Universidade Federal do Maranhão
Centro de Ciências Exatas e Tecnologia
Curso de Pós-graduação em Engenharia de Eletricidade

CHARLES SILVA NAHUZ

ALGORITMO ADAPTATIVO TIPO-LMS COM SOMA DO ERRO

São Luís
2016

CHARLES SILVA NAHUZ

ALGORITMO ADAPTATIVO TIPO-LMS COM SOMA DO ERRO

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Eletricidade da UFMA, como requisito parcial para a obtenção do grau de MESTRE em Engenharia de Eletricidade.

Orientador: Allan Kardec Duailibe Barros

Dr em Eng. Informação - Univ. de Nagoya

Co-orientador: Ewaldo Eder Santana

Dr em Eng. Elétrica - Univ. de Campina Grande

São Luís

2016

Silva Nahuz, Charles

Algoritmo Adaptativo Tipo-LMS Com Soma do Erro / Charles Silva Nahuz - 2016.

55f.

1.Processamento de Sinais. 2.Filtragem Adaptativa. I.Título.

CDU 616.391

CHARLES SILVA NAHUZ

ALGORITMO ADAPTATIVO TIPO-LMS COM SOMA DO ERRO

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Eletricidade da UFMA, como requisito parcial para a obtenção do grau de MESTRE em Engenharia de Eletricidade.

Aprovado em 11 de Março de 2016

BANCA EXAMINADORA

Allan Kardec Duailibe Barros

Dr em Eng. Informação - Univ. de Nagoya

João Viana Fonseca

Dr em Eng. Elétrica - Univ. da Unicamp

Raimundo Carlos Silvério Freire

Dr em Eng. Elétrica - Univ. da França

Aos meus pais, minha família e irmãos.

Aos amigos, pelo apoio e companheirismo.

Resumo

Neste trabalho, implementamos um novo filtro semelhante ao LMS, porém, com uma função de custo baseada na soma do erro. Como resultado, obtemos uma função bastante simples, produzindo uma rápida convergência e um pequeno desajuste quando comparado com o algoritmo LMS e com outros algoritmos.

O filtro adaptativo é baseado em funções não lineares como estimativa do gradiente de uma superfície de desempenho. Utilizamos o gradiente do algoritmo para atualização dos pesos. Essa atualização baseia-se nas estatísticas de alta ordem para obtenção de informações sobre os sinais envolvidos no processo, com o objetivo de melhorar a performace do filtro adaptativo.

As equações foram derivadas e baseadas em séries de Taylor das funções não lineares, Para obtenção dos critérios que garante a sua convergência. Também fazemos um estudo da covariância do vetor peso em regime estacionário e determinamos as equações que calculam as constantes de tempo em um processo adaptativo.

Apresentamos o algoritmo proposto, que utiliza uma função de custo onde foram feitas simulções de Monte Carlo com sinais reais para validar a teoria apresentada. Nessa função os coeficientes α_k foram otimizados para dar maior estabilidade e melhor desempenho na sua velocidade de convergência.

Palavras-chaves: Filtros Adaptativos, Processamento de Sinais , Filtragem Adaptativa, otimização, LMS.

Abstract

In this paper, implemented a new filter similar to the LMS, but, with a coast function based in the sum of the error. As a result, we obtain a very simple function, producing a rapid convergence and a small mismatch when compared with the LMS algorithm and other algorithms.

The adaptive filter is based on non-linear functions such as estimation of the gradient of a surface performance. We use the gradient algorithm to update the weights. this update is based on high-order statistics to obtain information about the signs involved in the process, in order to improve the performace of the adaptive filter.

Derive the equations based on Taylor series of non-linear functions, to achieve the criteria that ensures their convergence. We also do a weight vector covariance study in steady state and determine the equations that calculate the time constants in an adaptive process.

Here the algorithm proposed, which uses a cost function and were made simulções Monte Carlo with real signals to validate the theory presented. In this role the α coefficients have been optimized to provide increased stability and better performance in its convergence speed.

Keywords: Adaptive Filters, Signal Processing, Adaptive Filtering, optimization, LMS.

Agradecimentos

Ao professor Allan Kardec Barros pela confiança, amizade, orientação e dedicação e principalmente, pela paciência, sem a qual este trabalho não se realizaria. E pelas disciplinas ministradas para o nosso enriquecimento pessoal e profissional.

