1.054$	0.38%
Dr_{23}-LMS	$\hat{J}(7) = 21$	$\hat{J}(1053{:}1153) = 1.15$	$\hat{J}(I_1) = 1.053$; $\hat{J}(I_2) = 1.055$	0.19%

Tabela 6. Apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e $\%\Delta$ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos Dr_0-LMS, Dr_1-LMS, Dr_2-LMS, Dr_3-LMS, Dr_{01}LMS, Dr_{02}-LMS, Dr_{03}-LMS, Dr_{12}-LMS, Dr_{13}-LMS e Dr_{23}-LMS, sujeitos aos parâmetros $\mu = 0.02$, $\beta = 0.90$, $\delta = 10^{-4}$ e processo AR3.

6.5 Grupo de Simulações que Analisa o Desempenho dos Algoritmos $DSTr_i$-LMS e $DSTr_{ij}$-LMS

Denomina-se $DSTr_0$-LMS ao algoritmo DSTr-LMS que exclui a entrada decomposta $r_{DST,0}(n)$, isto é, não a utiliza. Essa entrada é uma das componentes do vetor $\underline{\mathbf{u}}_{DST}(n)$ definido pela Eq.(5.46.c), o qual está inserido na expressão de atualização dos pesos denotada pela Eq.(5.48). Caso duas entradas sejam excluídas, por exemplo $r_{DST,0}(n)$ e $r_{DST,1}(n)$, então, o algoritmo denominar-se-á de $DSTr_{0\,1}$-LMS.

Cada simulação desse grupo objetiva analisar o desempenho dos algoritmos $DSTr_i$-LMS e $DSTr_{ij}$-LMS com $0 \leq i, j \leq N-1$ (vide Tabelas 7, 8 e 9), inclusive comparando-o com o dos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS, DSTr-LMS e Dr_0-LMS.

Esses algoritmos são empregados no esquema de predição linear dado pela Fig.(6.1) com Δ=1, quando submetidos aos três processos autoregressivos de entrada, com diferentes valores para o número de condição (*nc*). Tanto a análise individual quanto a comparação entre os algoritmos são baseadas nos mesmos critérios de desempenho relatados anteriormente.

Para cada variação do algoritmo DSTr-LMS, isto é, $DSTr_i$-LMS ou $DSTr_{i\,j}$-LMS, definem-se as curvas de aprendizagem desses algoritmos relacionadas ao processo AR1 de CA_AR1, ao processo AR2 de CA_AR2 e ao processo AR3 de CA_AR3. As Tabelas 7, 8 e 9, que serão detalhadas a seguir, apresentam diversos valores extraídos dessas curvas de aprendizagem. A Tabela 7 apresenta os resultados relacionados a curva CA_AR1, a Tabela 8 refere-se a curva CA_AR2 e a Tabela 9 refere-se a curva CA_AR3.

Atribuiu-se aos parâmetros das Eqs.(5.48), (5.49) e (5.50), relacionadas ao algoritmo DSTr-LMS e suas variações, os valores $N=4$; $\mu=0.02$; $\beta=0.90$ e $\delta=10^{-4}$, necessários para geração das curvas de aprendizagem e tabelas mencionadas. A seguir apresentam-se os resultados obtidos de algumas simulações.

6.5.1 <u>Simulação-15</u>: Análise do Desempenho do Algoritmo $DSTr_0$-LMS Quando Submetido a Predição Linear dos Processos AR1, AR2 e AR3

Essa análise possui os seguintes objetivos:

- Verificar a variação do erro de regime permanente em função da variação do número de condição (*nc*) e comparar este resultado com os obtidos pelos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS, DSTr-LMS e Dr_0-LMS.
- Avaliar a taxa de convergência do algoritmo $DSTr_0$-LMS em função da variação do *nc* e comparar este resultado com os obtidos pelos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS, DSTr-LMS e Dr_0-LMS.
- Determinar entre os algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS, DSTr-LMS, Dr_0-LMS e $DSTr_0$-LMS, qual deles obteve o menor erro de regime permanente e a melhor taxa de convergência, para cada um dos processos.

Apresenta-se na Fig.(6.23.a) as curvas de aprendizagem CA_AR1, CA_AR2 e CA_AR3, referentes ao algoritmo $DSTr_0$-LMS. Os valores extraídos dessas curvas de aprendizagem são apresentados na primeira linha das Tabelas 7, 8 e 9. Foram calculados para cada uma das curvas, os valores de $\hat{J}(I_1)$ e $\hat{J}(I_2)$ e pela mesma justificativa dos casos anteriores, adotou-se o valor de $\hat{J}(I_1)$ como sendo o erro de regime permanente $\hat{J}(\infty)$, utilizado nos cálculos relacionados a esse algoritmo. Pelos valores dessas tabelas, verificou-se que com a mudança do valor de $nc = 7.83$ relacionado ao processo AR3 para $nc = 44.25$ relacionado ao processo AR2, ocorreu um aumento de 0.73% no $\hat{J}(\infty)$. Com a mudança de $nc = 7.83$ para $nc = 100.04$ relacionado ao processo AR1, ocorreu um aumento de 0.27% no $\hat{J}(\infty)$. Isso demonstra uma elevada insensibilidade desse algoritmo, pois os valores de $\hat{J}(\infty)$ mantiveram-se praticamente inalterados com a variação dos "*nc*".

Com a mudança dos extremos $nc = 7.83$ para $nc = 100.04$ verificou-se um desempenho superior dos algoritmos D-LMS, Dr_0-LMS, DSTr-LMS e $DSTr_0$-LMS sobre os algoritmos DCT-LMS, DST-LMS e NLMS, pois para o primeiro não houve variação, para o segundo houve uma variação do $\hat{J}(\infty)$ de apenas 0.09%, para o terceiro houve uma variação do $\hat{J}(\infty)$ de 0.18%, para o quarto houve uma variação de 0.27%, para o quinto houve uma variação de 0.47%, para o sexto de 2.5% e para o sétimo houve uma variação de 22.1%.

Os detalhes das curvas CA_AR1, CA_AR2 e CA_AR3 apresentados na Fig.(6.23.b) juntamente com os valores exibidos nas Tabelas 7, 8 e 9 auxiliam na avaliação da taxa de convergência do algoritmo $DSTr_0$-LMS submetido aos processos AR1, AR2 e AR3. Verifica-se nesses detalhes que a curva CA_AR1 possuiu a convergência mais lenta, por conseguinte, o I_3 com maior limite inferior, ou seja, I_3=225:325. A curva CA_AR3 obteve a convergência mais rápida com I_3=101:201.

O menor erro quadrático médio de regime permanente $\hat{J}(\infty) = 1.093$ foi obtido pela curva CA_AR3, no entanto, o erro de regime permanente registrado pela curva CA_AR2 ficou muito próximo desse valor, com uma diferença de apenas 0.73%, ou seja, $\hat{J}(\infty) = 1.101$ e para a curva CA_AR1 a diferença foi de apenas 0.27%, ou seja, $\hat{J}(\infty) = 1.096$.

Comparou-se a taxa de convergência entre os algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS, DSTr-LMS, Dr_0-LMS e $DSTr_0$-LMS, verificou-se que para a curva CA_AR1 o algoritmo DSTr-LMS foi o mais rápido, pois obteve o I_3 com o menor limite inferior, ou seja, I_3=134:234, em segundo lugar veio o $DSTr_0$-LMS com I_3=225:325, em terceiro veio o DCT-LMS com I_3=394:494, seguido pelos algoritmos D-LMS, Dr_0-LMS, DST-LMS e por último o NLMS.

Para a curva CA_AR2, o algoritmo DSTr-LMS foi o mais rápido com I_3=88:188, em segundo lugar veio o $DSTr_0$-LMS com I_3=124:224, em terceiro veio o DCT-LMS com I_3=237:337, seguido pelos algoritmos D-LMS, DST-LMS, Dr_0-LMS e por último o NLMS.

Para a curva CA_AR3, o algoritmo DSTr-LMS foi o mais rápido,em segundo lugar veio o $DSTr_0$-LMS, em terceiro veio o DST-LMS, seguido pelos algoritmos DCT-LMS, D-LMS, Dr_0-LMS e por último o NLMS.

O menor erro de regime permanente foi obtido pelo DCT-LMS para CA_AR1, ou seja, $\hat{J}(\infty)$=1.057, em segundo lugar veio o Dr_0-LMS com uma diferença de 0.7% ou seja, $\hat{J}(\infty)$=1.064, em terceiro veio o D-LMS com uma diferença de 1.8%, ou seja, $\hat{J}(\infty)$=1.076, em quarto veio o DST-LMS, em quinto veio o $DSTr_0$-LMS com uma diferença de 3.7%, em sétimo veio o DSTr-LMS com uma diferença de 4.9% e por último o NLMS.

Para a curva CA_AR2, o menor erro foi obtido pelo DCT-LMS, seguido pelos algoritmos DST-LMS, Dr_0-LMS, D-LMS, NLMS, $DSTr_0$-LMS e DSTr-LMS. Para a curva CA_AR3, o menor erro foi obtido pelo NLMS com $\hat{J}(\infty)$=1.030.

6.5.2 <u>Simulação-16</u>: Análise do Erro de Regime Permanente para os Algoritmos $DSTr_i$-LMS e $DSTr_{ij}$-LMS, Quando Submetidos a Predição Linear dos Processos AR1, AR2 e AR3

Essa análise possui o seguinte objetivo:

- Verificar a variação do erro de regime permanente para os algoritmos $DSTr_i$-LMS e $DSTr_{ij}$-LMS em relação ao erro de regime do algoritmo DSTr-LMS, ou seja, com a eliminação de uma ou mais entradas $r_{DST,i}(n)$ será observado, em diversas simulações, uma diminuição do $\hat{J}(\infty)$.

Observando as Tabelas 7, 8 e 9 e as Figs.(6.23.a), (6.23.b), (6.24.a), (6.24.b), (6.25.a), (6.25.b), (6.26.a), (6.26.b), (6.27.a), (6.27.b), (6.28.a), (6.28.b), (6.29.a), (6.29.b), (6.30.a), (6.30.b), (6.31.a), (6.31.b), (6.32.a) e (6.32.b), verifica-se que a eliminação de algumas entradas, de forma geral, levou a uma diminuição do erro de regime permanente em relação ao DSTr-LMS, pois para a curva CA_AR1 o valor desse parâmetro obtido pelo DSTr-LMS foi $\hat{J}(\infty)$=1.109, para o $DSTr_0$-LMS obteve-se $\hat{J}(\infty)$=1.096, para o $DSTr_1$-LMS obteve-se $\hat{J}(\infty)$=1.096, para o $DSTr_2$-LMS obteve-se $\hat{J}(\infty)$=1.096, para o $DSTr_3$-LMS obteve-se $\hat{J}(\infty)$=1.096, para o $DSTr_{01}$-LMS obteve-se $\hat{J}(\infty)$=1.084, para o $DSTr_{02}$-LMS e para o $DSTr_{13}$-LMS ocorreu uma estabilização do erro de regime permanente em valores muito acima do esperado, tornando-os inutilizáveis. Para o $DSTr_{03}$-LMS obteve-se $\hat{J}(\infty)$=1.084, para o $DSTr_{12}$-LMS obteve-se $\hat{J}(\infty)$=1.084 e para o $DSTr_{23}$-LMS obteve-se $\hat{J}(\infty)$=1.084.

Para a curva CA_AR2 também constatou-se que a eliminação de algumas entradas proporcionou uma diminuição do erro de regime permanente em relação ao DSTr-LMS, pois o valor desse parâmetro obtido pelo DSTr-LMS foi $\hat{J}(\infty)$=1.117, para o $DSTr_0$-LMS obteve-se $\hat{J}(\infty)$=1.101, para o $DSTr_1$-LMS obteve-se $\hat{J}(\infty)$=1.102, para o $DSTr_2$-LMS obteve-se $\hat{J}(\infty)$=1.101, para o $DSTr_3$-LMS obteve-se $\hat{J}(\infty)$=1.102, para o $DSTr_{01}$-LMS obteve-se $\hat{J}(\infty)$=1.087, para o $DSTr_{02}$-LMS e para o $DSTr_{13}$-LMS ocorreu uma estabilização do erro de regime permanente em valores muito acima do esperado, tornando-os inutilizáveis, etc. Para a curva CA_AR3 também constatou-se que a eliminação de algumas entradas proporcionou uma diminuição do erro de regime permanente em relação ao DSTr-LMS.

6.5.3 <u>Simulação-17</u>: Análise da Taxa de Convergência para os Algoritmos $DSTr_i$-LMS e $DSTr_{ij}$-LMS, Quando Submetidos a Predição Linear dos Processos AR1, AR2 e AR3

Essa análise possui o seguinte objetivo:

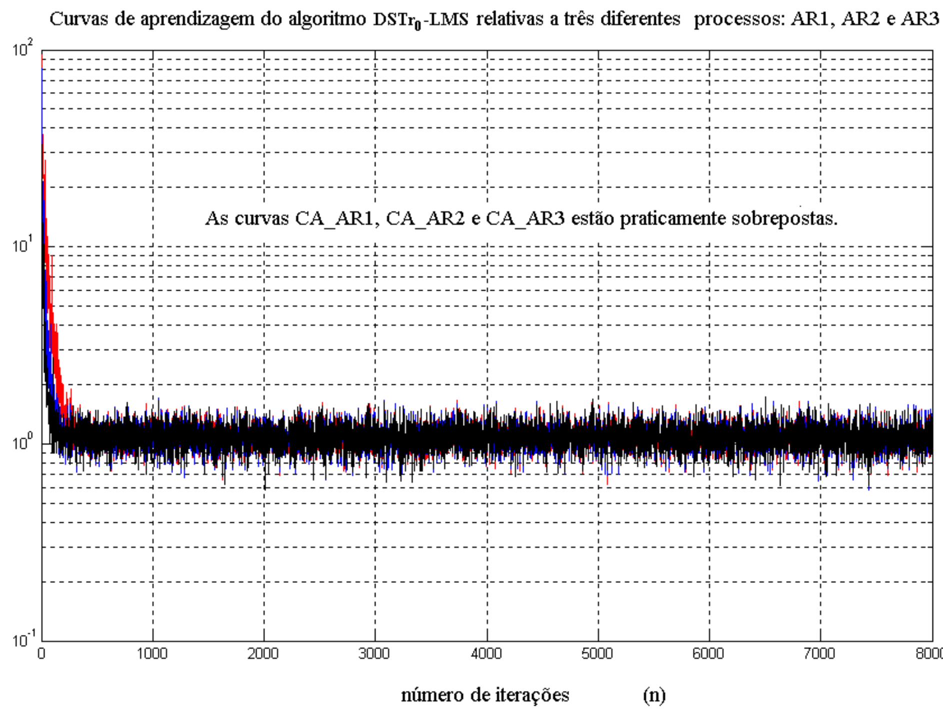
- Verificar a variação da taxa de convergência para os algoritmos $DSTr_i$-LMS e $DSTr_{ij}$-LMS em relação a taxa de convergência DSTr-LMS, ou seja, com a eliminação de uma ou mais entradas $r_{DST,i}(n)$ será observado, em diversas simulações, um leve aumento da taxa de convergência.

Com relação ao aspecto taxa de convergência, analisou-se as Tabelas 7, 8 e 9 e as Figs.(6.23.a), (6.23.b), (6.24.a), (6.24.b), (6.25.a), (6.25.b), (6.26.a), (6.26.b), (6.27.a), (6.27.b), (6.28.a), (6.28.b), (6.29.a), (6.29.b), (6.30.a), (6.30.b), (6.31.a), (6.31.b), (6.32.a) e (6.32.b), verificou-se que em certos casos, a eliminação de algumas entradas conduziu a um leve aumento na taxa de convergência obtida por esse procedimento em relação a obtida através do algoritmo DSTr-LMS. É importante salientar que apesar do pequeno aumento na taxa (pois ainda assim ela o é competitiva em relação a de outros algoritmos), obteve-se em contrapartida a vantagem de se gerar uma diminuição do erro de regime permanente em relação ao DSTr-LMS.

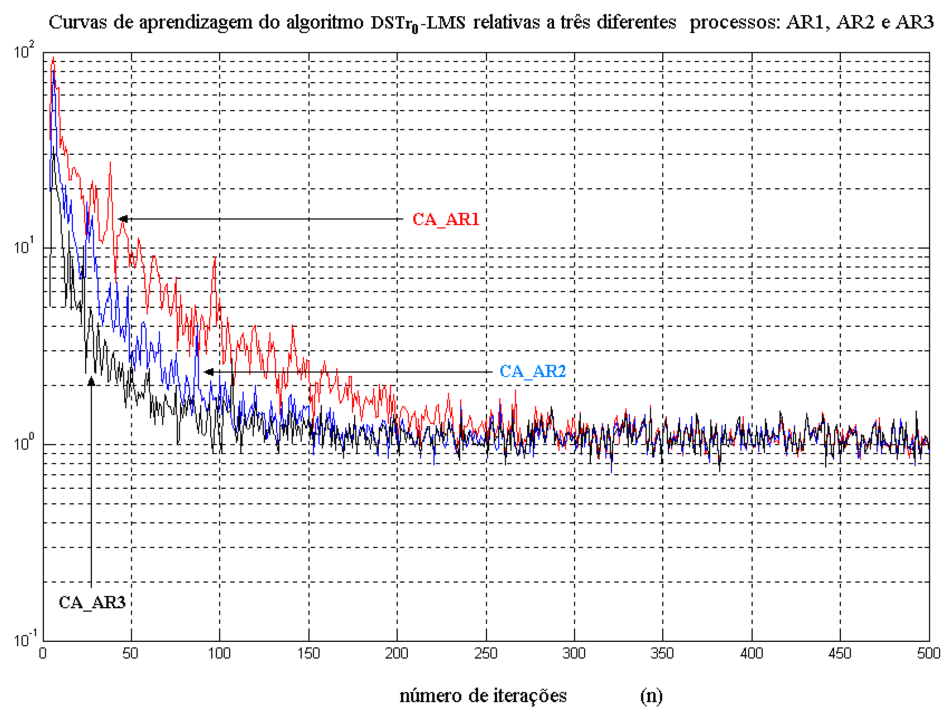
Para a curva CA_AR1, obteve-se I_3=134:234 para o DSTr-LMS, obteve-se I_3=151:251 para o $DSTr_3$-LMS, I_3=155:255 para o $DSTr_1$-LMS, I_3=205:305 para o $DSTr_{03}$-LMS e I_3=206:306 para o $DSTr_{01}$-LMS.

Para a curva CA_AR2, obteve-se I_3=88:188 para o DSTr-LMS, obteve-se I_3=112:212 para o $DSTr_3$-LMS e $DSTr_1$-LMS, obteve-se I_3=124:224 para o $DSTr_0$-LMS, I_3=130:230 para o $DSTr_{01}$-LMS e I_3=135:235 para o $DSTr_{03}$-LMS.

Para a curva CA_AR3, obteve-se I_3=75:175 para o DSTr-LMS, obteve-se I_3=94:194 para o $DSTr_1$-LMS, I_3=95:195 para o $DSTr_{01}$-LMS, I_3=101:201 para o $DSTr_0$-LMS, I_3=105:205 para o $DSTr_3$-LMS, etc.