Ao professor Ewaldo Eder Santana por ser o meu coorientador, pela amizade e pelos ensinamentos em sala de aulas nas disciplinas ministradas.

Ao Professor João Viana Fonseca pela amizade e pela dedicação dos ensinamentos nas disciplinas ministradas.

Em especial ao Professor Marcos Araujo pela revisão dos artigos e pelos ensinamentos nos cursos ministrados no PIB.

Aos professores Mairton Barros e Luis Fernando Amaral pela amizade e dedicação nos ensinamentos para conclusão deste trabalho.

Aos amigos do PIB pelo companheirismo pelos seus ensinamentos, que durante esses anos, contribuíram de algum modo para o nosso crescimento profissional.

À toda minha família, em especial aos meus filhos.

À CAPES pela bolsa a mim concedida.

Ao PPGEE pelos laboratórios e pelo espaço concedido pela UFMA para o desenvolvimento do estudo da pesquisa.

"Quando menos alguém entende, mais quer discordar".

Galileu Galilei

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

1.1 Tema e Proposta Deste Trabalho

O presente trabalho trata da filtragem adaptativa com implementação de um novo filtro semelhante ao LMS, porém, com uma função de custo baseada na soma do erro e tendo por objetivo específico explorar o algoritmo Least-Mean-Square (LMS) e outros algoritmos em experimentos de processamento digital adaptativo de sinais reais obtidos por digitalização [1].

O termo filtro é usado para descrever um sistema de hardware ou software, cuja finalidade pode ser tanto para recuperar informação contaminada por ruído ou simplesmente selecionar partes de interesse de um sinal, como por exemplo, componentes de uma certa banda de frequência. Filtros usados para tais propósitos podem ser a parâmetros fixos ou adaptativos.

Os filtros adaptativos operam com algoritmos recursivos, cuja principal vantagem consiste no ajuste automático dos parâmetros do filtro de acordo com as mudanças no sinal de entrada, tornando possível o funcionamento satisfatório destes filtros em ambientes não-estacionários [8].

Em ambientes estacionários, a minimização do erro quadrático médio resulta no filtro de Wiener, se considerada a classe dos sistemas lineares. O filtro de Wiener é, contudo, inadequado para situações de não-estacionaridade, quando soluções mais eficientes são obtidas por filtragem adaptativa [3].

Neste trabalho, é apresentada uma introdução ao estudo de um dos algoritmos recursivos mais usados na filtragem adaptativa, o algoritmo LMS é derivado do filtro de Wiener e sua formulação começa pelo desenvolvimento da solução de Wiener para o problema de minimização mencionado, usando filtros transversais lineares descrito no capítulo 2. O algoritmo LMS faz uso da estimativa instantânea do gradiente da função de desempenho para calcular os parâmetros do filtro. Devido à sua baixa complexidade computacional e robustez, este algoritmo constitui uma das soluções mais usadas para diferentes tipos de aplicação prática na área de processamento adaptativo de sinais [15].

1.2 Organização do Trabalho

O presente trabalho está organizado da seguinte forma.

No Capítulo 2 é introduzido o conceito e formulação do algoritmo LMS, começando pela teoria do filtro de Wiener, o algoritmo Steepest Descent e finalizando com o algoritmo LMS.

No Capítulo 3 abordamos a implementação do filtro adaptativo com suas devidas derivações e simulações em tempo real com os algoritmos implementados para o mesmo desajuste.

Em seguida, no Capítulo 5 é feita a implemetação do algoritmo Proposto onde fazemos suas derivações para garantir a sua convergência através dos parametros escalares $\alpha_{\mathbf{k}}$ que foram otimizados, apresentamos todo o seu desenvolvimento com comentários conclusivos, e finalmente a lista das referências bibliográficas.

2 Algoritmo LMS

2.1 Introdução

Neste capítulo, é apresentada a teoria de um dos algoritmos mais importantes e usado em filtragem adaptativa, o algoritmo LMS (Least Mean Square). Este algoritmo é baseado em uma estimativa do gradiente da função de desempenho J. Umas das principais vantagens do algoritmo LMS é a sua simplicidade e facilidade de implementação. Caracterizado apenas por três equações, compostas de operações simples de multiplicação, adição e subtração, esse algoritmo constitui geralmente uma das melhores escolhas para muitos dos diferentes tipos de aplicação na área de processamento adaptativo de sinais [1].

2.2 Filtro de Wiener

Para melhor entender os conceitos associados à filtragem adaptativa, é conveniente, primeiramente, abordar o filtro de Wiener. O filtro de Wiener é um filtro linear e ótimo no sentido do erro quadrático médio, como explicitado mais adiante [3].
Considere o diagrama de blocos da Figura (2.1) com um combinador linear adaptativo que apresenta um filtro linear discreto de resposta ao impulso $[\mathbf{w}_0, \mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_{M-1}]$ e cuja entrada consiste na sequência $\mathbf{u}(n)$. Em um instante n qualquer, o filtro produz a saída $y(n)$. O sinal $y(n)$ é, então, comparado à resposta desejada $d(n)$, sendo a diferença entre os dois o erro de estimação $e(n)$ (2.1). A entrada e a resposta desejada $d(n)$ são consideradas amostras de processos aleatórios [8].

$$e(n) = d(n) - y(n) \tag{2.1}$$

Pode se notar que em (2.1), quanto menor for o sinal do erro, mais próxima é a saída do filtro $y(n)$ e da resposta desejada $d(n)$. Tendo isso em mente, o objetivo é ajustar os parâmetros do filtro de modo a minimizar o erro. Dentre os vários critérios que podem ser usados para orientar esse ajuste o critério adotado, nos filtros de Wiener, é o da

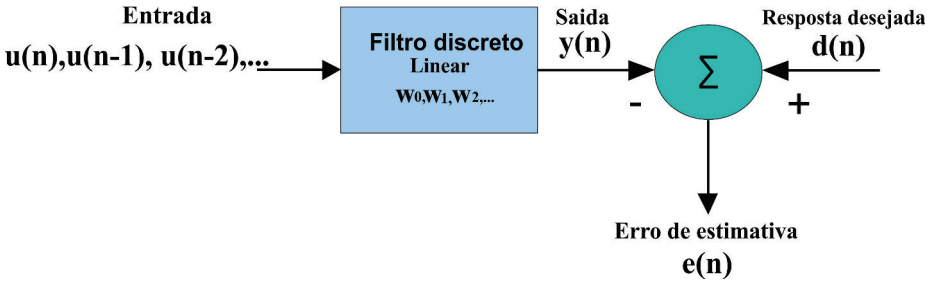

Figura 2.1: Diagrama de blocos da representação do problema estatístico de filltragem

minimização do erro quadrático médio, cuja formulação é dada por:

$$J = E[|e(n)|^2] \tag{2.2}$$

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

onde E[.] denota o operador do valor esperado; * representa o complexo conjugado; e J é a função-desempenho, ou função-custo (erro quadrático médio), a ser minimizada.

Essa função satisfaz duas importantes exigências:

1. É uma função simples, o que facilita a manipulação matemática; e
2. É quadrática, possuindo um único ponto ótimo.

Em particular, no caso de filtros de resposta ao impulso finita (finite impulse response - FIR), a superfície de desempenho é um parabolóide com concavidade orientada no sentido positivo com um único ponto de mínimo[10].

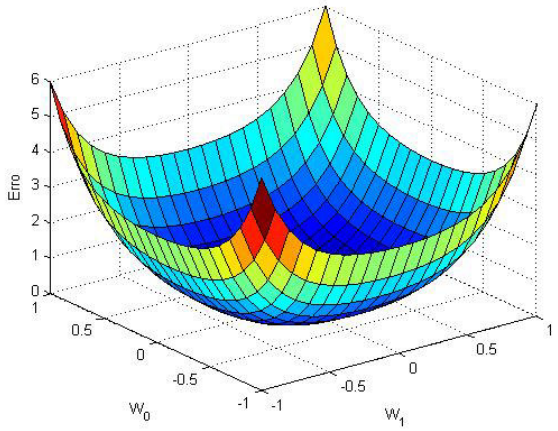

Figura 2.2: Superfície quadrática tridimensional, juntamente com o erro quadrático médio plotado na vertical, $\mathbf{W_0}$ e $\mathbf{W_1}$ variam de -1 a 1

2.2.1 Solução de Wiener para Filtros Transversais

Considere o filtro transversal da Figura 2 com um combinador linear adaptativo transversal, cuja entrada a cada instante n é caracterizada pelo vetor $\mathbf{u}(n) = [\mathbf{u}(n), \mathbf{u}(n-1), \mathbf{u}(n-2), ..., \mathbf{u}(n-M+1)]^T$. Assume-se que tanto a entrada $\mathbf{u}(n)$ assim como a sua resposta desejada $d(n)$ são processos aleatórios estacionários e, em geral, complexos. É considerado também que os parâmetros do filtro $[\mathbf{w}_0, \mathbf{w}_1, ..., \mathbf{w}_{M-1}]^T$ são complexos[11].