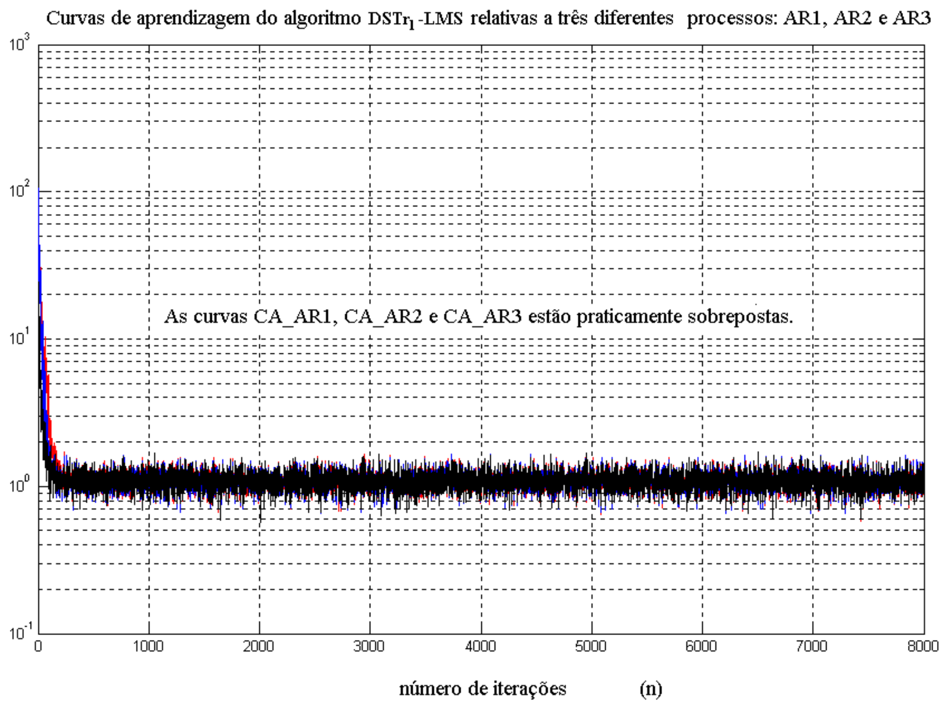


a)

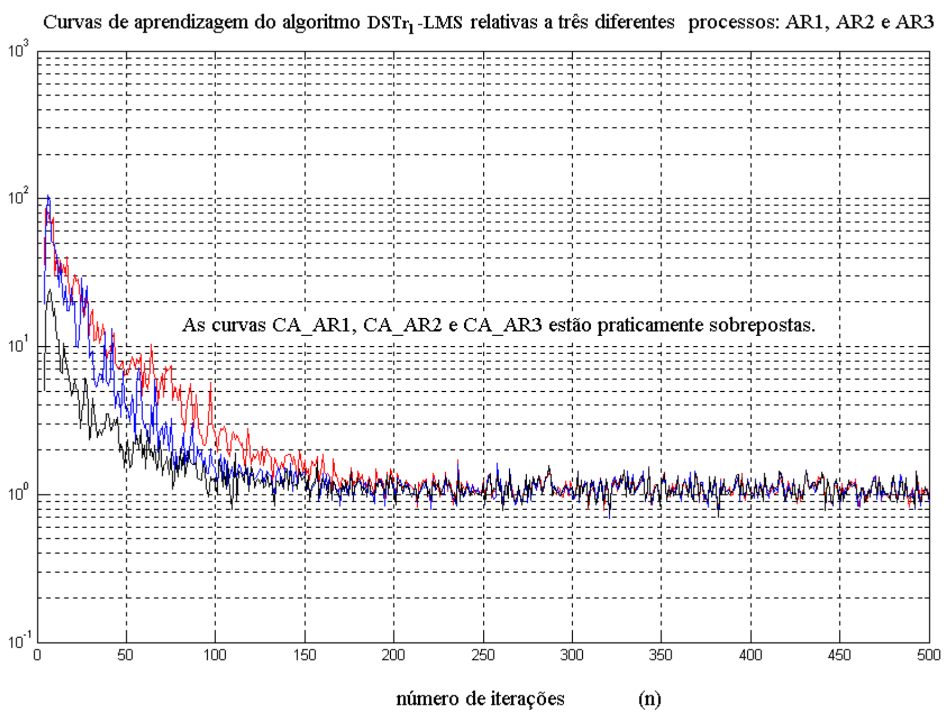


b)

Fig.6.23. a)Curvas de aprendizagem para o algoritmo $DSTr_0$-LMS usado em predição linear, considerando-se que N=4; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

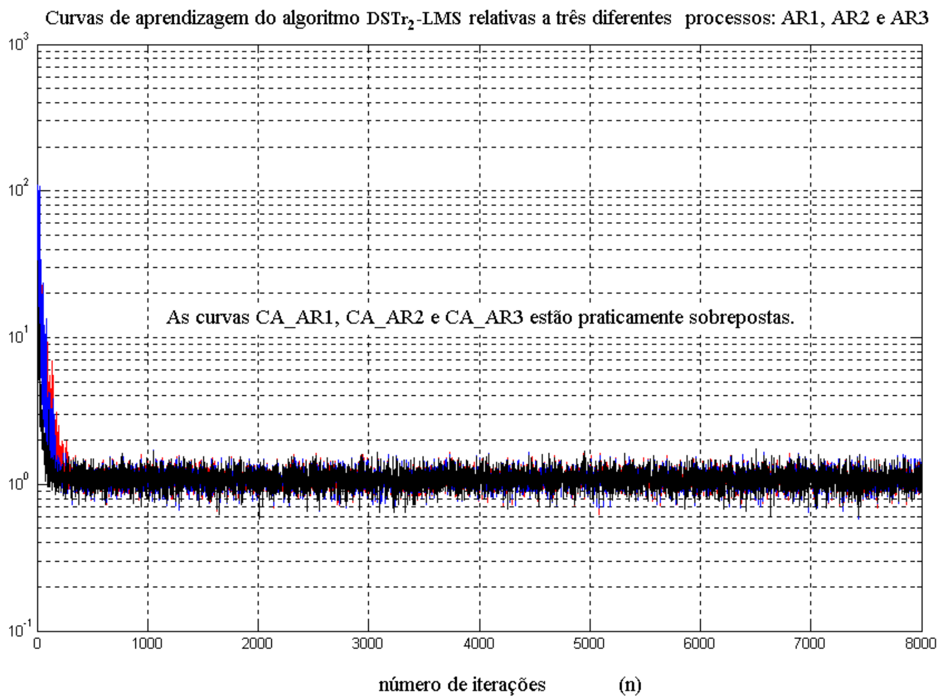


a)



b)

Fig.6.24. a)Curvas de aprendizagem para o algoritmo $DSTr_1$-LMS usado em predição linear, considerando-se que N=4; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.



a)

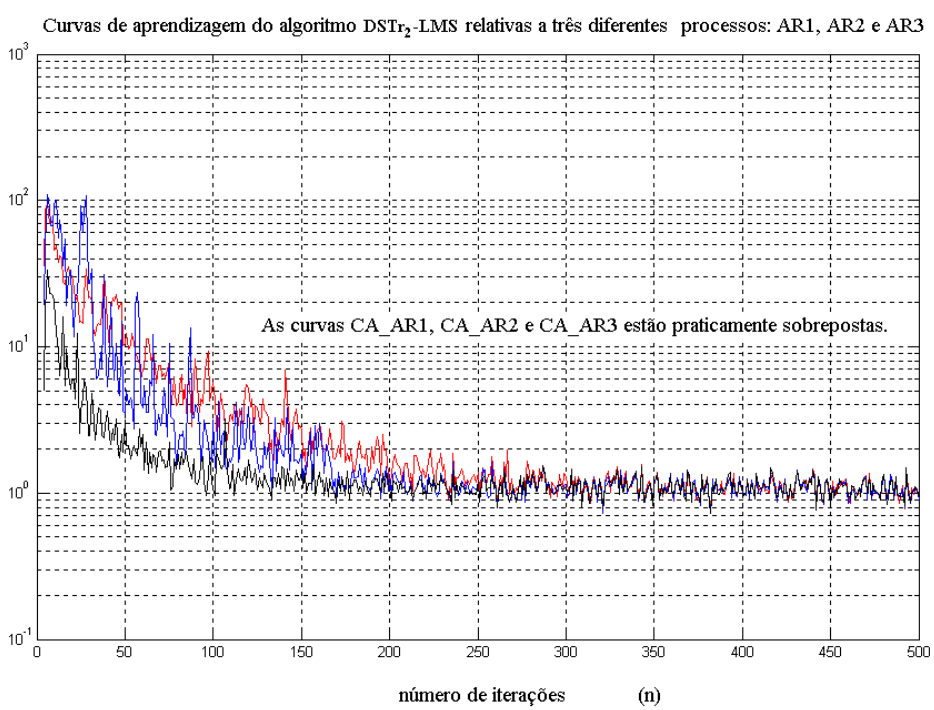
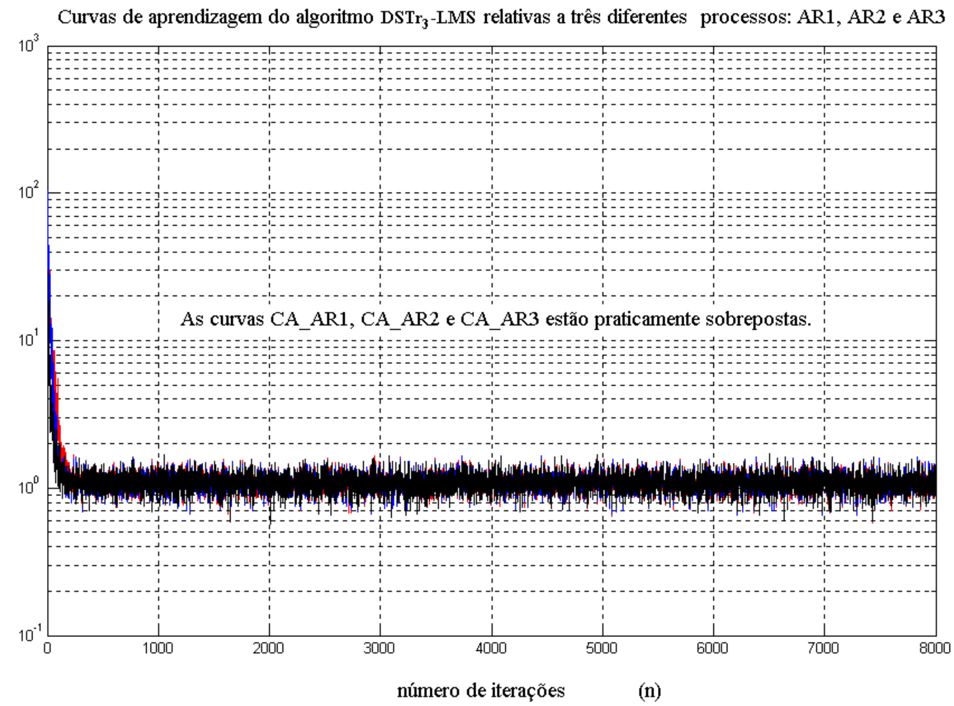


Fig.6.25. a)Curvas de aprendizagem para o algoritmo $DSTr_2$-LMS usado em predição linear, considerando-se que N=4; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.



a)

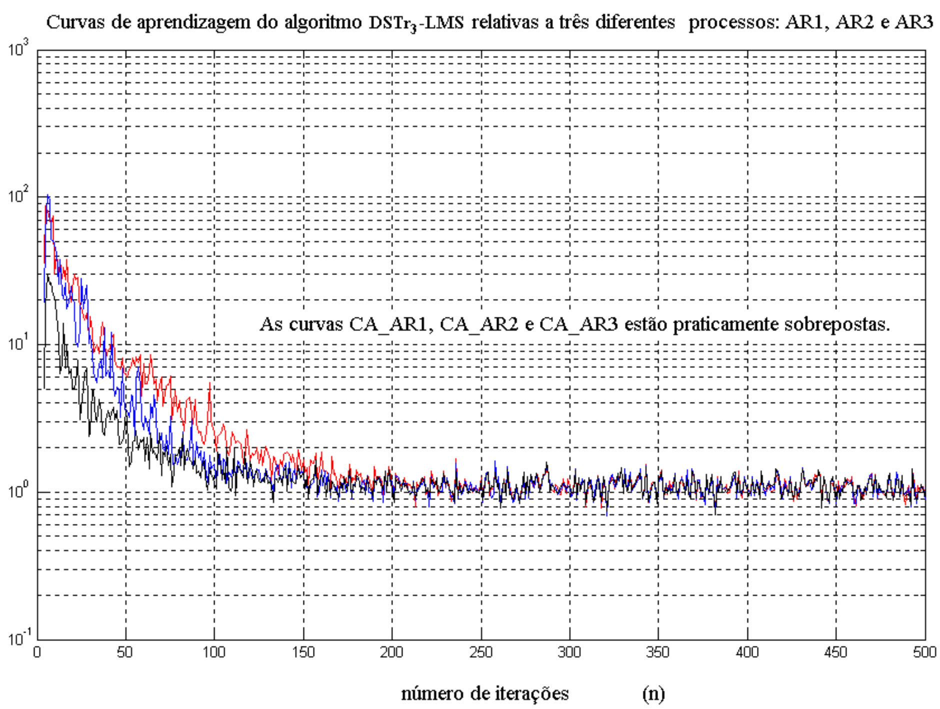
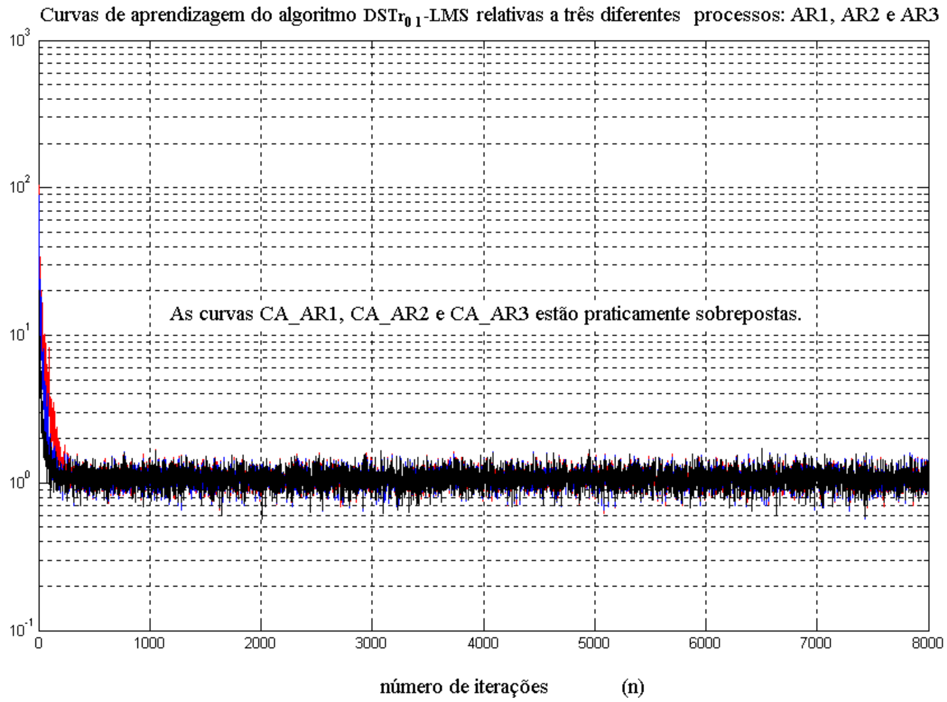
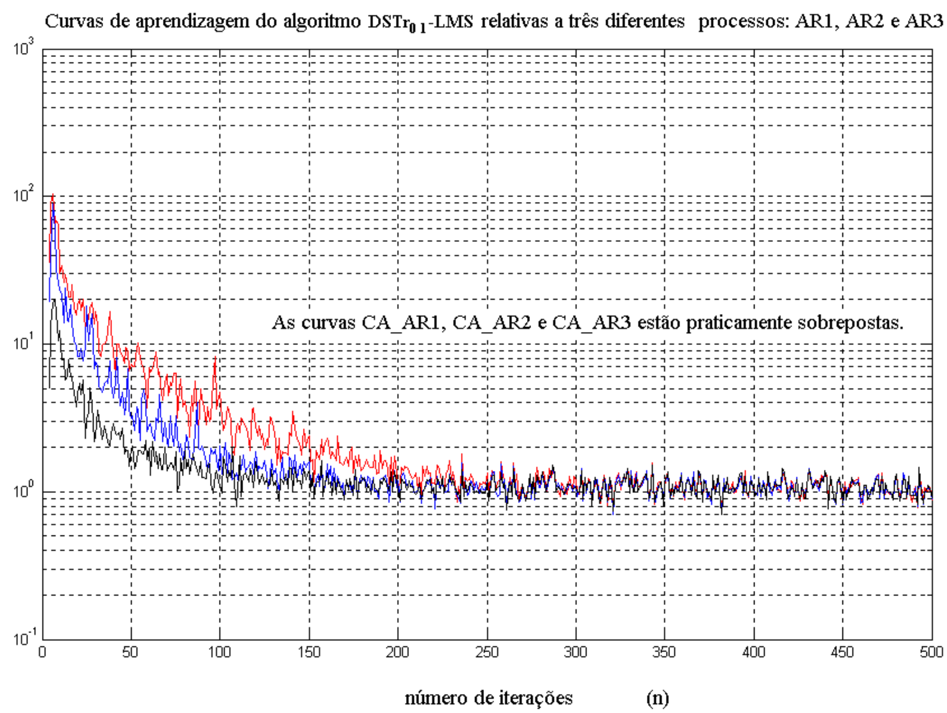


Fig.6.26. a)Curvas de aprendizagem para o algoritmo $DSTr_3$-LMS usado em predição linear, considerando-se que N=4; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

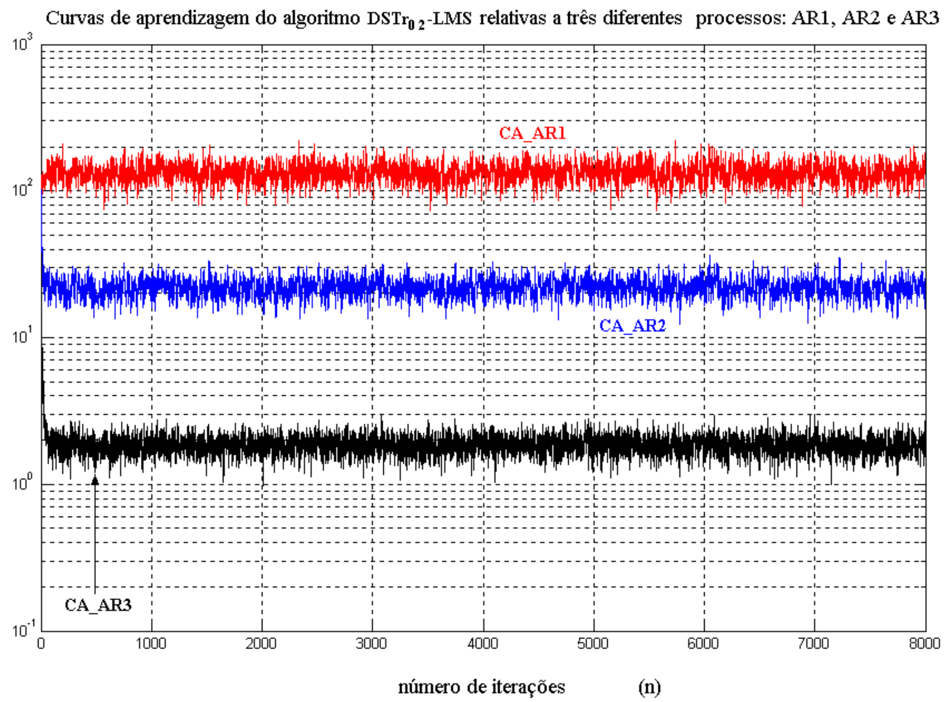


a)