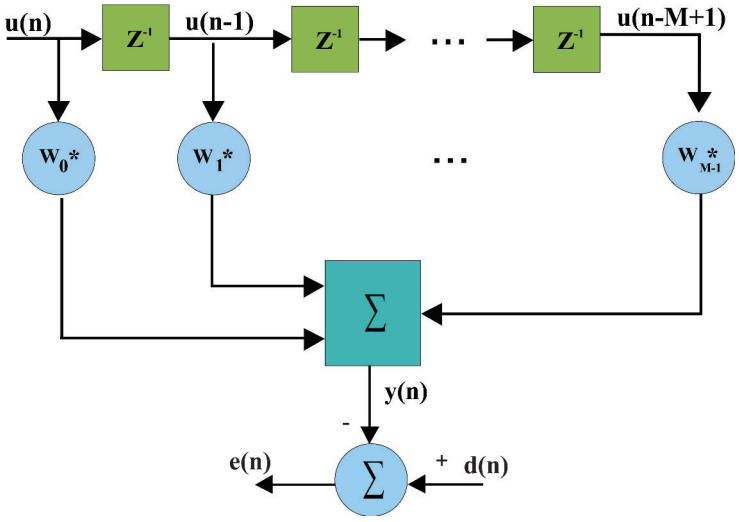

Figura 2.3: Diagrama em blocos de um filtro transversal

Os passos a sequir apresentam um breve desenvolvimento da solução de Wiener para o problema de minimização da função J considerando filtros transversais. Os vetores de entrada $\mathbf{u}(n)$ e de parâmetros do filtro $\mathbf{w}$ são definidos, respectivamente, como:

$$\mathbf{u}(n) = [\mathbf{u}(n), \mathbf{u}(n-1), ..., \mathbf{u}(n-M+1)]^T \tag{2.4}$$

$$\mathbf{w} = [\mathbf{w}_0, \mathbf{w}_1, ..., \mathbf{w}_{M-1}]^T \tag{2.5}$$

Por sua vez, a saída do filtro y(n) é dada por:

$$y(n) = \sum_{k=0}^{m-1} \mathbf{w}_k^* \mathbf{u}(n-k) \tag{2.6}$$

$$y(n) = \mathbf{w}^T \mathbf{u}(n), \tag{2.7}$$

onde os sobrescritos T e k significa, respectivamente, transposto e indice.

Considerando a formulação vetorial, o sinal de erro pode ser expresso como:

$$e(n) = d(n) - y(n) \tag{2.8}$$

$$e(n) = d(n) - \mathbf{w}^T\mathbf{u}(n) \tag{2.9}$$

Substituindo (2.9) em (2.3), obtém-se:

$$J = E[d(n) - \mathbf{w}^T\mathbf{u}(n)][d^*(n) - \mathbf{u}(n)\mathbf{w}] \tag{2.10}$$

$$J = E[|d(n)|^2] - \mathbf{p}^T\mathbf{w} - \mathbf{w}^T\mathbf{p} + \mathbf{w}^T\mathbf{R}\mathbf{w} \tag{2.11}$$

onde $\mathbf{p}$ e $\mathbf{R}$ são, respectivamente: o vetor de correlação cruzada entre $\mathbf{u}(n)$ e $d(n)$; e a matriz de autocorrelação de entrada $\mathbf{u}(n)$.

Matematicamente, $\mathbf{p}$ e $\mathbf{R}$ são definidos como:

$$\mathbf{p} = E[\mathbf{u}(n)d^*(n)] \tag{2.12}$$

$$\mathbf{R} = E[\mathbf{u}(n)\mathbf{u}^T(n)] = E\begin{bmatrix} r(0)^2 & r(1) & \cdots & r(M-1) \\ r^*(1) & r(1)^2 & \cdots & r(M-2) \\ \vdots & \vdots & \ddots & \vdots \\ r^*(M-1) & r^*(M-2) & \cdots & r(m-n)^2 \end{bmatrix} \tag{2.13}$$

A Expressão (2.11) é uma função quadrática dependente dos parâmetro do filtro transversal. Pode-se dizer que ao mínimo de J está associado um vetor $\mathbf{w}_0$ que corresponde aos parâmetros do filtro ótimo. O processo para determinação de $\mathbf{w}_0$ é tradicional: calcula-se o gradiente de J, igualando-o a zero. A partir de (2.11), o gradiente de J resulta em[15]:

$$\nabla J = \left[\frac{\partial J}{\partial \mathbf{w}_0}, \frac{\partial J}{\partial \mathbf{w}_1}, ..., \frac{\partial J}{\partial \mathbf{w}_{M-1}}\right]^T \tag{2.14}$$

$$\nabla J = 2\mathbf{p} - 2\mathbf{R}\mathbf{w} \tag{2.15}$$

que igualado a zero, permite a determinação de $\mathbf{w}_o$ ou $\mathbf{w}_*$.

$$\mathbf{R}\mathbf{w}_* = \mathbf{p} \tag{2.16}$$

$$\mathbf{w}_* = \mathbf{R}^{-1}\mathbf{p} \tag{2.17}$$

O sistema de equações representado por (2.16) é conhecido por equações de Wiener-Hopf. A solução de tal sistema, como apresentado em (2.17), assume que $\mathbf{R}$ é uma matriz não-singular. Dessa forma, calculam-se os parâmetros de um filtro linear e transversal que minimiza (2.11)(filtro de Wiener).

2.2.2 Princípio da Ortogonalidade

O princípio da ortogonalidade decorre do cálculo do gradiente de J, obtido através das derivadas parciais de J em relação às partes real e imaginária de cada parâmetro do filtro, uma vez que, em geral, esses parâmetros são complexos[8]. Assim, a derivada da função J em relação ao parâmetro $\mathbf{w}_k$ do filtro é dada por

$$\nabla_k J = E\left[\frac{\partial e(n)}{\partial a_k}e^*(n) + \frac{\partial e^*(n)}{\partial a_k}e(n) + \frac{\partial e(n)}{\partial b_k}Je^*(n) + \frac{\partial e^*(n)}{\partial b_k}Je(n)\right] \tag{2.18}$$

onde a_k e b_k são as partes real e imaginária de $\mathbf{w}_k$, respectivamente, como mostrado em (2.21).

$$\mathbf{w}_k = a_k + jb_k, \qquad k = 0, 1, ..., M-1 \tag{2.19}$$

A partir de (2.9), chega-se às expressões para as derivadas parciais necessárias em (2.20):

$$\frac{\partial e(n)}{\partial a_k} = -\mathbf{u}(n-k) \tag{2.20}$$

$$\frac{\partial e(n)}{\partial b_k} = J\mathbf{u}(n-k) \tag{2.21}$$

$$\frac{\partial e^*(n)}{\partial a_k} = -\mathbf{u}^*(n-k) \tag{2.22}$$

$$\frac{\partial e^*(n)}{\partial b_k} = -J\mathbf{u}^*(n-k) \tag{2.23}$$

Substituindo (2.22) - (2.25) em (2.20), obtém-se:

$$\nabla_k J = -2E[\mathbf{u}(n-k)e^*(n)] \tag{2.24}$$

Igualando o resultado de (2.25) a zero, chega-se a:

$$E[\mathbf{u}(n-k)e^*(n)] = 0, \qquad k = 0, 1, 2, ... \tag{2.25}$$

Esta equação é conhecida como o princípio da ortogonalidade e mostra que a condição necessária e suficiente para que a função de desempenho J atinja o seu valor mínimo é aquela em que o sinal de erro é ortogonal (não-correlacionado) ao sinal de entrada que é usado para a estimação da resposta desejada[10].

Como consequência de (2.25), quando o filtro atinge a condição ótima, tem-se que:

$$E[y(n-k)e^*(n)] = 0, \qquad k = 0, 1, 2, ... \tag{2.26}$$

Ou seja, a saída do filtro de Wiener é também ortogonal ao sinal de erro.