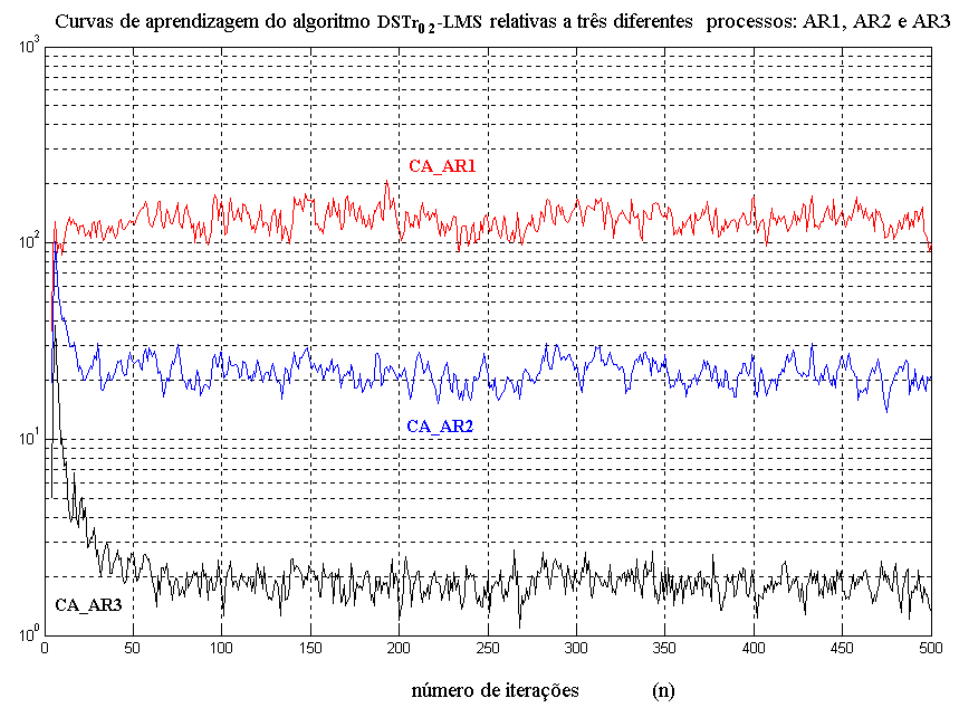


b)

Fig.6.27. a)Curvas de aprendizagem para o algoritmo $DSTr_{01}$-LMS usado em predição linear, considerando-se que N=4; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

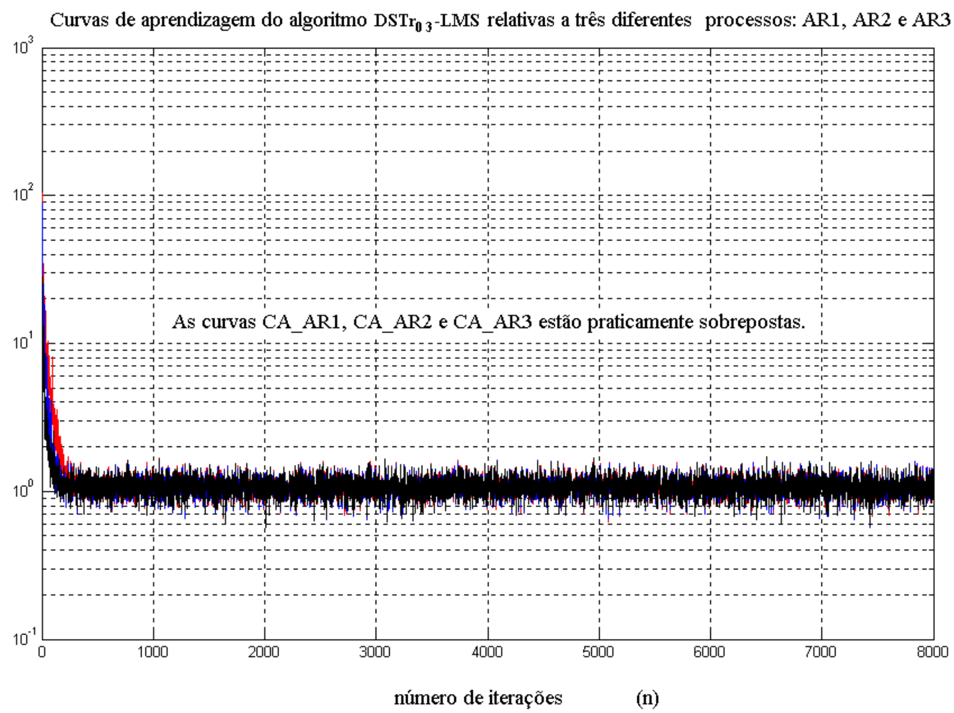


a)

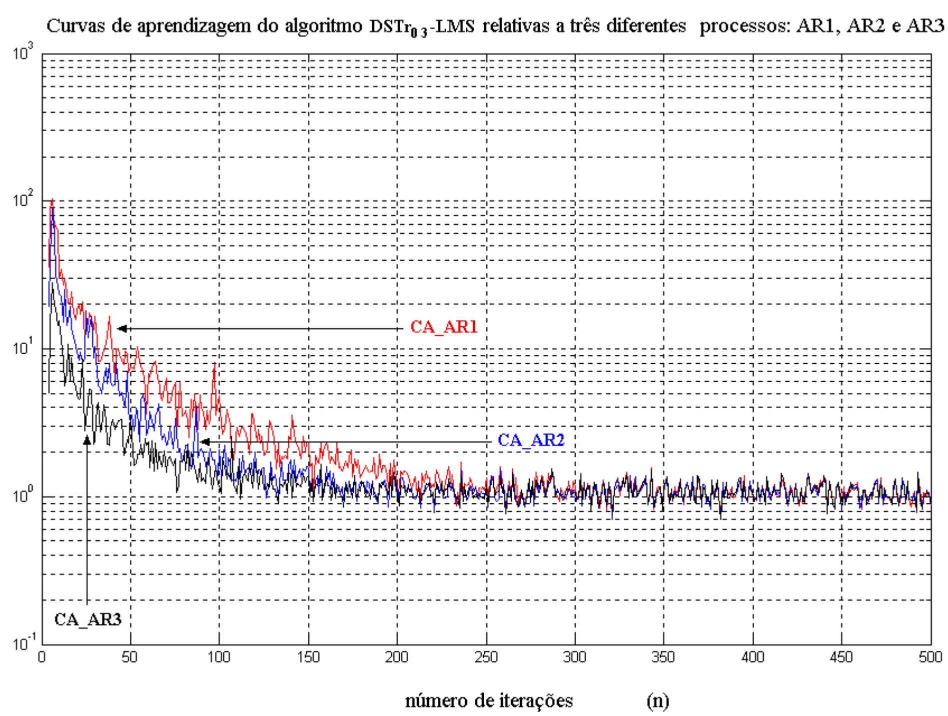


b)

Fig.6.28. a)Curvas de aprendizagem para o algoritmo DSTr$_{02}$-LMS usado em predição linear, considerando-se que $N=4$; $\mu=0.02$; $\beta=0.90$; $\delta= 10^{-4}$. b)Detalhes dessa curva.

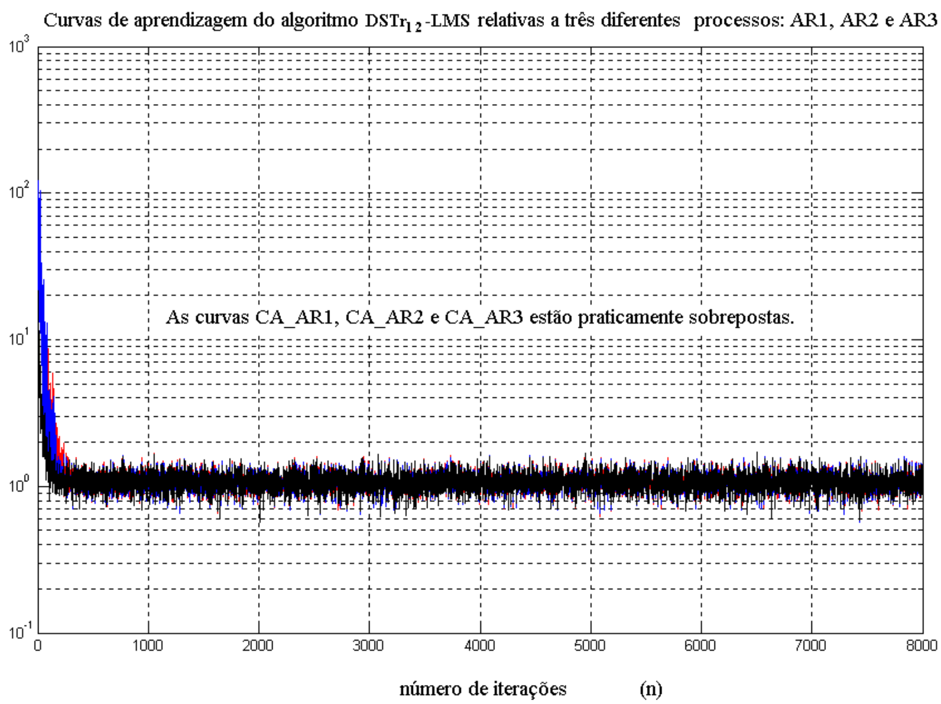


a)

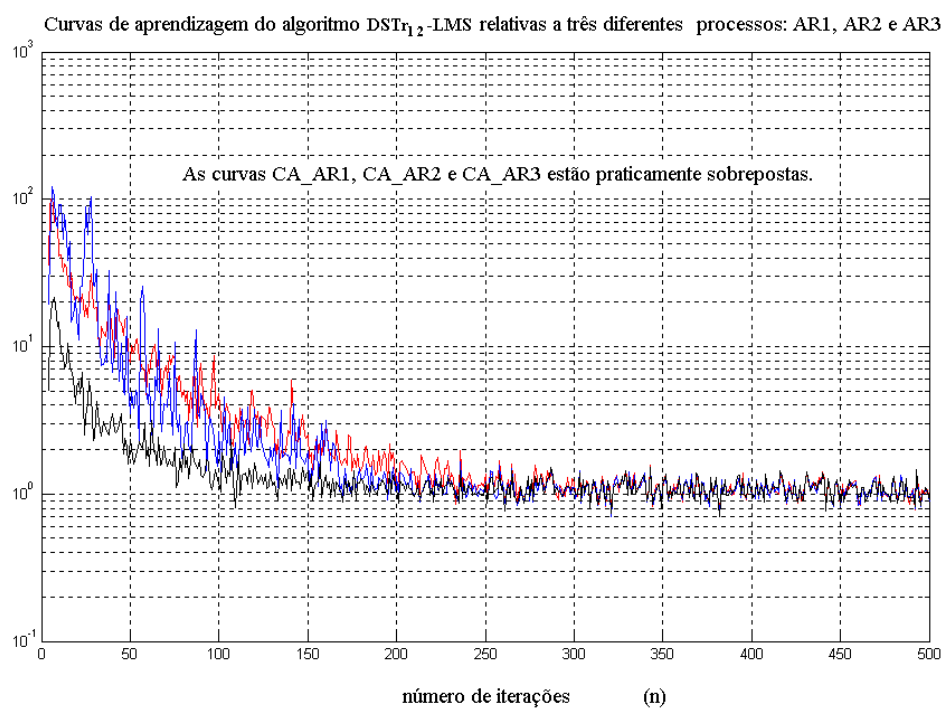


b)

Fig.6.29. a)Curvas de aprendizagem para o algoritmo $DSTr_{03}$-LMS usado em predição linear, considerando-se que $N=4$; $\mu=0.02$; $\beta=0.90$; $\delta= 10^{-4}$. b)Detalhes dessa curva.

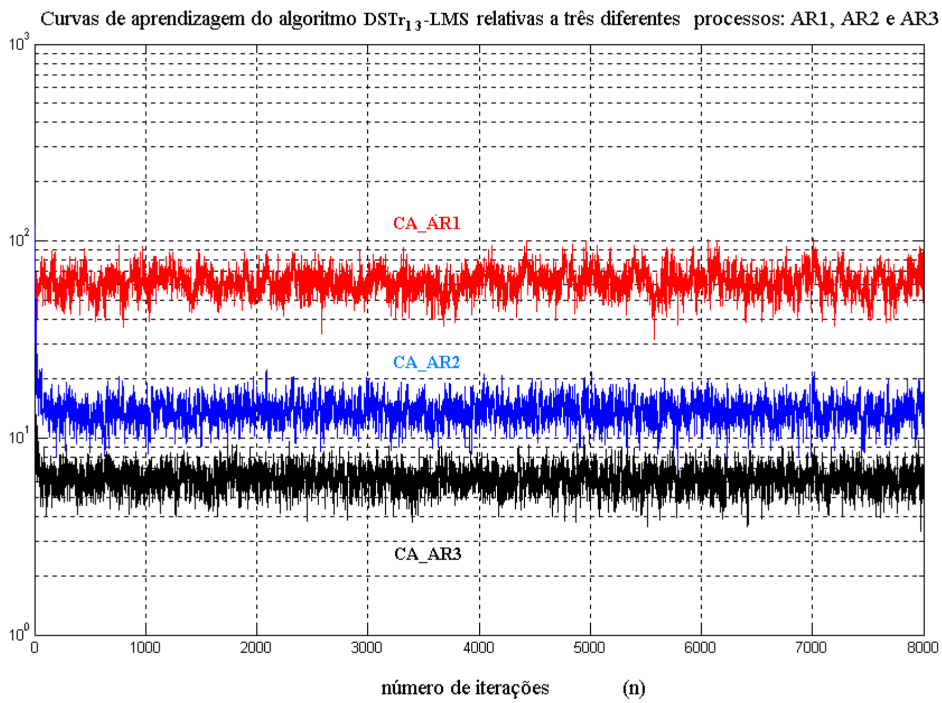


a)

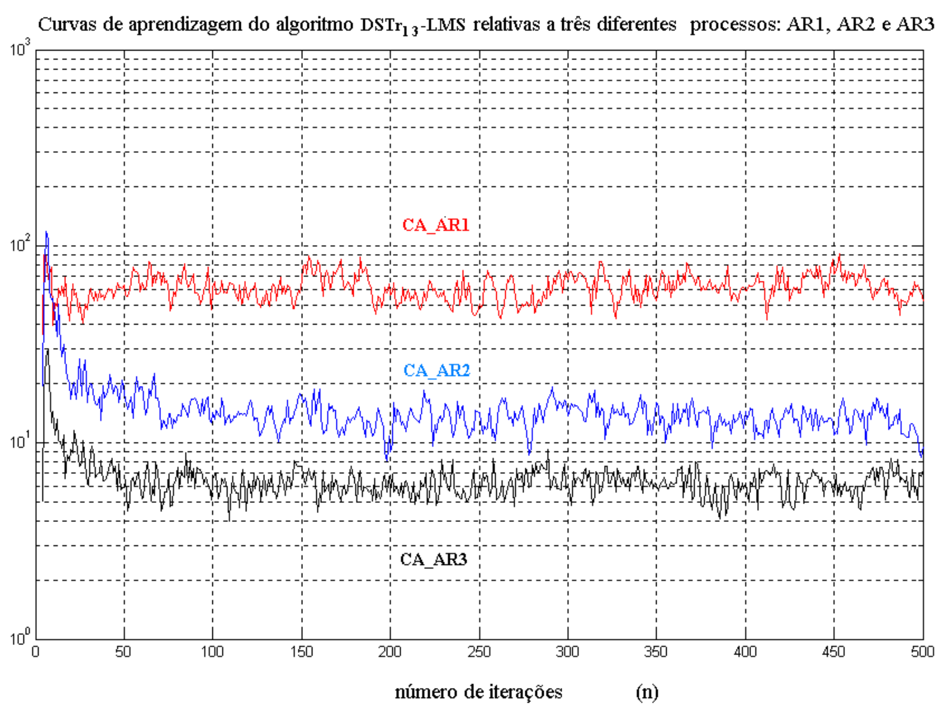


b)

Fig.6.30. a)Curvas de aprendizagem para o algoritmo $DSTr_{12}$-LMS usado em predição linear, considerando-se que $N=4$; $\mu=0.02$; $\beta=0.90$; $\delta= 10^{-4}$. b)Detalhes dessa curva.

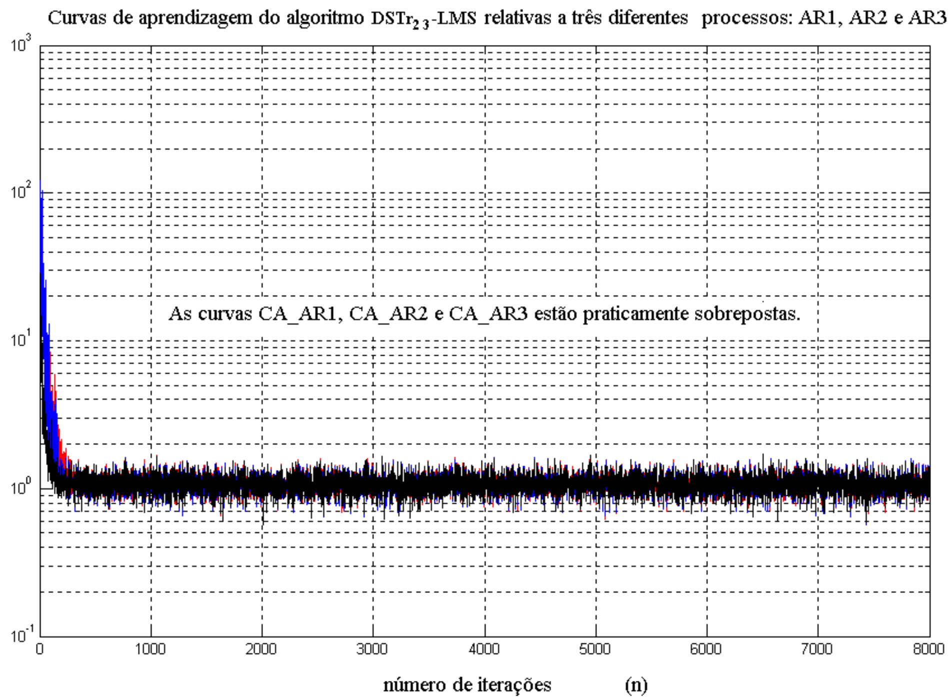


a)

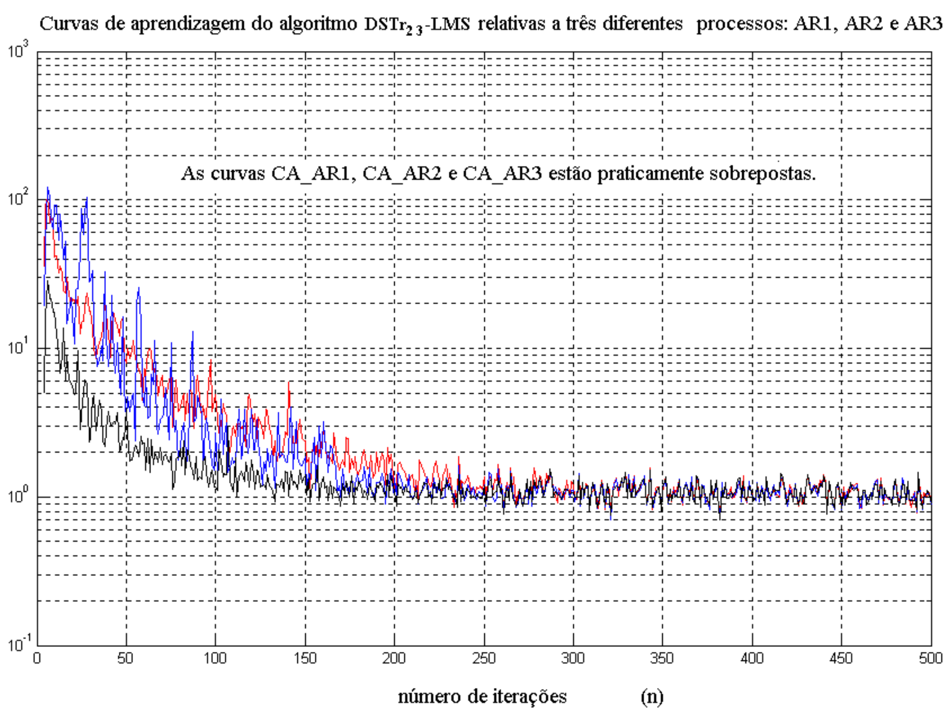


b)

Fig.6.31. a)Curvas de aprendizagem para o algoritmo $DSTr_{13}$-LMS usado em predição linear, considerando-se que N=4; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.



a)



b)

Fig.6.32. a)Curvas de aprendizagem para o algoritmo $DSTr_{23}$-LMS usado em predição linear, considerando-se que N=4; μ=0.02 ; β=0.90 ; δ= 10^{-4}. b)Detalhes dessa curva.

	Processo AR1 ; $\mu = 0.02$, $\beta = 0.90$ e $\delta = 10^{-4}$ (Dados obtidos da Curva CA_AR1, relacionada ao processo AR1)			
	Valor de pico	Valores Médios $\hat{J}(I_3) \leq 1.1 \times \hat{J}(\infty)$ (Intervalo I_3 é variável)	Erro de regime permanente $\hat{J}(\infty)$ $\hat{J}(\infty) = \hat{J}(I_1) \cong \hat{J}(I_2)$ ou $I_1 = 7800{:}7999$; $I_2 = 7000{:}7199$	$D = \hat{J}(I_2)/\hat{J}(\infty)$; $\%\Delta = (D\text{-}1)\times 100$
DSTr$_0$-LMS	$\hat{J}(6) = 95$	$\hat{J}(225{:}325) = 1.20$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.096$; $\hat{J}(I_2) = 1.098$	0.18 %
DSTr$_1$-LMS	$\hat{J}(5) = 86$	$\hat{J}(155{:}255) = 1.20$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.096$; $\hat{J}(I_2) = 1.099$	0.27 %
DSTr$_2$-LMS	$\hat{J}(6) = 95$	$\hat{J}(248{:}348) = 1.20$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.096$; $\hat{J}(I_2) = 1.098$	0.18 %
DSTr$_3$-LMS	$\hat{J}(5) = 87$	$\hat{J}(151{:}251) = 1.20$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.096$; $\hat{J}(I_2) = 1.099$	0.27 %
DSTr$_{01}$LMS	$\hat{J}(6) = 104$	$\hat{J}(206{:}306) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.084$; $\hat{J}(I_2) = 1.085$	0.09 %
DSTr$_{02}$-LMS	-	-	$\hat{J}(\infty) = \hat{J}(I_1) = 134.1$; $\hat{J}(I_2) = 134.9$	0.6 %
DSTr$_{03}$-LMS	$\hat{J}(6) = 104$	$\hat{J}(205{:}305) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.084$; $\hat{J}(I_2) = 1.085$	0.09 %
DSTr$_{12}$-LMS	$\hat{J}(6) = 105$	$\hat{J}(220{:}320) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.084$; $\hat{J}(I_2) = 1.085$	0.09 %
DSTr$_{13}$-LMS	-	-	$\hat{J}(\infty) = \hat{J}(I_1) = 65.5$; $\hat{J}(I_2) = 64.7$	-1.2 %
DSTr$_{23}$-LMS	$\hat{J}(6) = 105$	$\hat{J}(218{:}318) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.084$; $\hat{J}(I_2) = 1.085$	0.09 %

Tabela 7. Apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e $\%\Delta$ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos DSTr$_0$-LMS, DSTr$_1$-LMS, DSTr$_2$-LMS, DSTr$_3$-LMS, DSTr$_{01}$LMS, DSTr$_{02}$-LMS, DSTr$_{03}$-LMS, DSTr$_{12}$-LMS, DSTr$_{13}$-LMS e DSTr$_{23}$-LMS, sujeitos aos parâmetros $\mu = 0.02$, $\beta = 0.90$, $\delta = 10^{-4}$ e processo AR1.

	Processo AR2 ; $\mu = 0.02$, $\beta = 0.90$ e $\delta = 10^{-4}$ (Dados obtidos da Curva CA_AR2, relacionada ao processo AR2)			
	Valor de pico	Valores Médios $\hat{J}(I_3) \le 1.1 \times \hat{J}(\infty)$ (Intervalo I_3 é variável)	Erro de regime permanente $\hat{J}(\infty)$ $\hat{J}(\infty) = \hat{J}(I_1) \cong \hat{J}(I_2)$ $I_1 = 7800{:}7999$; $I_2 = 7000{:}7199$	$D = \hat{J}(I_2)/\hat{J}(\infty)$; $\%\Delta = (D-1)\times 100$
$DSTr_0$-LMS	$\hat{J}(6) = 81$	$\hat{J}(124{:}224) = 1.21$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.101$; $\hat{J}(I_2) = 1.095$	-0.54 %
$DSTr_1$-LMS	$\hat{J}(6) = 105$	$\hat{J}(112{:}212) = 1.21$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.102$; $\hat{J}(I_2) = 1.097$	-0.45 %
$DSTr_2$-LMS	$\hat{J}(6) = 109$	$\hat{J}(161{:}261) = 1.21$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.101$; $\hat{J}(I_2) = 1.095$	-0.5 %
$DSTr_3$-LMS	$\hat{J}(6) = 105$	$\hat{J}(112{:}212) = 1.21$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.102$; $\hat{J}(I_2) = 1.097$	-0.45 %
$DSTr_{01}$LMS	$\hat{J}(6) = 89$	$\hat{J}(130{:}230) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.087$; $\hat{J}(I_2) = 1.081$	-0.55 %
$DSTr_{02}$-LMS	-	-	$\hat{J}(\infty) = \hat{J}(I_1) = 22.35$; $\hat{J}(I_2) = 21.96$	-1.74 %
$DSTr_{03}$-LMS	$\hat{J}(6) = 89$	$\hat{J}(135{:}235) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.087$; $\hat{J}(I_2) = 1.081$	-0.55 %
$DSTr_{12}$-LMS	$\hat{J}(6) = 121$	$\hat{J}(164{:}264) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.087$; $\hat{J}(I_2) = 1.081$	-0.55 %
$DSTr_{13}$-LMS	-	-	$\hat{J}(\infty) = \hat{J}(I_1) = 14.03$;$\hat{J}(I_2) = 13.70$	-2.35 %
$DSTr_{23}$-LMS	$\hat{J}(6) = 121$	$\hat{J}(164{:}264) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.087$; $\hat{J}(I_2) = 1.081$	-0.55 %

Tabela 8. Apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e $\%\Delta$ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos $DSTr_0$-LMS, $DSTr_1$-LMS, $DSTr_2$-LMS, $DSTr_3$-LMS, $DSTr_{01}$LMS, $DSTr_{02}$-LMS, $DSTr_{03}$-LMS, $DSTr_{12}$-LMS, $DSTr_{13}$-LMS e $DSTr_{23}$-LMS, sujeitos aos parâmetros $\mu = 0.02$, $\beta = 0.90,$ $\delta = 10^{-4}$ e processo AR2.

	Processo AR3 ; $\mu = 0.02$, $\beta = 0.90$ e $\delta = 10^{-4}$ (Dados obtidos da Curva CA_AR3, relacionada ao processo AR3)			
	Valor de pico	Valores Médios $\hat{J}(I_3) \leq 1.1 \times \hat{J}(\infty)$ (Intervalo I_3 é variável)	Erro de regime permanente $\hat{J}(\infty)$ $\hat{J}(\infty) = \hat{J}(I_1) \cong \hat{J}(I_2)$ $I_1 = 7800{:}7999$; $I_2 = 7000{:}7199$	D = $\hat{J}(I_2)/\hat{J}(\infty)$; $\%\Delta = (D-1)\times 100$
$DSTr_0$-LMS	$\hat{J}(6) = 33$	$\hat{J}(101{:}201) = 1.20$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.093$; $\hat{J}(I_2) = 1.095$	0.18 %
$DSTr_1$-LMS	$\hat{J}(7) = 24$	$\hat{J}(94{:}194) = 1.20$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.093$; $\hat{J}(I_2) = 1.095$	0.18 %
$DSTr_2$-LMS	$\hat{J}(6) = 33$	$\hat{J}(108{:}208) = 1.18$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.093$; $\hat{J}(I_2) = 1.095$	0.18 %
$DSTr_3$-LMS	$\hat{J}(6) = 30$	$\hat{J}(105{:}205) = 1.20$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.093$; $\hat{J}(I_2) = 1.095$	0.18 %
$DSTr_{01}$LMS	$\hat{J}(7) = 20$	$\hat{J}(95{:}195) = 1.18$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.079$; $\hat{J}(I_2) = 1.080$	0.09 %
$DSTr_{02}$-LMS	-	-	$\hat{J}(\infty) = \hat{J}(I_1) = 1.858$; $\hat{J}(I_2) = 1.852$	1 %
$DSTr_{03}$-LMS	$\hat{J}(6) = 28$	$\hat{J}(110{:}210) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.079$; $\hat{J}(I_2) = 1.080$	0.09 %
$DSTr_{12}$-LMS	$\hat{J}(7) = 22$	$\hat{J}(108{:}208) = 1.18$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.079$; $\hat{J}(I_2) = 1.080$	0.09 %
$DSTr_{13}$-LMS	-	-	$\hat{J}(\infty) = \hat{J}(I_1) = 6.26$; $\hat{J}(I_2) = 6.20$	-0.96 %
$DSTr_{23}$-LMS	$\hat{J}(6) = 29$	$\hat{J}(114{:}214) = 1.19$	$\hat{J}(\infty) = \hat{J}(I_1) = 1.079$; $\hat{J}(I_2) = 1.080$	0.09 %

Tabela 9. Apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e $\%\Delta$ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos $DSTr_0$-LMS, $DSTr_1$-LMS, $DSTr_2$-LMS, $DSTr_3$-LMS, $DSTr_{01}$LMS, $DSTr_{02}$-LMS, $DSTr_{03}$-LMS, $DSTr_{12}$-LMS, $DSTr_{13}$-LMS e $DSTr_{23}$-LMS, sujeitos aos parâmetros $\mu = 0.02$, $\beta = 0.90,$ $\delta = 10^{-4}$ e processo AR3.

6.6 <u>Simulação-18</u>: Simulação que Analisa o Desempenho dos Algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS Utilizados na Predição Linear de Sinais EMG

Esta simulação diz respeito à predição linear de sinais eletromiográficos (EMG), obtida conforme o esquema apresentado na Fig.(6.1) e empregando-se os seguintes algoritmos adaptativos: NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS.

Esses sinais são resultantes da atividade elétrica que acompanha a contração muscular, assim, para o caso específico do músculo cardíaco tem-se o eletrocardiograma que é o traçado das correntes produzidas por esse músculo.

Os sinais elétricos utilizados nessa simulação foram detectados através de eletrodos de superfície, posicionados segundo as duas configurações a seguir:

a)Utilizaram-se 5 pares de eletrodos, sendo dois pares posicionados sobre o grupo muscular bíceps e três sobre o grupo muscular tríceps.
b)Um par situado sobre o plexo braquial.

Adicionalmente, tem-se a utilização de um eletrodo de referência situado sobre o ombro nas duas configurações.

Os sinais coletados associados à contração muscular estão relacionados a 4 classes de movimento do braço: flexão dos cotovelos, extensão dos cotovelos, pronação dos punhos e por último, supinação dos punhos que é um movimento de rotação do antebraço que traz a palma da mão para frente ou para cima e o polegar para fora [59]. Além disso, cada um dos movimentos foi executado considerando-se duas formas, ou seja, que a contração do músculo seja isométrica ou isotônica.

A contração muscular isométrica é definida como sendo aquela em que o músculo não altera seu comprimento, apesar de haver aumento da força desenvolvida por ele. A contração muscular isotônica é definida como sendo aquela em que a força desenvolvida permanece constante durante a variação do comprimento do músculo [59].

A taxa de amostragem utilizada para digitalização desses sinais foi de 5Kamostras/s. Disponibiliza-se de um total de 140 sinais eletromiográficos (EMG), cada um deles contendo 943 amostras. Esses sinais são relativos aos quatro tipos de movimentos e as duas formas de contração muscular. Maiores detalhes sobre o tipo e o posicionamento dos eletrodos sobre os músculos, características do equipamento eletrônico de aquisição de sinais, etc, podem ser encontrados em [60].

Os algoritmos adaptativos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS empregados no esquema de predição fornecido pela Fig.(6.1), utilizaram os valores de N= 4 e 5, visto que processos autoregressivos de 4ª ou 5ª ordem são adequados na representação de sinais EMG [61,62].

A Tabela 10 apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e %Δ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS, sujeitos aos parâmetros $\alpha = \mu = 0.02$, $\beta = 0.90,$ δ=10^{-4} , N=4 derivações e sinais EMG na entrada do preditor.

A Tabela 11 apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e %Δ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS, sujeitos aos mesmos parâmetros anteriores, só que considerando N=5. Admitiu-se uma diferença máxima de 20% entre os valores de $J(I_1)$ e $J(I_2)$ para que $J(I_1)$ fosse considerado o valor do erro de regime permanente; os valores para os intervalos I_1, I_2 e I_3 estão especificados nas próprias tabelas.

Considerando-se que o sinais EMG sejam modelados através de um processo AR(4), pode-se concluir, através dos valores contidos na Tabela 10, que o algoritmo DSTr-LMS:

- Atingiu um desempenho muito melhor do que o algoritmo NLMS, seja em termos de erro de regime permanente ou em termos de taxa de convergência.
- Originou um erro de regime permanente quase tão baixo quanto o do algoritmo DCT-LMS, só que associado à uma taxa de convergência muito melhor.
- Obteve uma taxa de convergência tão boa quanto a obtida pelo algoritmo DST-LMS, só que associado a um erro de regime permanente mais baixo.

Considerando-se que o sinais EMG sejam modelados através de um processo AR(5), pode-se extrair as mesmas conclusões para a Tabela 11 que as obtidas para a Tabela 10, só que acrescentado-se o seguinte:

- O desempenho do algoritmo D-LMS melhorou tanto em termos de erro de regime permanente quanto em termos de taxa de convergência, com a modificação de N=4 para N=5. Para N=5 o seu erro de regime permanente foi melhor do que o dos algoritmos NLMS e DST-LMS.
- O desempenho do algoritmo DSTr-LMS manteve-se basicamente constante com a modificação de N=4 para N=5.

	$\alpha = \mu = 0.02$, $\beta = 0.90$, $\delta = 10^{-4}$ e N= 4 (Dados obtidos da Curva CA_EMG, relacionada aos sinais EMG)			
	Valor de pico ($\times 10^{-3}$)	Valores Médios $\hat{J}(I_3) \leq 1.1 \times \hat{J}(\infty)$ ($\times 10^{-3}$) (Intervalo I_3 é variável)	Erro de regime permanente $\hat{J}(\infty)$ $\hat{J}(\infty) = \hat{J}(I_1) \cong \hat{J}(I_2)$ ($\times 10^{-3}$) $I_1 = 943{:}993$; $I_2 = 900{:}943$	D = $\hat{J}(I_2)/\hat{J}(\infty)$; $\%\Delta = (D-1) \times 100$
NLMS	$\hat{J}(4) = 133$	$\hat{J}(641{:}741) = 8.10$	$\hat{J}(\infty) = \hat{J}(I_1) = 7.38$; $\hat{J}(I_2) = 6.71$	10 %
DST-LMS	$\hat{J}(4) = 29$	$\hat{J}(5{:}105) = 4.98$	$\hat{J}(\infty) = \hat{J}(I_1) = 4.64$; $\hat{J}(I_2) = 3.98$	16.6%
DCT-LMS	$\hat{J}(1) = 52$	$\hat{J}(62{:}162) = 4.00$	$\hat{J}(\infty) = \hat{J}(I_1) = 3.64$; $\hat{J}(I_2) = 3.38$	7.7%
D-LMS	$\hat{J}(206) = 68$	$\hat{J}(675{:}775) = 7.02$	$\hat{J}(\infty) = \hat{J}(I_1) = 6.39$; $\hat{J}(I_2) = 6.14$	4%
DSTr-LMS	$\hat{J}(3) = 27$	$\hat{J}(8{:}108) = 4.30$	$\hat{J}(\infty) = \hat{J}(I_1) = 3.94$; $\hat{J}(I_2) = 3.53$	11.6%

Tabela 10. Apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e $\%\Delta$ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS, sujeitos aos parâmetros $\alpha = \mu = 0.02$, $\beta = 0.90$, $\delta = 10^{-4}$, N=4 e sinal EMG na entrada do preditor.

	$\alpha = \mu = 0.02$, $\beta = 0.90$, $\delta = 10^{-4}$ e N= 5 (Dados obtidos da Curva CA_EMG, relacionada aos sinais EMG)			
	Valor de pico ($\times 10^{-3}$)	Valores Médios $\hat{J}(I_3) \leq 1.1 \times \hat{J}(\infty)$ ($\times 10^{-3}$) (Intervalo I_3 é variável)	Erro de regime permanente $\hat{J}(\infty)$ $\hat{J}(\infty) = \hat{J}(I_1) \cong \hat{J}(I_2)$ ($\times 10^{-3}$) $I_1 = 943{:}993$; $I_2 = 900{:}943$	D = $\hat{J}(I_2)/\hat{J}(\infty)$; $\%\Delta = (D-1) \times 100$
NLMS	$\hat{J}(4) = 139$	$\hat{J}(548{:}648) = 9.51$	$\hat{J}(\infty) = \hat{J}(I_1) = 8.65$; $\hat{J}(I_2) = 7.55$	14.5 %
DST-LMS	$\hat{J}(2) = 19$	$\hat{J}(4{:}104) = 4.90$	$\hat{J}(\infty) = \hat{J}(I_1) = 4.52$; $\hat{J}(I_2) = 3.99$	13.3%
DCT-LMS	$\hat{J}(91) = 60$	$\hat{J}(129{:}229) = 4.01$	$\hat{J}(\infty) = \hat{J}(I_1) = 3.66$; $\hat{J}(I_2) = 3.42$	7%
D-LMS	$\hat{J}(206) = 78$	$\hat{J}(579{:}679) = 4.51$	$\hat{J}(\infty) = \hat{J}(I_1) = 4.10$; $\hat{J}(I_2) = 3.81$	7.6%
DSTr-LMS	$\hat{J}(3) = 25$	$\hat{J}(9{:}109) = 4.41$	$\hat{J}(\infty) = \hat{J}(I_1) = 4.05$; $\hat{J}(I_2) = 3.61$	12.2%

Tabela 11. Apresenta os valores de $\hat{J}(I_1)$, $\hat{J}(I_2)$, $\hat{J}(I_3)$, $\hat{J}(\infty)$ e $\%\Delta$ com os respectivos intervalos I_i, obtidos a partir das curvas de aprendizagem dos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS, sujeitos aos parâmetros $\alpha = \mu = 0.02$, $\beta = 0.90$, $\delta = 10^{-4}$, N=5 e sinal EMG na entrada do preditor.

CAPÍTULO 7

Conclusões e Trabalhos Futuros

7.1 Introdução

Dentro da área de processamento digital de sinais aleatórios, especificamente a filtragem adaptativa, vem sendo estudada há longo tempo. No entanto, continua ainda a ser muito pesquisada, seja em função de sua vasta gama de aplicações, seja em função da busca de novas técnicas de filtragem, de novos algoritmos adaptativos ou mesmo de novos resultados que confirmem as propriedades inerentes dos algoritmos pré-existentes.

7.2. Conclusões, Contribuições Mais Importantes e Trabalhos Futuros

Neste trabalho procurou-se investigar um algoritmo muito utilizado e pesquisado em filtragem adaptativa denominado de algoritmo LMS no domínio da transformada. O foco do trabalho é investigar como incluir na estrutura desse algoritmo a Transformada Discreta Seno com Pré-rotação de Eixos.

Essa transformada é muito recente e sua aplicação se estende desde a interpolação de sinais até a redução do efeito de blocos em imagens comprimidas. Partindo-se da definição da Transformada DSTr, observou-se que a mesma efetuava basicamente a seguinte operação: decompor o sinal de entrada e em seguida aplicar a Transformada Discreta Seno em uma das partes decompostas.

Antes de investigar o problema da inclusão dessa transformada no algoritmo LMS, optou-se por primeiramente fazer uma análise dessa inclusão no problema da filtragem de Wiener, por se tratar de um passo que conduzirá a um esquema tradicional de filtragem adaptativa, isto é, aquele que utiliza o algoritmo LMS para solução das equações de Wiener-Hopf, só que de forma iterativa e sem o conhecimento prévio da estatística dos sinais envolvidos.

Feitas essas ressalvas, no Capítulo 2 fez-se uma introdução sobre o problema da filtragem linear e desenvolveu-se as equações de Wiener-Hopf para três situações muito conhecidas: filtro IIR não causal, filtro IIR causal e por último e de maior interesse para este trabalho, o filtro FIR causal. Esse capítulo finaliza abordando as propriedades da função custo J e a sua interpretação geométrica.

A primeira contribuição deste trabalho aparece no Capítulo 3 com a proposta da inclusão da decomposição do sinal de entrada no problema da filtragem de Wiener e o desenvolvimento de uma correspondente expressão para a função custo J. O objetivo desse procedimento continuou o mesmo da filtragem de Wiener, isto é, obter um conjunto de coeficientes para o sistema de forma a minimizar a função custo J, ou em outras palavras, fazer a melhor estimação do sinal desejado d(n) em termos do critério da MSE.

Após a inclusão dessa decomposição e da minimização dessa função custo, foi desenvolvido ainda no Capítulo 3 um conjunto de equações denominadas de equações D ou sistema D, que fornece esses coeficientes em função das entradas decompostas e envolvidas no processo de minimização.

No Capítulo 4, a contribuição se deu na forma de implementações de diversas simulações relacionadas à predição linear de um processo autoregressivo de primeira ordem AR(1), empregando tanto as equações de Wiener-Hopf quanto as equações relativas ao sistema D. As soluções envolvendo a filtragem de Wiener-Hopf foram apresentadas para servirem de referência em relação aos resultados encontrados a partir do sistema de equações D.

As Figs.(4.5.a) e (4.6.a), referentes as equações de Wiener, apresentaram as variações dos coeficientes médios $\overline{h_{1_{ap}}^{f}}$ versus o número de amostras, considerando-se diferentes processos AR(1), um deles com a_1=-0.99 e o outro com a_1=-0.20 (sinal com pouca correlação). Para esses mesmos processos foram apresentados através das Figs.(4.7.a), (4.8.a), (4.9.a), (4.10.a), (4.11.a) e (4.12.a) gráficos com as variações dos coeficientes médios relativos ao sistema D e envolvendo os processos AR(1) anteriormente citados.

Constatou-se o mesmo comportamento dos coeficientes gerados pelas equações de Wiener, ou seja, convergiram para valores específicos à medida que aumentava o número de amostras utilizadas na estimação dos parâmetros estatísticos.

Pode-se observar através das Figs.(4.14.a), (4.14.b), (4.16.a) e (4.16.b), que tanto as curvas relativas a variância do erro de predição quanto as do erro relativo, associadas aos dois sistemas de equação, possuem um comportamento muito próximo em termos de velocidade de convergência. Verificou-se que em torno da qüinquagésima amostra elas praticamente coincidem.

Uma outra conclusão é que foi obtido o preditor ótimo baseando-se na solução das equações do sistema D, pois conforme introduzido no Capítulo 4, esse preditor deveria gerar uma variância do erro de predição igual a um. Isso foi obtido pelo preditor proposto considerando-se os dois processos autoregressivos de entrada, pois se comparada a variância do erro de predição gerado pelo preditor que utiliza os coeficientes de Wiener com a variância do preditor que utiliza os coeficientes do sistema D, as curvas relativas a esse erro e relacionadas a cada um dos sistemas, tenderam a uma sobreposição conforme apresentam as Figs.(4.7.b), (4.7.c), (4.8.b), (4.8.c), (4.9.b), (4.9.c), (4.10.b), (4.10.c), (4.11.b), (4.11.c), (4.12.b), (4.12.c), (4.13.a), (4.13.b), (4.14.a) e (4.14.b), (4.15.a), (4.15.b), (4.16.a) e (4.16.b).

No Capítulo 5 foi proposta uma estrutura de filtro adaptativo baseado no algoritmo LMS e que incorpora em si o processo de decomposição do sinal de entrada. A esse algoritmo denominou-se de algoritmo D-LMS.

A modificação da função custo J vista no Capítulo 3, para uma função custo J(n) variável com o tempo proposta no Capítulo 4, foi o primeiro passo para obtenção do algoritmo D-LMS. A minimização de J(n) expressa pela Eq.(5.32.d) ou Eq.(5.33.c) e sua estimativa instantânea empregada no algoritmo gradiente descendente Eq.(5.34) conduziu a uma expressão de atualização do vetor peso $\underline{\boldsymbol{\theta}}(n)$ dada pela Eq.(5.37), sendo que o passo de degrau variável expresso pela Eq.(5.38) presente nessa equação é o mesmo utilizado em algoritmos LMS no domínio transformado.

Uma variante desse algoritmo também foi proposta e a ela denominou-se de algoritmo DSTr-LMS. Nesse segundo algoritmo, aplicou-se a DST em cada um dos vetores decompostos $\underline{\mathbf{xr}}(n)$ e $\underline{\mathbf{r}}(n)$, resultando em um algoritmo com uma melhor taxa de convergência.

No Capítulo 6, a contribuição se deu na forma de implementações de diversas simulações envolvendo predição linear de processos AR(1) e AR(2), além da predição linear de sinais eletromiográficos. As simulações foram implementadas visando sempre fazer comparações de desempenho em relação aos resultados obtidos por algoritmos da mesma classe, ou seja, os algoritmos NLMS, DST-LMS e DCT-LMS.

As simulações de 1 a 6 utilizaram o algoritmo D-LMS em um problema de predição linear, onde os processos autoregressivos de entrada eram os mesmos apresentados nas simulações do Capítulo 4, ou seja: AR(1) com a_1=-0.99 e AR(1) com a_1=-0.20. Em função da entrada decomposta, obteve-se a convergência dos pesos para valores específicos, que dependiam da entrada decomposta utilizada pelo sistema.

As simulações de 7 a 11 referem-se a predição linear de 3 diferentes processos AR(2) denominados de AR1, AR2 e AR3. Cada um desses processos possui diferentes graus de correlação e conseqüentemente diferentes valores para o número de condição associado a matriz de correlação do sinal de entrada.

Os algoritmos adaptativos NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS geraram curvas de aprendizagem dadas pelas Figs.(6.8), (6.9), (6.10), (6.11) e (6.12). Elas estão relacionadas a cada um dos três processos AR(2), no esquema de predição linear apresentado na Fig.(6.1).

As Tabelas 1, 2 e 3 possuem parâmetros de desempenho extraídos dessas curvas, a saber: taxa de convergência, expressa através do limite inferior no intervalo I_3 e o erro de regime permanente expresso por $J(\infty)$, que retratam o bom desempenho dos algoritmos propostos em relação aos algoritmos tomados como referência.

As Tabelas 4, 5 e 6 foram obtidas de dados extraídos das curvas de aprendizagem dadas pelas Figs.(6.13), (6.14), (6.15), (6.16), (6.17), (6.18), (6.19), (6.20), (6.21) e (6.22), relacionadas, respectivamente, aos algoritmos Dr_0-LMS, Dr_1-LMS, Dr_2-LMS, Dr_3-LMS, Dr_{01}LMS, Dr_{02}-LMS, Dr_{03}-LMS, Dr_{12}-LMS, Dr_{13}-LMS e Dr_{23}-LMS.

Conforme explicado anteriormente, esses algoritmos foram obtidos através da eliminação de algumas entradas decompostas do D-LMS. As simulações mostraram que em alguns casos ocorreu uma melhora no erro de regime permanente, se comparados com os obtidos pelo algoritmo D-LMS.

Um procedimento análogo a esse, de eliminação de uma ou mais componentes dos vetores transformados $\underline{\mathbf{u}}_{DST}(n)$ expresso pela Eq.(5.46.c) gerou os algoritmos $DSTr_0$-LMS, $DSTr_1$-LMS, $DSTr_2$-LMS, $DSTr_3$-LMS, $DSTr_{01}$LMS, $DSTr_{02}$-LMS, $DSTr_{03}$-LMS, $DSTr_{12}$-LMS, $DSTr_{13}$-LMS e $DSTr_{23}$-LMS. Dizem respeito a esses algoritmos as curvas de aprendizagem fornecidas respectivamente pelas Figs.(6.23), (6.24), (6.25), (6.26), (6.27), (6.28), (6.29), (6.30), (6.31) e (6.32).

As Tabelas 7, 8 e 9 extraídas dessas curvas de aprendizagem demonstraram que em geral também houve uma melhoria no erro de regime permanente, se comparado com o erro do algoritmo DSTr-LMS.

De forma bastante sucinta, quando se compara o desempenho dos algoritmos NLMS, DST-LMS, DCT-LMS, D-LMS e suas variações, ou seja, Dr_0-LMS, Dr_1-LMS, Dr_2-LMS, Dr_3-LMS, Dr_{01}LMS, Dr_{02}-LMS, Dr_{03}-LMS, Dr_{12}-LMS, Dr_{13}-LMS e Dr_{23}-LMS, DSTr-LMS e suas variações, ou seja, $DSTr_0$-LMS, $DSTr_1$-LMS, $DSTr_2$-LMS, $DSTr_3$-LMS, $DSTr_{01}$LMS, $DSTr_{02}$-LMS, $DSTr_{03}$-LMS, $DSTr_{12}$-LMS, $DSTr_{13}$-LMS e $DSTr_{23}$-LMS, na predição linear dos processos AR(2) designados por AR1, AR2 e AR3, constatou-se:

-Para o processo AR1 que:

- O menor valor de erro de regime permanente foi obtido pelos algoritmos Dr_{02}-LMS e Dr_{13}-LMS, ambos com $\hat{J}(\infty) = 1.053$.
- A taxa de convergência mais rápida foi obtida pelo algoritmo DSTr-LMS.

-Para o processo AR2 que:

- O menores valores de erro de regime permanente foram obtidos pelos algoritmos Dr_{01}-LMS com $\hat{J}(\infty) = 1.053$ e Dr_{03}-LMS com $\hat{J}(\infty) = 1.058$.
- A taxa de convergência mais rápida foi obtida pelo algoritmo DSTr-LMS.

-Para o processo AR3 que:

- O menores valores de erro de regime permanente foram obtidos pelos algoritmos NLMS com $\hat{J}(\infty) = 1.030$ e Dr_{03}-LMS, Dr_{12}-LMS, Dr_{13}-LMS todos com $\hat{J}(\infty) = 1.050$.
- A taxa de convergência mais rápida foi obtida pelo algoritmo DSTr-LMS.

É importante salientar alguns comentários relativos a complexidade computacional dos algoritmos propostos. Nesse sentido, pode-se determinar a complexidade do algoritmo D-LMS e DSTr-LMS em função da complexidade dos algoritmos NLMS e DCT-LMS.

De forma aproximada, tem-se que a complexidade do algoritmo D-LMS situa-se em torno de duas vezes a do NLMS, uma vez que a Eq.(5.37) de atualização dos pesos possui duas vezes mais parâmetros para serem atualizados do que equação do NLMS. Por outro lado, deve haver uma redução nessa complexidade, pois não são atualizados os pesos relativos as entradas $xr_0(n)$ e $xr_{(N-1)}(n)$, por serem sempre entradas nulas. E finalmente, deve haver um incremento computacional, pois existe uma “carga” devido a necessidade de se fazer a decomposição dos sinais de entrada. Essas mesmas considerações devem valer para o algoritmo DSTr-LMS, pois se houve um aumento computacional pelo uso da transformada Discreta Seno no algoritmo DSTr-LMS, também houve esse aumento com relação ao algoritmo de referência DCT-LMS.

Cabem algumas observações relativas as Tabelas 10 e 11. Elas dizem respeito a predição linear de sinais EMG, considerando-se os seguintes algoritmos: NLMS, DST-LMS, DCT-LMS, D-LMS e DSTr-LMS. Os algoritmos que atingiram as melhores taxas de convergência foram o DST-LMS e o DSTr-LMS (valores muito próximos um do outro). No entanto, o erro de regime permanente alcançado pelo DSTr-LMS foi melhor do que o alcançado pelo DST-LMS.

Como sugestão de novas pesquisas, pode-se citar as seguintes:

-Emprego dos algoritmos D-LMS e DSTr-LMS na geração de coeficientes necessários para o reconhecimento, através da utilização de redes neurais, de anomalias presentes em sinais EMG.

-Investigação do emprego desses algoritmos na diminuição de ruídos presentes em imagens.

-Investigar nos algoritmos propostos a utilização de outras transformadas tais como Haar, Hadamard, Walsh-Hadamard, etc.

APÊNDICE - A

Equações das Transformadas DST-I e DCT-II

Transformada Discreta Seno do tipo I (DST-I) dada pela Eq.(5.40) [57, 58], sobre as amostras do vetor **xr**(n) para gerar o vetor transformado $\mathbf{xr}_{DST}(n)$.

$$S_N^I(m,n) = \sqrt{\frac{2}{N+1}}\,\mathbf{sin}\left(\frac{(m+1)(n+1)}{N+1}\,pi\right), \qquad m,\ n = 0,1,\cdots,N-1$$

onde $S_N^I(m,n)$ representa cada um dos elementos da matriz de transformação da DST-I de dimensão N x N, denotada por $\underline{\mathbf{S}}_N^I$. Portanto, os elementos $xr_{DST,m}(n)$ do vetor transformado $\mathbf{xr}_{DST}(n)$ serão calculados por:

$$xr_{DST,m}(n) = \sum_{t=0}^{N-1} S_N^I(m,t)\,xr_t(n), \qquad m = 0,1,\cdots,N-1$$

Transformada DCT do tipo II fornecida pela Eq.(6.5) [57, 58].

$$C_N^{II}(m,n) = \sqrt{\frac{2}{N}}\cdot k_m \cdot \mathbf{cos}\left(\frac{m\cdot(n+\frac{1}{2})}{N}\cdot\pi\right), \qquad m,\ n = 0,1,\cdots,N-1$$

$$\text{onde } k_m = \begin{cases} \dfrac{1}{\sqrt{2}} & ,\text{se } m = 0 \\ 1 & ,\text{se } m = 1 \end{cases}$$

onde, $C_N^{II}(m,n)$ representa cada um dos elementos da matriz de transformação da DCT-II de dimensão N x N, denotada por $\underline{\mathbf{C}}_N^{II}$. Portanto, dado $\underline{\mathbf{x}}(n) = \left[x_0(n) \quad x_1(n) \quad \ldots \quad x_{(N-1)}(n)\right]^T$ um vetor qualquer de N elementos, então, tem-se que os elementos $x_{DCT,m}(n)$ do vetor transformado $\underline{\mathbf{x}}_{DCT}(n)$ serão calculados por:

$$x_{DCT,m}(n) = \sum_{t=0}^{N-1} C_N^{II}(m,t)x_t(n), \qquad m = 0,1,...,N-1$$

APÊNDICE - B

Programas em Matlab

São listados basicamente três programas em Matlab. O primeiro programa é responsável pela geração das Figs(4.7.a), (4.7.b) e (4.7.c). Esse programa também é a base da geração das Figs(4.8.a.), (4.8.b), (4.8.c), (4.9.a), (4.9.b), (4.9.c), (4.10.a), (4.10.b), (4.10.c), (4.11.a), (4.11.b), (4.11.c), (4.12.a), (4.12.b), (4.12.c),(4.13),(4.14), (4.15) e (4.16).

O segundo programa e o terceiro são a base da geração dos gráficos relativos as curvas de aprendizagem esboçadas no Capítulo 6.

```
function
[vezes,N1,N2,passo,pot,a1,Offset,e_2_1,e_3_2,e_4_3,var_e_2_1,var_e_3_2,var_e_4
_3,m_var_e_2_1,...

m_var_e_3_2,m_var_e_4_3,var_noise,m_var_noise,noise,pesos_pp,saida,pesos_2_1,p
esos_3_2,pesos_4_3,m_pesos_2_1,...

m_pesos_3_2,m_pesos_4_3,m_er_2_1,m_er_3_2,m_er_4_3,m_pesos_1_4,m_var_e_1_4,m_e
r_1_4,m_pesos_4_4,m_var_e_4_4,m_er_4_4,...
    m_pesos_4_3a,m_var_e_4_3a,m_er_4_3a] = simula_4_1_bb;



t = cputime                                 ;
Offset = 2                                  ;
vezes = 20                                   ;

N1 = 10                                     ;
passo = 10                                   ;
N2 = N1 + 4000                              ;
N2 = N1 + 4000                              ;
N2_e = floor( (N2-N1+1)/passo  )            ;
Order       = 3                                    ;
pesos_1_4 = zeros(vezes,N2_e,3)              ;
pesos_2_1 = zeros(vezes,N2_e,3)              ;
pesos_3_2 = zeros(vezes,N2_e,3)              ;
pesos_4_3 = zeros(vezes,N2_e,3)              ;
pesos_4_3a = zeros(vezes,N2_e,3)              ;
pesos_4_4 = zeros(vezes,N2_e,4)              ;
xd        = zeros(1,N2)                         ;
x_hat     = zeros(1,N2)                         ;
e_1_4     = zeros(1,N2)                         ;
e_2_1     = zeros(1,N2)                         ;
e_3_2     = zeros(1,N2)                         ;
e_4_3     = zeros(1,N2)                         ;
e_4_3a     = zeros(1,N2)                         ;
e_4_4     = zeros(1,N2)                         ;
var_noise = zeros(vezes,N2_e)              ;
var_e_1_4 = zeros(vezes,N2_e)              ;
var_e_2_1 = zeros(vezes,N2_e)              ;
var_e_3_2 = zeros(vezes,N2_e)              ;
var_e_4_3 = zeros(vezes,N2_e)              ;
var_e_4_3a = zeros(vezes,N2_e)              ;
var_e_4_4 = zeros(vezes,N2_e)              ;
er_1_4    = zeros(vezes,N2_e)                ;
er_2_1    = zeros(vezes,N2_e)                ;
er_3_2    = zeros(vezes,N2_e)                ;
er_4_3    = zeros(vezes,N2_e)                ;
er_4_3a    = zeros(vezes,N2_e)                ;
er_4_4    = zeros(vezes,N2_e)                ;
m_var_noise = zeros(1,N2_e)                  ;
m_var_e_1_4 = zeros(1,N2_e)                  ;
m_var_e_2_1 = zeros(1,N2_e)                  ;
m_var_e_3_2 = zeros(1,N2_e)                 ;
```

```
m_var_e_4_3 = zeros(1,N2_e)                   ;
m_var_e_4_3a = zeros(1,N2_e)                   ;
m_var_e_4_4 = zeros(1,N2_e)                   ;
m_er_1_4    = zeros(1,N2_e)               ;
m_er_2_1    = zeros(1,N2_e)               ;
m_er_3_2    = zeros(1,N2_e)               ;
m_er_4_3    = zeros(1,N2_e)               ;
m_er_4_3a    = zeros(1,N2_e)               ;
m_er_4_4    = zeros(1,N2_e)               ;
m_pesos_1_4 = zeros(3,N2_e)                   ;
m_pesos_2_1 = zeros(3,N2_e)                   ;
m_pesos_3_2 = zeros(3,N2_e)                   ;
m_pesos_4_3 = zeros(3,N2_e)                   ;
m_pesos_4_3a = zeros(3,N2_e)                   ;
m_pesos_4_4 = zeros(4,N2_e)                   ;


for kk = 1:vezes,

    fprintf('kk = %8.2f\n',kk)
    i = 0;
    [noise,xd,x,xr, r, xd_xr, xd_r] = gera_sinais_6_in(N2,Order,kk) ;
    noise_100 = noise(1,N2)
    for N = N1:passo:N2,  %N e´ igual a numero de amostras (NoOfData)
        fprintf('kk = %8.2f\n',kk)
        fprintf('NoOfData = %8.2f\n',N)
        i = i + 1;
        if N == N2,
            noise_ = noise(1,N2);
        end
        W   = zeros(N,4)                ;
        W1 = W                                            ;
        sinal =   r(2:N,1)'.*r(2:N,1)'                    ;
        XX  =     mean(sinal)       ;
        sinal =   r(2:N,1)'.*r(2:N,2)'                    ;
        XY  =     mean(sinal)       ;
        sinal =   r(2:N,1)'.*r(2:N,3)'                    ;
        XZ  =     mean(sinal)       ;
        sinal =   r(2:N,2)'.*r(2:N,2)'                    ;
        YY  =     mean(sinal)       ;
        sinal =   r(2:N,2)'.*r(2:N,3)'                    ;
        YZ  =     mean(sinal)       ;
        sinal =   r(2:N,3)'.*r(2:N,3)'                    ;
        ZZ  =     mean(sinal)       ;
        sinal =   xr(2:N,2)'.*xr(2:N,2)'                  ;
        KK        =     mean(sinal)       ;
        sinal =   xr(2:N,2)'.*r(2:N,1)'                         ;
        KX        =     mean(sinal)       ;
        sinal =   xr(2:N,2)'.*r(2:N,2)'                         ;
        KY        =     mean(sinal)       ;
        sinal =   xr(2:N,2)'.*r(2:N,3)'                         ;
        KZ        =     mean(sinal)       ;
        sinal =   xd_xr(2:N,2)
                        ;
```

```
        Xd_K      =     mean(sinal)       ;
        sinal =   xd_r(2:N,1)
                        ;
        Xd_X      =     mean(sinal)       ;
        sinal =   xd_r(2:N,2)
                        ;
        Xd_Y      =     mean(sinal)       ;
        sinal =   xd_r(2:N,3)
                        ;
        Xd_Z      =     mean(sinal)       ;
        M = [ ( KX ) (KY )  (KZ )  (KK ) ; ...
                ( XX ) (XY )  (XZ )  (KX )      ; ...
                ( XY ) (YY )  (YZ )  (KY )      ; ...
                ( XZ ) (YZ )  (ZZ )  (KZ )]  ;
        P = [     Xd_K ; Xd_X  ;Xd_Y ; Xd_Z]    ;
        pesos_1_4(kk,i,:) = calc_pesos_1_4(Xd_X,Xd_Y,Xd_Z,M)        ;
        pesos_2_1(kk,i,:) = calc_pesos_2_1(Xd_K,Xd_Y,Xd_Z,M)        ;
        pesos_3_2(kk,i,:) = calc_pesos_3_2(Xd_K,Xd_X,Xd_Z,M)        ;
        pesos_4_3(kk,i,:) = calc_pesos_4_3(Xd_K,Xd_X,Xd_Y,M)        ;
        pesos_4_3a(kk,i,:) = calc_pesos_4_3a(Xd_K,Xd_X,Xd_Y,M)      ;
        pesos_4_4(kk,i,:) = calc_pesos_4_4(P,M)                     ;
        for n = 2:N,
            x_hat(1,n)= r(n,1)*pesos_1_4(kk,i,1)+ r(n,2)*pesos_1_4(kk,i,2) +
r(n,3)*pesos_1_4(kk,i,3) ;
            e_1_4(1,n)= xd(1,n)-x_hat(1,n)                    ;
        end
        for n = 2:N,
            x_hat(1,n)= r(n,2)*pesos_2_1(kk,i,1)+ r(n,3)*pesos_2_1(kk,i,2) +
xr(n,2)*pesos_2_1(kk,i,3) ;
            e_2_1(1,n)= xd(1,n)-x_hat(1,n)                    ;
        end
        for n = 2:N,
            x_hat(1,n)= r(n,1)*pesos_3_2(kk,i,1)+ r(n,3)*pesos_3_2(kk,i,2) +
xr(n,2)*pesos_3_2(kk,i,3) ;
            e_3_2(1,n)= xd(1,n)-x_hat(1,n)                    ;
        end
        for n = 2:N,
            x_hat(1,n)= r(n,1)*pesos_4_3(kk,i,1)+ r(n,2)*pesos_4_3(kk,i,2) +
xr(n,2)*pesos_4_3(kk,i,3) ;
            e_4_3(1,n)= xd(1,n)-x_hat(1,n)                    ;
        end
        for n = 2:N,
            x_hat(1,n)= r(n,1)*pesos_4_3a(kk,i,1)+ r(n,2)*pesos_4_3a(kk,i,2) +
xr(n,2)*pesos_4_3a(kk,i,3) ;
            e_4_3a(1,n)= xd(1,n)-x_hat(1,n)                    ;
        end
        for n = 2:N,
            x_hat(1,n)= r(n,1)*pesos_4_4(kk,i,1)+ r(n,2)*pesos_4_4(kk,i,2) +
r(n,3)*pesos_4_4(kk,i,3) + xr(n,2)*pesos_4_4(kk,i,4);
            e_4_4(1,n)= xd(1,n)-x_hat(1,n)                    ;
        end
        var_noise(kk,i) = var(noise(1,Offset:N))                    ;
        var_e_2_1(kk,i) = var(e_2_1(1,Offset:N))                    ;
        clear e_2_1 ;
```

```
        var_e_3_2(kk,i) = var(e_3_2(1,Offset:N))                    ;
        clear e_3_2 ;
        var_e_4_3(kk,i) = var(e_4_3(1,Offset:N))                    ;
        clear e_4_3 ;
        var_e_4_3a(kk,i) = var(e_4_3a(1,Offset:N))                    ;
        clear e_4_3a ;
        var_e_1_4(kk,i) = var(e_1_4(1,Offset:N))                    ;
        clear e_1_4 ;
        var_e_4_4(kk,i) = var(e_4_4(1,Offset:N))                    ;
        clear e_4_4 ;
        er_1_4(kk,i) = ( (var_e_1_4(kk,i) - var_noise(kk,i))/ var_noise(kk,i)
)    ;
        er_2_1(kk,i) = ( (var_e_2_1(kk,i) - var_noise(kk,i))/ var_noise(kk,i)
)    ;
        er_3_2(kk,i) = ( (var_e_3_2(kk,i) - var_noise(kk,i))/ var_noise(kk,i)
)    ;
        er_4_3(kk,i) = ( (var_e_4_3(kk,i) - var_noise(kk,i))/ var_noise(kk,i)
)    ;
        er_4_3a(kk,i) = ( (var_e_4_3a(kk,i) - var_noise(kk,i))/
var_noise(kk,i) )  ;
        er_4_4(kk,i) = ( (var_e_4_4(kk,i) - var_noise(kk,i))/ var_noise(kk,i)
)    ;
    end
end
[vezes,N,AR] = size(pesos_2_1)                                          ;
for i=1:AR, for j = 1:N, m_pesos_1_4(i,j)= mean(pesos_1_4(:,j,i));end;end
for i=1:AR, for j = 1:N, m_pesos_2_1(i,j)= mean(pesos_2_1(:,j,i));end;end
for i=1:AR, for j = 1:N, m_pesos_3_2(i,j)= mean(pesos_3_2(:,j,i));end;end
for i=1:AR, for j = 1:N, m_pesos_4_3(i,j)= mean(pesos_4_3(:,j,i));end;end
for i=1:AR, for j = 1:N, m_pesos_4_3a(i,j)= mean(pesos_4_3a(:,j,i));end;end
for i=1:(AR+1), for j = 1:N, m_pesos_4_4(i,j)= mean(pesos_4_4(:,j,i));end;end
[vezes,N] = size(var_e_2_1)    ;
for i = 1:N, m_var_noise(1,i)= mean(var_noise(:,i));end
for i = 1:N, m_var_e_1_4(1,i)= mean(var_e_1_4(:,i));end
for i = 1:N, m_var_e_2_1(1,i)= mean(var_e_2_1(:,i));end
for i = 1:N, m_var_e_3_2(1,i)= mean(var_e_3_2(:,i));end
for i = 1:N, m_var_e_4_3(1,i)= mean(var_e_4_3(:,i));end
for i = 1:N, m_var_e_4_3a(1,i)= mean(var_e_4_3a(:,i));end
for i = 1:N, m_var_e_4_4(1,i)= mean(var_e_4_4(:,i));end
for i = 1:N, m_er_1_4(1,i)= mean(er_1_4(:,i));end
for i = 1:N, m_er_2_1(1,i)= mean(er_2_1(:,i));end
for i = 1:N, m_er_3_2(1,i)= mean(er_3_2(:,i));end
for i = 1:N, m_er_4_3(1,i)= mean(er_4_3(:,i));end
for i = 1:N, m_er_4_3a(1,i)= mean(er_4_3a(:,i));end
for i = 1:N, m_er_4_4(1,i)= mean(er_4_4(:,i));end
tempo_de_processamento = (cputime-t)/3600 % O valor e´ dado em horas
figure(1);plot(m_pesos_1_4(1,:),'b');hold on;plot(m_pesos_1_4(2,:),'r');hold
on;plot(m_pesos_1_4(3,:),'k');grid on
figure(2);plot(m_pesos_2_1(1,:),'b');hold on;plot(m_pesos_2_1(2,:),'r');hold
on;plot(m_pesos_2_1(3,:),'k');grid on
figure(3);plot(m_pesos_3_2(1,:),'b');hold on;plot(m_pesos_3_2(2,:),'r');hold
on;plot(m_pesos_3_2(3,:),'k');grid on
figure(4);plot(m_pesos_4_3(1,:),'b');hold on;plot(m_pesos_4_3(2,:),'r');hold
on;plot(m_pesos_4_3(3,:),'k');grid on
```

```
figure(4);hold on;plot(m_pesos_4_3a(1,:),'g');hold
on;plot(m_pesos_4_3a(2,:),'g');hold on;plot(m_pesos_4_3a(3,:),'g');grid on
figure(5);plot(m_pesos_4_4(1,:),'b');hold on;plot(m_pesos_4_4(2,:),'r');hold
on;plot(m_pesos_4_4(3,:),'k');hold on;plot(m_pesos_4_4(4,:),'m');grid on
figure(6);plot(m_var_noise,'b');hold on;plot(m_var_e_1_4,'r');grid on
figure(7);plot(m_var_noise,'b');hold on;plot(m_var_e_2_1,'r');grid on
figure(8);plot(m_var_noise,'b');hold on;plot(m_var_e_3_2,'r');grid on
figure(9);plot(m_var_noise,'b');hold on;plot(m_var_e_4_3,'r');grid on
figure(9);hold on;plot(m_var_noise,'b');hold on;plot(m_var_e_4_3a,'g');grid on
figure(10);plot(m_var_noise,'b');hold on;plot(m_var_e_4_4,'r');grid on
figure(11);plot(m_er_1_4,'b');grid on
figure(12);plot(m_er_2_1,'b');grid on
figure(13);plot(m_er_3_2,'b');grid on
figure(14);plot(m_er_4_3,'b');grid on
figure(15);hold on;plot(m_er_4_3a,'g');grid on
figure(16);plot(m_er_4_4,'b');grid on

function [noise,xd,x,xr, r, xd_xr, xd_r] = gera_sinais_6_in(NoOfData,Order,kk)
;

[noise,x,xd,xr,r,xd_xr,xd_r] = identifica_input1_gera_in(NoOfData,Order,kk)  ;


%============================================================================
function [noise,x,xd,xr,r,xd_xr,xd_r] =
identifica_input1_gera_in(NoOfData,Order,kk)

State = 12024*kk
                  ;
%State = 1024
                  ;
pot = 1;
randn('state',State)
            ;
noise = sqrt(pot)*randn(1,NoOfData)
            ;
%alfa       =     0.75
                        ;
%h1   =     [1 -alfa]
            ;
%x          =     filter(sqrt(1 - alfa^2),h1,noise)              ;
%h1      =   [1 -0.9 0.2]
h1      =   [1 -0.9 0.2];
h1      =   [1 -0.99]                                    ;
h1      =   [1 -0.2]                                    ;
x           =     filter(1,h1,noise)                 ;

%Planta a ser reconhecida (problema de modelagem)
%h2 = [1 -4 ]
                  ;

%Sinal desejado xd(n)
%xd = filter(h2,1,x)
            ;
```

```
xd = x;
[i] = size(x,2);
x  = [0 xd(1,1:(i-1))];

% D-LMS Adaptation;
for n = Order : NoOfData  %O valor de n vaira de um em um, a partir de Order
    D = x(1,n:-1:n-Order+1)                                             ;
    xsegm(1,1:Order)    =     x(1,n:-1:n-Order+1)                 ;
    nn                  =     size(xsegm,2)
            ;
    yint          =     zeros(1,nn)
      ;
    yout    =  zeros(1,nn)                                                    ;
    Dinterp =  zeros(1,Order)                                           ;
    if nn > 1
        alfa            = ( xsegm(1)-xsegm(nn) ) / (nn -1)  ;     %jjjjjk
    else
        alfa = 0;
    end
    for i=1:nn,
        yint(i)    =     alfa*(i-1) + xsegm(nn)              ;     %jjkjj
    end
    yint(1,1:nn) = yint(1,nn:-1:1)                    ;     %jkjkjj
    % Sinal diferença
    yout          =     xsegm -     yint                                ;
    Dinterp(1,1:Order)  =     yout(1,1:Order)   ;
    D1 = Dinterp                                                        ;

    D2 = yint(1,1:Order)                                                ;
    %Vai ser usado para o calculo das estatísticas
    %=============================================
    r(n,:)                         =  D2
            ;
    xr(n,:)             =  D1                                                 ;
    xd_r(n,:)           =  xd(n)* D2                                    ;
    xd_xr(n,:)          =  xd(n)* D1                                    ;
    %=============================================
end
function inversos_M = inv_pesos_menores(M,P);

[lin,col] = size(M);
for k = 1:lin,
    for kk = 1:col,
        if (   ( (k ==2) &  (kk ==1) ) | ( (k ==3) &  (kk ==2) ) | ( (k ==4) &
(kk ==3) )  ),
            M_menor = menores(M,k,kk);
            Pf = [] ;
            for f = 1:col,
                if f ~= k,
                    Pf = [Pf ;P(f) ];
                end
            end
            k_ = k;
            kk_ = kk;
```

```
            cond_M = cond(M_menor);
            inversos_M = inv(M_menor)*Pf;
        end

    end
end



function saida = calc_pesos1(Xd_X,Xd_Z,M) ;

MM                 = [M(2,1) M(2,3);M(4,1) M(4,3)]     ;
P_MM               = [   Xd_X  ;Xd_Z]                           ;

valores          =  bicg(MM,P_MM)                            ;
saida(1,:)  =     valores'                                         ;

function saida = calc_pesos2(Xd_X,Xd_Z,M) ;

MM                 = [M(2,1) M(2,3);M(4,1) M(4,3)]     ;
P_MM               = [   Xd_X  ;Xd_Z]                           ;
valores          =  bicg(MM,P_MM)                            ;
saida(1,:)  =     valores'                                  ;


function saida = calc_pesos_1_4(Xd_X,Xd_Y,Xd_Z,M) ;

MM                 = M(2:4,1:3)      ;
P_MM               = [Xd_X ;   Xd_Y  ;Xd_Z]                           ;
valores          =  bicg(MM,P_MM)                            ;
saida(1,:)  =     valores'                            ;

function saida = calc_pesos_2_1(Xd_K,Xd_Y,Xd_Z,M) ;

MM                 = [M(1,2) M(1,3)M(1,4);  M(3,2) M(3,3) M(3,4) ;M(4,2) M(4,3)
M(4,4)]     ;
P_MM               = [Xd_K ;   Xd_Y  ;Xd_Z]                           ;

valores          =  bicg(MM,P_MM)                            ;
saida(1,:)  =     valores'                            ;

function saida = calc_pesos_3_2(Xd_K,Xd_X,Xd_Z,M) ;

MM                 = [M(1,1) M(1,3)M(1,4);  M(2,1) M(2,3) M(2,4) ;M(4,1) M(4,3)
M(4,4)]     ;
P_MM               = [Xd_K ;   Xd_X  ;Xd_Z]                           ;

valores          =  bicg(MM,P_MM)                            ;
saida(1,:)  =     valores'                            ;

function saida = calc_pesos_4_3(Xd_K,Xd_X,Xd_Y,M) ;

MM                 = [M(1,1) M(1,2) M(1,4);  M(2,1) M(2,2) M(2,4) ;M(3,1)
M(3,2) M(3,4)]    ;
```

```
P_MM              = [Xd_K ;   Xd_X  ;Xd_Y]                            ;

valores          =  bicg(MM,P_MM)                               ;
saida(1,:)  =     valores'                               ;

function saida = calc_pesos_4_3a(Xd_K,Xd_X,Xd_Y,M) ;

MM                = [M(1,1) M(1,2) M(1,4);  M(2,1) M(2,2) M(2,4) ;M(3,1)
M(3,2) M(3,4)]    ;
P_MM              = [Xd_K ;   Xd_X  ;Xd_Y]                            ;
inversa_MM  =     inv(MM)                                             ;
valores           =  inversa_MM * P_MM                         ;

saida(1,:)  =     valores'                               ;

function saida = calc_pesos_4_4(P,M) ;

MM                    = M                 ;
P_MM              = P                         ;
[valores]          =  bicg(MM,P_MM)                             ;
saida(1,:)  =     valores'                               ;
```

```
function intervalo_mio(curva_string) ;
%Exemplo
%curva_string = 'curva_NLMS_AR1_02' ; intervalo_mio(curva_string) ;

%Calcula o Jrp que se refere ao intervalo I1, o J que se refere a I2, o Jrp*1.1 e o respectivo
intervalo I3 e finalmente
%calcula tambem o Jmin*2 que e´ relativo ao intervalo I4
%Calcula o intervalo_I4, isto e´, o primeiro intervalo, a partir do valor de pico da curva de
aprendizagem, que resulta em 2*Jmin


%load curvas_AR(2)_8000_auto.mat
%load todos_mio_resample.mat ;
%load todos_mio_vezes_20_resample.mat ;
%load todos_mio_vezes_30_resample.mat ;
%load individual_mio_resample.mat
load todos_mio_resample.mat  ;


curva = eval( curva_string )               ;


if (size(curva,2)) == 7999,
   I1 = [7800  7999]                  ;
   I2 = [7000  7199]                  ;
elseif (size(curva,2)) == 11999
   I1 = [11800  11999]                ;
   I2 = [11000  11199]                ;
elseif (size(curva,2)) == 3998
   I1 = [3870  3970]                  ;
   I2 = [3770  3970]                  ;
elseif (size(curva,2)) == 1998
   I1 = [1870  1970]                  ;
   I2 = [1770  1970]                  ;
else (size(curva,2)) == 998
   I1 = [948  998]                    ;
   I2 = [900  948]                    ;
end


fprintf('====================================\n');
fprintf('\n')       ;
fprintf(curva_string)      ;
fprintf('\n \n');
```

```
[i,j]  = size(curva)                              ;
[referencia,indice]= max(curva)                            ;

controlI3  = 0                                   ;
controlI4  = 0                                   ;


fprintf('\n');
fprintf('Valor de pico  = %1.5f\n',referencia);
%fprintf('Valor de pico  = %1.10f\n',referencia);
fprintf('Indice         = %8d\n',indice);


fprintf('\n');
fprintf('I1     = %1.5f',I1(1));
fprintf('  -----  %1.5f\n',I1(2));
Jrp = mean( curva( 1,I1(1):I1(2) ) )  ;
fprintf('I1_Jrp = %1.5f\n',Jrp);

fprintf('\n');
fprintf('I2     = %1.5f',I2(1));
fprintf('  -----  %1.5f\n',I2(2));
fprintf('I2_J   = %1.5f\n',mean( curva( 1,I2(1):I2(2) ) ) )     ;
J10     = 1.1*Jrp                                ;


for k = indice:(j-100),
   saida_I3 = mean(curva(1,(k:(k+100))))                   ;

   if  ( saida_I3 <= J10 ) & ( controlI3 == 0 ), %Determina o primeiro intervalo que resulta em
uma valor menor do que J10
       I3 = [k k+100]                            ;
       saida_3 = saida_I3                          ;
       controlI3 = 1                             ;

   end


end
```

```
fprintf('\n');
fprintf('I3       = %1.5f',I3(1))         ;
fprintf('  -----  %1.5f\n',I3(2) )         ;
%fprintf('\n');
%fprintf('J10    =  %1.4f\n',J10);
fprintf('I3_J10    = %1.5f\n',saida_3)      ;
%Valor obtido pela media tem que ser igual ou MENOR do que J10!!
```

function [saida1] = Learn_rotLmsa_rxx_mio_interpo(vezes,NoOfData_1,Order,Mu,Beta,inter,r) ;
%[saida_rotLmsa_4_02_90_3_r] = Learn_rotLmsa_rxx_mio_interpo(20,1000,4,0.02,0.90,3,'r') ;

% inter = 2 ; inter e´ o parametro de interpolaçao. Se inter = 1 ===> nao existe interpolaçao. Caso contrario existira´.
% Onde r pode ser igual a 'r23' ou 'r34', etc.

% Onde r pode ser igual a 'r23' ou 'r34', etc.
%[saida_dstr] = Learndstr_mio(20,600) ;
% 1 < vezes < 40 (vezes indica quantos sinais serão utilizados em cada classe
% de movimento. Se vezes = 39 ++> W1000_extensao39.txt)
%
% 1 < NoOfData < 1000 (NoOfData indica o número de amostras de cada sinal)

```
V = 1                                          ;
for ar = 5:5,
   for c = 1:4,
      for i = 1: vezes,
         z = i
         if c ==1,
            eval(['load W1000_extensao' num2str(z) 'a.txt'])              ;
            saida = eval(['W1000_extensao' num2str(z) 'a'])                      ;
         elseif c == 2,
            eval(['load W1000_flexao' num2str(z) 'a.txt'])                       ;
            saida = eval(['W1000_flexao' num2str(z) 'a'])                        ;
         elseif c == 3,
            eval(['load W1000_pronacao' num2str(z) 'a.txt'])              ;
            saida = eval(['W1000_pronacao' num2str(z) 'a'])                      ;
         elseif c ==4,
            eval(['load W1000_supinacao' num2str(z) 'a.txt'])                    ;
            saida = eval(['W1000_supinacao' num2str(z) 'a'])                     ;
         end



         pares = 5
                                                         ;
         %pares = 1
```

```
        %ar    = 6
                                                                    ;

        for k = 1:pares,


            sinal = saida(k,1:NoOfData_1)                                  ;
            if inter == 1,
                saida_i = sinal ;
            else
                %saida_i = interp(sinal,inter)  ;
                saida_i = resample(sinal,inter,1)  ;
            end
            NoOfData = size(saida_i,2)                                ;
            xd = saida_i(1,2:NoOfData)
                                        ;
            x  = saida_i(1,1:NoOfData-1)
                                    ;


            %              xd = saida(k,2:NoOfData)
                                            ;
            %              x  = saida(k,1:NoOfData-1)
                                            ;
            %[xd_hat_vet,e_vet,w_vet] = Lmsa(x,xd,ar,0.001)                              ;
            %[xd_hat_vet,e_vet,w_vet]  = Dstr7Lmsa(x,xd, 4,0.01)              ;
            %[xd_hat_vet,e_vet,w_vet]  = Dst6Lmsa(x,xd, 4,0.01)               ;

            %[xd_hat_vet,e_vet,w_vet]  = rotLmsa_6_in(x,xd, 4,0.02,0.90,NoOfData,'r23')
                             ;
            [xd_hat_vet,e_vet,w_vet]  = rotLmsa_6_in(x,xd, Order,Mu,Beta,NoOfData,r)
                    ;

            E(V,:) = e_vet
                                                ;

            if k < pares,
                V = V + 1
                                            ;
            end

        end

    end
end
```

```
end


V

%Calcula a variancia de cada coluna da matriz E

for k =1:V,
   for n = 1:NoOfData-2,%novo
      E2(k,n) = E(k,n)^2  ;
   end
end

for n =1:NoOfData-2,%novo
   Learn(1,n) = mean(E2(:,n))           ;
end

saida1 = Learn          ;




%==========================================================



function [xd_hat_vet,e_vet,w_vet1,w_vet2,grad1,xx,x1] =
rotLmsa_6_in(x,xd,Order,Mu,Beta,NoOfData,r)

% Inicializa o LMS

% [xd1,x1,xr1, r1, xd_xr1, xd_r1] = gera_sinais_6(NoOfData,Order) ;
% grad1 = zeros(1,NoOfData);


NoOfData = size(x,2)                    ;

w = zeros(1,Order)                      ;

Mu_i1 = zeros(1,Order)                          ;

Mu_i2 = zeros(1,Order)                          ;


P1                 = zeros(1,Order)                 ;
```

```
P2              = zeros(1,Order)              ;

w_vet1 = zeros(NoOfData,Order)    ;

w_vet2 = zeros(NoOfData,Order)    ;

Beta1 = Beta          ;
Beta2 = Beta          ;

% Beta1 = 0.9                       ;
% Beta2 = 0.9                       ;
% Beta1 = 0.95                           ;
% Beta2 = 0.95                           ;
% Beta1 = 0.99                           ;
% Beta2 = 0.99                           ;

% LMS Adaptation

for n = Order : NoOfData
   %n_ = n
   D = x(1,n:-1:n-Order+1)                          ;
   xsegm(1,1:Order)  =        x(1,n:-1:n-Order+1)   ;

   nn                =        size(xsegm,2)                  ;
   yint          =        zeros(1,nn)                        ;
   yout          =  zeros(1,nn)                              ;
   Dinterp       =  zeros(1,Order)                                   ;
   if nn > 1
      alfa               = ( xsegm(1)-xsegm(nn) ) / (nn -1)   ;          %jjjjjk
   else
      alfa = 0;
   end


   for i=1:nn,
      yint(i)     =        alfa*(i-1) + xsegm(nn)                    ;          %jjkjj
   end
   yint(1,1:nn) = yint(1,nn:-1:1)                            ;          %jkjkjj
   % Sinal diferença
   yout          =        xsegm -        yint                        ;
   Dinterp(1,1:Order) =        yout(1,1:Order)         ;

   D1 = Dinterp                                                      ;
```

```
D2 = yint(1,1:Order)                                        ;


%    D1 = dst(D1)                                           ;
%    D2 = dst(D2)                                           ;
%
%    D1 = dstI(D1)                                          ; %Rotina que calcula a DSTI
%    D2 = dstI(D2)                                          ;

xd_hat_vet1(1,n) = w_vet1(n,:)*D1'        ;%Cria o vetor d_hat_vet(n) para
                                          %armazenar os "n" valores de d_hat

xd_hat_vet2(1,n) = w_vet2(n,:)*D2'                                        ;

e_vet(1,n) = xd(1,n)- xd_hat_vet1(1,n) - xd_hat_vet2(1,n)           ;

if n > 700,
   %xd(1,n)
   %soma = xd_hat_vet1(1,n) + xd_hat_vet2(1,n);
   %soma

   %dif = xd(1,n) - med;
   %dif
   %xd_hat_vet1(1,n)
   %xd_hat_vet2(1,n)
   %med
   %e_vet(1,n)

   %keyboard
end


for i = 1:Order,

   P1(1,i) = Beta1 * P1(1,i) + (1-Beta1)* (abs( D1(1,i) )) ^ 2     ;
   P2(1,i) = Beta2 * P2(1,i) + (1-Beta2)* (abs( D2(1,i) )) ^ 2     ;


   %Mu_i1(1,i) = Mu / P1(1,i)
                ;
   %Mu_i1(1,i) = Mu / ( P1(1,i) + 0.0000000001 )
;
   %Mu_i2(1,i) = Mu / ( P2(1,i) + 0.0000000001 )
;
```

```
    %      Mu_i1(1,i) = Mu / ( P1(1,i) + 0.000001 )
;
    %      Mu_i2(1,i) = Mu / ( P2(1,i) + 0.000001 )
;
    Mu_i1(1,i) = Mu / ( P1(1,i) + 0.0001 )                              ;
    Mu_i2(1,i) = Mu / ( P2(1,i) + 0.0001 )                              ;
    %Mu_i2(1,i) = Mu / ( P2(1,i)  )
          ;

  end

  for i = 1:Order,

    %      if (i ==22)|(i==33)  ,
    %          w_vet1(n+1,i)  = 0;
    %      else
    %          w_vet1(n+1,i) = w_vet1(n,i) + 2 * Mu_i1(1,i) * e_vet(1,n)* conj(D1(1,i));
    %      end

    w_vet1(n+1,i) = w_vet1(n,i) + Mu_i1(1,i) * e_vet(1,n)* conj(D1(1,i));


    %     if (i ==11)|(i ==2)| (i ==33)

    if strcmp(r,'r'),
       w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
    elseif strcmp(r,'r1'),
       if (i ==1)
          %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
          %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
          w_vet2(n+1,i) = 0;
       else
          w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
       end
    elseif strcmp(r,'r2'),
       if (i ==2)
          %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
          %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
          w_vet2(n+1,i) = 0;
       else
          w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
       end
    elseif strcmp(r,'r3'),
       if (i ==3)
          %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
```

```
    %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
    w_vet2(n+1,i) = 0;
  else
    w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
  end
elseif strcmp(r,'r4'),
  if (i ==4)
    %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
    %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
    w_vet2(n+1,i) = 0;
  else
    w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
  end
elseif strcmp(r,'r12'),
  if (i ==1) | (i ==2)
    %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
    %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
    w_vet2(n+1,i) = 0;
  else
    w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
  end
elseif strcmp(r,'r13'),
  if (i ==1) | (i ==3)
    %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
    %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
    w_vet2(n+1,i) = 0;
  else
    w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
  end
elseif strcmp(r,'r14'),
  if (i ==1) | (i ==4)
    %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
    %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
    w_vet2(n+1,i) = 0;
  else
    w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
  end
elseif strcmp(r,'r23'),
  if (i ==2) | (i ==3)
    %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
    %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
    w_vet2(n+1,i) = 0;
  else
    w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
  end
```

```
        elseif strcmp(r,'r24'),
            if (i ==2) | (i ==4)
                %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
                %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
                w_vet2(n+1,i) = 0;
            else
                w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
            end
        elseif strcmp(r,'r34'),
            if (i ==3) | (i ==4)
                %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
                %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
                w_vet2(n+1,i) = 0;
            else
                w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
            end

        end



    end


end

xd_hat_vet = xd_hat_vet1 + xd_hat_vet2  ;



%============================================================
```

```
function [saida1] = Learn_dstr_rxx_mio_interpo(vezes,NoOfData_1,Order,Mu,Beta,inter,r) ;
%[saida_dstr_4_02_90_3_r] = Learn_dstr_rxx_mio_interpo(20,1000,4,0.02,0.90,3,'r') ;


% inter = 2 ; inter e´ o parametro de interpolaçao. Se inter = 1 ===> nao existe interpolaçao. Caso
contrario existira´.
% Onde r pode ser igual a 'r23' ou 'r34', etc.


%[saida_dstr] = Learndstr_mio(20,600) ;
%  1 <    vezes   < 40 (vezes indica quantos sinais serão utilizados em cada classe
%                  de movimento. Se vezes = 39 ++> W1000_extensao39.txt      )
%
%  1 <  NoOfData  < 1000  (NoOfData  indica o número de amostras de cada sinal)

V = 1                                              ;
for ar = 5:5,
   for c = 1:4,
      for i = 1: vezes,
         z = i
         if c ==1,
            eval(['load W1000_extensao' num2str(z) 'a.txt'])                          ;
            saida = eval(['W1000_extensao' num2str(z) 'a'])                                    ;
         elseif c == 2,
            eval(['load W1000_flexao' num2str(z) 'a.txt'])                                     ;
            saida = eval(['W1000_flexao' num2str(z) 'a'])                                      ;
         elseif c == 3,
            eval(['load W1000_pronacao' num2str(z) 'a.txt'])                          ;
            saida = eval(['W1000_pronacao' num2str(z) 'a'])                                    ;
         elseif c ==4,
            eval(['load W1000_supinacao' num2str(z) 'a.txt'])                                  ;
            saida = eval(['W1000_supinacao' num2str(z) 'a'])                                   ;
         end



         pares = 5
                                                              ;
         %pares = 1
         %ar    = 6
                                                              ;

         for k = 1:pares,

            sinal = saida(k,1:NoOfData_1)                          ;
```

```
        if inter == 1,
           saida_i = sinal ;
        else
           %saida_i = interp(sinal,inter)  ;
           saida_i = resample(sinal,inter,1)  ;
        end
        NoOfData = size(saida_i,2)                                    ;
        xd = saida_i(1,2:NoOfData)
                                  ;
        x  = saida_i(1,1:NoOfData-1)
                                ;


        %              xd = saida(k,2:NoOfData)
                                    ;
        %              x  = saida(k,1:NoOfData-1)
                                    ;
        %[xd_hat_vet,e_vet,w_vet] = Lmsa(x,xd,ar,0.001)                                        ;
        %[xd_hat_vet,e_vet,w_vet]  = Dstr7Lmsa(x,xd, 4,0.01)            ;
        %[xd_hat_vet,e_vet,w_vet]  = Dst6Lmsa(x,xd, 4,0.01)             ;

        %[xd_hat_vet,e_vet,w_vet]  = Dstr7Lmsa_in2(x,xd, 4,0.02,0.90,NoOfData,'r23')
                  ;
        %[xd_hat_vet,e_vet,w_vet]  = Dstr7Lmsa_in2(x,xd, 4,0.02,0.90,NoOfData,'r34')
                  ;
        [xd_hat_vet,e_vet,w_vet]  = Dstr7Lmsa_in2(x,xd, Order,Mu,Beta,NoOfData,r)
                  ;

        E(V,:) = e_vet
                                          ;

        if k < pares,
           V = V + 1
                                    ;
        end

      end

    end
  end
end


V
```

```
%Calcula a variancia de cada coluna da matriz E

for k =1:V,
   for n = 1:NoOfData-2,%novo
      E2(k,n) = E(k,n)^2  ;
   end
end

for n =1:NoOfData-2,%novo
   Learn(1,n) = mean(E2(:,n))          ;
end

saida1 = Learn          ;



%=====================================================================



function [xd_hat_vet,e_vet,w_vet1,w_vet2,grad1,xx,x1] =
Dstr7Lmsa_in2(x,xd,Order,Mu,Beta,NoOfData,r)

% Inicializa o LMS

% [xd1,x1,xr1, r1, xd_xr1, xd_r1] = gera_sinais_6(NoOfData,Order) ;
% grad1 = zeros(1,NoOfData);


NoOfData = size(x,2)                    ;

w = zeros(1,Order)                      ;

Mu_i1 = zeros(1,Order)                        ;

Mu_i2 = zeros(1,Order)                        ;


P1              = zeros(1,Order)              ;

P2              = zeros(1,Order)              ;


w_vet1 = zeros(NoOfData,Order)    ;
```

```
w_vet2 = zeros(NoOfData,Order)     ;

% Beta1 = 0.9                      ;
% Beta2 = 0.9                      ;
% Beta1 = 0.95                              ;
% Beta2 = 0.95                              ;
% Beta1 = 0.99                              ;
% Beta2 = 0.99                              ;
Beta1   =   Beta          ;
Beta2   =   Beta          ;

% LMS Adaptation

for n = Order : NoOfData
   %n_ = n
   D = x(1,n:-1:n-Order+1)                          ;
   xsegm(1,1:Order)  =        x(1,n:-1:n-Order+1)   ;

   nn                       =        size(xsegm,2)            ;
   yint           =        zeros(1,nn)                        ;
   yout           =  zeros(1,nn)                              ;
   Dinterp        =  zeros(1,Order)                                   ;
   if nn > 1
      alfa                  = ( xsegm(1)-xsegm(nn) ) / (nn -1)   ;        %jjjjjk
   else
      alfa = 0;
   end


   for i=1:nn,
      yint(i)     =        alfa*(i-1) + xsegm(nn)                        ;        %jjkjj
   end
   yint(1,1:nn) = yint(1,nn:-1:1)                        ;        %jkjkjj
   % Sinal diferença
   yout           =        xsegm -        yint                  ;
   Dinterp(1,1:Order) =        yout(1,1:Order)         ;

   D1 = Dinterp                                                  ;
   D2 = yint(1,1:Order)                                          ;


   %    D1 = dst(D1)                                              ;
   %    D2 = dst(D2)                                              ;
   %
```

```
D1 = dstI(D1)                                          ; %Rotina que calcula a DSTI
D2 = dstI(D2)                                          ;

xd_hat_vet1(1,n) = w_vet1(n,:)*D1'     ;%Cria o vetor d_hat_vet(n) para
                                        %armazenar os "n" valores de d_hat

xd_hat_vet2(1,n) = w_vet2(n,:)*D2'                                   ;

e_vet(1,n) = xd(1,n)- xd_hat_vet1(1,n) - xd_hat_vet2(1,n)            ;

if n > 700,
   %xd(1,n)
   %soma = xd_hat_vet1(1,n) + xd_hat_vet2(1,n);
   %soma

   %dif = xd(1,n) - med;
   %dif
   %xd_hat_vet1(1,n)
   %xd_hat_vet2(1,n)
   %med
   %e_vet(1,n)

   %keyboard
end


for i = 1:Order,

   P1(1,i) = Beta1 * P1(1,i) + (1-Beta1)* (abs( D1(1,i) )) ^ 2     ;
   P2(1,i) = Beta2 * P2(1,i) + (1-Beta2)* (abs( D2(1,i) )) ^ 2     ;


   %Mu_i1(1,i) = Mu / P1(1,i)
                    ;
   %Mu_i1(1,i) = Mu / ( P1(1,i) + 0.0000000001 )
;
   %Mu_i2(1,i) = Mu / ( P2(1,i) + 0.0000000001 )
;
   %      Mu_i1(1,i) = Mu / ( P1(1,i) + 0.000001 )
;
   %      Mu_i2(1,i) = Mu / ( P2(1,i) + 0.000001 )
;
   Mu_i1(1,i) = Mu / ( P1(1,i) + 0.0001 )                            ;
   Mu_i2(1,i) = Mu / ( P2(1,i) + 0.0001 )                            ;
```

```
    %Mu_i2(1,i) = Mu / ( P2(1,i)  )
        ;

  end

  for i = 1:Order,

    %      if (i ==22)|(i==33)  ,
    %         w_vet1(n+1,i)  = 0;
    %      else
    %         w_vet1(n+1,i) = w_vet1(n,i) + 2 * Mu_i1(1,i) * e_vet(1,n)* conj(D1(1,i));
    %      end

    w_vet1(n+1,i) = w_vet1(n,i) + Mu_i1(1,i) * e_vet(1,n)* conj(D1(1,i));

    %     if (i ==11)|(i ==2)| (i ==33)
    %       if (i ==11)|(i ==2)| (i ==33)| (i ==4)
    %           %      if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
    %           %      if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
    %           w_vet2(n+1,i) = 0;
    %       else
    %           %w_vet2(n+1,i) = w_vet2(n,i) + 2 * Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
    %                w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
    %       end


    if strcmp(r,'r'),
       w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
    elseif strcmp(r,'r1'),
       if (i ==1)
          %      if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
          %      if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
          w_vet2(n+1,i) = 0;
       else
          w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
       end
    elseif strcmp(r,'r2'),
       if (i ==2)
          %      if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
          %      if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
          w_vet2(n+1,i) = 0;
       else
          w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
       end
    elseif strcmp(r,'r3'),
```

```
    if (i ==3)
      %      if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
      %      if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
      w_vet2(n+1,i) = 0;
    else
      w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
    end
  elseif strcmp(r,'r4'),
    if (i ==4)
      %      if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
      %      if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
      w_vet2(n+1,i) = 0;
    else
      w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
    end
  elseif strcmp(r,'r12'),
    if (i ==1) | (i ==2)
      %      if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
      %      if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
      w_vet2(n+1,i) = 0;
    else
      w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
    end
  elseif strcmp(r,'r13'),
    if (i ==1) | (i ==3)
      %      if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
      %      if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
      w_vet2(n+1,i) = 0;
    else
      w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
    end
  elseif strcmp(r,'r14'),
    if (i ==1) | (i ==4)
      %      if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
      %      if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
      w_vet2(n+1,i) = 0;
    else
      w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
    end
  elseif strcmp(r,'r23'),
    if (i ==2) | (i ==3)
      %      if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
      %      if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
      w_vet2(n+1,i) = 0;
    else
```

```
            w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
         end
      elseif strcmp(r,'r24'),
         if (i ==2) | (i ==4)
            %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
            %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
            w_vet2(n+1,i) = 0;
         else
            w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
         end
      elseif strcmp(r,'r34'),
         if (i ==3) | (i ==4)
            %     if (i ==11)|(i ==2)| (i ==33)| (i ==4) | (i ==55)
            %     if (i ==11)|(i ==22)| (i ==33)| (i ==44) | (i ==55) ,
            w_vet2(n+1,i) = 0;
         else
            w_vet2(n+1,i) = w_vet2(n,i) + Mu_i2(1,i) * e_vet(1,n)* conj(D2(1,i));
         end
      else
         nada = 'nao faz nada'  ;

      end


   end



end

xd_hat_vet = xd_hat_vet1 + xd_hat_vet2  ;
```

Referências Bibliográficas

[1] Kuo, S. M., Y. C. Huang and Z. Pan, "Acoustic Noise and Echo Cancellation Microphone System for Videoconferencing," IEEE Trans. on Consumer Electronics, vol. 41, no. 4, pp. 1150-1158, Nov. 1995.

[2] Gilloire, A. and M. Vertterli, "Adaptive Filtering in Subbands with Critical Sampling: Analysis, Experiments, and Application to Acoustic Echo Cancellation," IEEE Trans. on Signal Processing, vol. 40, no. 8, pp. 1862-1875, Aug. 1992.

[3] Kuo, S. M. and D. R. Morgan, "Active Noise Control: A Tutorial Review," Proceedings of The IEEE, vol. 87, no. 6, pp. 943-973, Jun. 1999.

[4] Inoue, H. and A. Miyazaki, "Noise Reduction Method for ECG Signals Using the Dyadic Wavelet Transform," IEICE Trans. Fundamentals, vol. E81-A, no. 6, pp.1001-1006, Jun. 1998.

[5] Yelderman, M., B. Widrow, J. M. Cioffi, E. Hesler and J. A. Leddy, "ECG Enhancement by Adaptive Noise Cancellation of Electrosurgical Interference," IEEE Tans. Biomed. Eng. BME7, pp.392-398, 1983.

[6] Senhadji L., G. Carrault, J. J. Bellanger, and G. F. Passariella, "A Comparative Study of Wavelet Transforms for the Recognition of Cardiac Patterns", IEEE Engineering in Medicine and Biology Magazine, pp.167-173, Mar. 1995.

[7] Rabiner, L. R., "Digital Processing of Speech Signals," Prentice-Hall, 1978.

[8] "WSQ Fingerprint Image Compression Encoder/Decoder Certification Guidelines," http://www.itl.nist.gov/iad/894.03/fing/cert_gui.html , January 12, 1999

[9] Brislawn, C., "The FBI Fingerprint Image Compression Standard," http://www.c3.lanl.gov/~brislawn/FBI/FBI.html

[10] Rabiner, L. R. and B. Gold, "Theory and Application of Digital Signal Processing," Prentice-Hall, Englewood Cliffs, NJ, 1975.

[11] Peled, A. and B. Liu, "Digital Signal Processing: Theory, Design and Implementation," Wiley, New York, 1976.

[12] Jong, M. T., "Methods of Discrete Signal and System Analysis," McGraw-Hill, New York, 1982.

[13] Stanley, W. D., G. R. Dougherty and R. Dougherty, “Digital Signal Processing,” Prentice-Hall, 2º ed., 1984.

[14] Kuc, R., “Introduction to Digital Signal Processing,” McGraw-Hill, New York, 1982.

[15] Oppenheim, A. V., and R. W. Schafer, “Discrete-Time Signal Processing,” Prentice-Hall, 1989.

[16] Ifeachor, E. C. and B. W. Jervis, “Digital Signal Processing: A Practical Approach,” Addison-Wesley, N.Y., 1993.

[17] Honig, M. L. and D. G. Messerschmitt, “Adaptive Filters: Structures, Algorithms and Applications,” Kluwer Academic, London, 1984.

[18] Widrow, B. and S. D. Stearns, “Adaptive Signal Processing,” Prentice-Hall, Englewood Cliffs, N.J., 1985.

[19] Haykin, S., “Adaptive Filter Theory,” Prentice Hall, N.J., third ed., 1996.

[20] Widrow, B. and M. E. Hoff, JR., “Adaptive Switching Circuits,” IRE WESCON Conv. Rec., part 4, pp.96-104, 1960.

[21] Widrow, B., J. McCool and M. Ball, “The Complex LMS Algorithm,” Proc. IEEE, vol. 63, pp. 719-720, Apr. 1975.

[22] Shynk, J. J., “Adaptive IIR Filtering,” IEEE ASSP Mag., vol. 6, pp. 4-21,1989.

[23] Dentino, M., J. McCool, and B. Widrow, “Adaptive Filtering in the Frequency Domain,” Proc. IEEE, vol.66, pp. 1658-1659, Dec. 1978.

[24] Ferrara, E. R., JR., “Fast Implementation of LMS Adaptive Filters,” IEEE Trans. Acoustics, Speech and Signal Process., vol. ASSP-28, pp. 474-475, Aug. 1980.

[25] Cowan, C. F. N. and P. M. Grant, “Adaptive Filters,” Prentice-Hall, Englewood Cliffs, N. J., 1985.

[26] Clark, G. A., S. K. Mitra, and S. R. Parker, “Block Implementation of Adaptive Digital Filters,” IEEE Trans. Circuits and Syst., vol. CAS-28, pp. 584-592, 1981.

[27] Shynk, J. J., “Frequency-domain and Multirate Adaptive Filtering,” IEEE Signal Process. Mag., vol. 9, no. 1, pp. 14-37, Jan. 1992.

[28] Widrow, B., J. M. McCool, M. G. Larimore and C. R. Johnson, JR., “Stationary and Nonstationary Learning Characteristics of the LMS Adaptive Filter,” Proc. IEEE, vol. 64, no. 8, pp. 1151-1162, Aug. 1976.

[29] Narayan, S. S. and A. M. Peterson, “Frequency Domain Least-Mean-Square Algorithm”, Proc. IEEE, vol. 69, no. 1, pp. 124-126, Jan. 1981.

[30] Narayan, S. S., A. M. Peterson and M. J. Narasimha, “Transform Domain LMS Algorithm,” IEEE Trans. Acoustics, Speech and Signal Process., vol. ASSP-31, no. 3, pp. 609-615, June 1983.

[31] Lee, J. C. and C. K. Un, “Performance of Transform-Domain LMS Adaptive Digital Filters,” IEEE Trans. Acoustics, Speech and Signal Process., vol. ASSP-34, no. 3, pp. 499-510, June 1986.

[32] Marshall, D. F., W. K. Jenkins and J. J. Murphy, “The Use of Orthogonal Transforms for Improving Performance of Adaptive Filters,” IEEE Trans. Circuits and Syst., vol. CAS-36, no.4, pp. 474-483, Ap. 1989.

[33] Farhang-Boroujeny, B. and S. Gazor, “Performance Analysis of Transform Domain Normalized LMS Algorithm,” in Proc. ICASSP’91 Conf., Toronto, Canada, pp. 2133-2136.

[34] Farhang-Boroujeny, B. and S. Gazor, “Selection of Orthonormal Transforms for Improving the Performance of the Transform Domain Normalised LMS Algorithm,” IEE Proceedings, part F, vol. 139, no. 5, pp. 327-335, Oct. 1992.

[35] Beaufays, F. and B. Widrow, “Transform Domain Adaptive Filters: An Analytical Approach,” IEEE Trans. On Signal Processing, vol. 43, no.2, Feb. 1995.

[36] Farhang-Boroujeny, B., Y. Lee and C. C. Ko, “Sliding Transforms for Efficient Implementation of Transform Domain Adaptive Filters,” Elsevier Signal Processing, 52, pp. 83-96, 1996.

[37] Kim, D. I. and P. D. Wilde, “Performance Analysis of the DCT-LMS Adaptive Filtering Algorithm,” Elesevier Signal Processing, 80, pp. 1628-1654, 2000.

[38] Haykin, S., “Modern Filters,” Macmillan, N. Y., 1989.

[39] Hayes, M. H., “Statistical Digital Signal Processing and Modeling,” John Wiley & Sons, N.Y., 1996.

[40] Zelniker, G. and F. J. Taylor, “Advanced Digital Signal Processing: Theory and Applications,” Marcel Dekker, N. Y., 1994.

[41] Araújo, A. M. L. and A. M. Faria JR., “Interpolação pela Transformada Discreta em Seno com pré e pós-Rotação,” XIV SBT, Curitiba-PR, Sept. 1996.

[42] Pelaes, E. G. and Y. Iano, "Image coding with low Blocking Effects Using Discrete Sine with Axis Rotation", Proc of XV International Telecommunication Symposium – ITS98, São Paulo-SP, Brazil, Aug. 1998.

[43] Pelaes, E. G., "Transformada Seno Discreta com Rotação de Eixos Bidimensional (DSTr-2D): Aplicações na Codificação e Interpolação de Imagens para Redução de Efeito de Blocos," UNICAMP, Tese de doutorado, 1998.

[44] Pelaes, E. G. and Y. Iano, "Image coding Using Discrete Sine Transform with Axis Rotation," IEEE Transactions on Consumer Electronics, vol.44, pp. 1284-1290, Nov. 1998.

[45] A.C.P. Veiga e Y. Iano, "Adaptive Filtering in Frequency Domain Through the Use of Discrete Sine Transform With Axis Rotation", International Conference on Signal Processing Applications & Technology, Dallas, EUA, *http://www.icspat.com/, 2000.*

[46] A.C.P. Veiga, Y. Iano, L.V. Lima e A.B. Soares, "Electromyographic Signal Modelling by The Use of Discrete Sine Transform With Axis Rotation Adaptive Filter", Proc. of the 10th International Conference on Biomedical Engineering, Singapura, 143-144, 2000.

[47] A.C.P. Veiga, Y. Iano e G.A. Carrijo, "A Comparative Performance of DSTr Adaptive Filter With NLMS, DST-LMS, DCT-LMS and RLS Algorithms", Proc. of The IASTED International Conference Modelling, Identification, and Control, Innsbruck, Austria, vol. II, 818-823, 2001.

[48] A.C.P. Veiga, Y. Iano, L.V. Lima and K. Yamanaca, "Insertion of the Discrete Sine Transform with Pre-Rotation in the LMS Algorithm", submetido e aprovado para o V INDUSCON - IEEE - Industry Applications Society, 2002.

[49] A.C.P. Veiga, G.A.Carrijo e Y. Iano, "Electromyographic Signal Modelling Using an Adaptative Filter: Comparative Study of DST-LMS, DCT-LMS and DSTr-LMS", World Multiconference on Systemics, Cybernetics and Informatics, Florida, EUA, vol.VI, 425-429, 2001.

[50] A.C.P. Veiga, Y. Iano e G.A.Carrijo, "A New Adaptive Filter Structure: Comparative Study of NLMS, DST-LMS and DCT-LMS Schemes Applied to Electromyographic Signal Modelling", 2001 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, Victoria, Canada, vol. II, 555-558, 2001.

[51] Benedetto S., Biglieri E. and Castellari V., “Digital Transmission Theory,” Prentice-Hall, 1987.

[52] Boulos, P. e I. Camargo, “Geometria Analítica: Um Tratamento Vetorial,” McGraw-Hill, 1987.

[53] Valladares, R. J. C., “Geometria Analítica: Do Plano e do Espaço,” Livros Técnicos e Científicos, 1990.

[54] Akay, M., “Biomedical Signal Processing,” Academic Press, N. Y., 1994.

[55] Kay, M. Steven, “Modern Spectral Estimation: Theory and Application,” Prentice Hall, Englewood Cliffs, N. J., 1988.

[56] Farhang-Boroujeny, B., “Adaptive Filters: Theory and Applications,” John Wiley & Sons, N. Y., 1998.

[57] Yip, P. and Rao, K. R., “On the Shift Property of DCT’s and DST’s,” IEEE Trans. Acoustics, Speech and Signal Process., vol. ASSP-35, no. 3, pp. 404-406, Mar. 1987.

[58] Clarke, R. J., “Transform Coding of Images,” Academic Press, N. Y., 1985.

[59] Grande Enciclopédia Larousse Cultural, vol. I, pp. 3251 e 3253, 1998.

[60] Andrade, A. O., “Metodologia para Classificação de Sinais EMG no Controle de Membros Artificiais,” UFU, Tese de Mestrado, 2000.

[61] Hefftner, G., Walter Z. and George G. J., “The Electromyogram (EMG) as a Control Signal for Functional Neuromuscular Stimulation-Part I: Autoregressive Modeling as a Means of EMG Signature Discrimination,” IEEE Trans. on Biomedical Engineering, vol.35, no. 4, pp 230-236, Apr. 1988.

[62] Xiong, F. Q. and E. Shwedyk, “Some Aspects of Nonstationary Myoelectric Signal Processing,” IEEE Trans. on Biomedical Engineering, vol. BME-34, no. 2, Feb. 1987.

LISTA DE PUBLICAÇÕES:

1. Veiga, A. C. P., E. G. Pelaes, Y. Iano and C. Bezzan, “HDTV: The New Perspectives of Television in Brazil,” International Electronic Engineering Conference INTERCOM, vol. I, pp. 37-42 , Peru, 1996.

2. Pelaes, E. G., A. C. P. Veiga, Y. Iano and E. Ahmed, “HDTV Transmission in Brazil: COFDM or 8-VSB,” International Electronic Engineering Conference INTERCOM, vol. I, pp. 52-57, Peru, 1996.

3. Silva, C. A. L., A. C. P. Veiga, G. A. Carrijo, Y. Iano, “Use of Fast Wavelet and Fourier Transforms in Musical Notes in Polyphonics Wave Files,” Conference on Computational and Mathematical Methods in Music DIDEROT’99, Austria, 1999.

4. A.C.P. Veiga e Y. Iano, “Adaptive Filtering in Frequency Domain Through the Use of Discrete Sine Transform With Axis Rotation”, International Conference on Signal Processing Applications & Technology, Dallas, EUA, *http://www.icspat.com/*, 2000.

5. A.C.P. Veiga, Y. Iano, L.V. Lima e A.B. Soares, “Electromyographic Signal Modelling by The Use of Discrete Sine Transform With Axis Rotation Adaptive Filter”, Proc. of the 10th International Conference on Biomedical Engineering, Singapura, 143-144, 2000.

6. A.C.P. Veiga, Y. Iano e G.A. Carrijo, “A Comparative Performance of DSTr Adaptive Filter With NLMS, DST-LMS, DCT-LMS and RLS Algorithms”, Proc. of The IASTED International Conference Modelling, Identification, and Control, Innsbruck, Austria, vol. II, 818-823, 2001.

7. A.C.P. Veiga, G.A.Carrijo e Y. Iano, “Electromyographic Signal Modelling Using an Adaptative Filter: Comparative Study of DST-LMS, DCT-LMS and DSTr-LMS”, World

Multiconference on Systemics, Cybernetics and Informatics, Florida, EUA, vol.VI, 425-429, 2001.

8. A.C.P. Veiga, Y. Iano e G.A.Carrijo, “A New Adaptive Filter Structure: Comparative Study of NLMS, DST-LMS and DCT-LMS Schemes Applied to Electromyographic Signal Modelling”, 2001 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, Victoria, Canada, vol. II, 555-558, 2001.

9. A.C.P. Veiga, Y. Iano, L.V. Lima and K. Yamanaca, “Insertion of the Discrete Sine Transform with Pre-Rotation in the LMS Algorithm”, submetido e aprovado para o V INDUSCON - IEEE - Industry Applications Society, 2002.

10. A.C.P. Veiga, Y. Iano, L.V. Lima and K. Yamanaca, “Insertion of the Discrete Sine Transform with Pre-Rotation in the LMS Algorithm,” submetido para a IEEE Transaction on Industry Applications, 2002.

11. Silva, C. A. L., G. A. Carrijo, L. V. Lima e A. C. P. Veiga, “Automatic Identification of Frequencies of Musical Notes in Polyphonics Wave Files,” Symposium on Computer Music SBCM2000, Brasil, 2000.

12. LOPES DA SILVA, C. A.—LIMA, L. V.—FERREIRA, W. B.—PASCHOARELLI VEIGA, A. C.—YAMANAKA, K.—IANO, Y.: *Automatic learning composition based on polyphonic wave signals and musical histograms*, Tatra Mt. Math. Publ. **23** (2001), 141–163