2.2.3 Erro Quadrático Médio Mínimo

Para a determinação do valor mínimo da função de desempenho J_{min}, pode-se rescrever(2.11) como:

$$J_{min} = E[|d(n)|^2] - \mathbf{p}^T\mathbf{w}_o - \mathbf{w}_o^T\mathbf{p} + \mathbf{w}_o^T\mathbf{R}\mathbf{w}_o \tag{2.27}$$

Substituindo o $\mathbf{w}_o$ pela expressão (2.19), tem-se:

$$J_{min} = E[|d(n)|^2] - \mathbf{p}^T\mathbf{R}^{-1}\mathbf{p} - (\mathbf{R^{-1}p})^T\mathbf{p} + (\mathbf{R^{-1}p})^T\mathbf{R}\mathbf{R}^{-1}\mathbf{p}, \tag{2.28}$$

$$J_{min} = \sigma_d^2 - \mathbf{p}^T\mathbf{R}^{-1}\mathbf{p}, \tag{2.29}$$

onde σ_d^2 representa a variância da resposta desejada $d(n)$.

2.3 O Algoritmo Steepest Descent

Como visto na Seção 2.3 , para achar os parâmetros do filtro de Wiener deve-se, a princípio, resolver o sistema de equações de Wiener-Hopf (2.18). A resolução de tal sistema é computacionalmente dispendiosa e numericamente mal-condicionada devido à inversâo da matriz $\mathbf{R}$. Tais problemas tendem a se agravar na medida do crescimento da ordem do filtro[15].

O algoritmo Steepest Descent é um modo alternativo, em relação à resolução explícita das equações de Wiener-Hopf, para achar a solução de Wiener de um filtro transversal. A abordagem usada consiste na estratégia de atualizar iterativamente os parâmetros do filtro transversal utilizando o gradiente da função de desempenho.

O algoritmo Steepest Descent funciona da seguinte forma:

1. Começa-se atribuindo a $\mathbf{w}$ um valor inicial qualquer que corresponde ao $\mathbf{w}(0)$, ou seja a estimativa dos parâmetros do filtro no instante 0 (zero). A menos que se tenha algum conhecimento a priori sobre os parâmetros do filtro $\mathbf{w}(n)$, $\mathbf{w}(0)$ é usualmente igualado ao vetor nulo.

2. Para n = 0,1,2,...

(a) Calcula-se o vetor gradiente da função de desempenho na n-ésima iteração $\nabla J(n)$, de acordo com a expressão (2.32), que por sua vez é baseada em (2.17).

$$\nabla J(n) = -2\mathbf{p} + 2\mathbf{R}\mathbf{w}(n) \tag{2.30}$$

(b) Como o gradiente aponta no sentido do crescimento da função, o vetor de parâmetros é atualizado no sentido oposto ao do vetor gradiente:

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu[\nabla J(n)] \tag{2.31}$$

onde μ denota o passo de adaptação.

O parâmetro μ controla a estabilidade e a velocidade de convergência do algoritmo. Quanto maior o μ, mais rápida será a convergência. Por outro lado, valores muito altos de μ podem provocar instabilidade no processo de estimação. Para que se garanta à estabilidade do sistema, o parâmetro μ deve obedecer a seguinte desigualdade

$$0 < \mu < \frac{1}{\lambda_{max}} \tag{2.32}$$

onde λ_{max} é o maior autovalor da matriz de auto-correlação $\mathbf{R}$.

2.4 O Algoritmo Least-Mean-Square

O Least-Mean-Square (LMS) é um algoritmo amplamente usado em filtragem adaptativa devido à sua baixa complexidade computacional e robustez às mudanças nas estatísticas do sinal de entrada . No contexto da filtragem adaptativa, os parâmetros do filtro são considerados variantes no tempo, acomodando-se continuamente a eventuais mudanças nas estatísticas do sinal da entrada[11]. Quando os sinais a processar são estacionários, o algoritmo LMS tende para a solução ótima de Wiener.

O LMS executa dois processos básicos:

1. Filtragem, que implica o cálculo da saída gerada por um filtro discreto e a geração de um sinal de erro estimado proveniente da comparação dessa saída do filtro com a saída desejada.

2. Adaptação, que ajusta automaticamente os coeficientes do filtro de acordo com as mudanças nas características do sinal de entrada.

Como visto na Seção 2.3, o método Steepest Descent requer fazer o cálculo do vetor gradiente da função de desempenho a cada iteração. No LMS, tal cálculo é substituído, para efeito de redução da complexidade computacional, por uma estimativa do vetor gradiente. O consequente erro ou ruído de estimação se deve ao fato de que cada componente da estimativa do vetor gradiente é obtida através de uma única amostra do vetor de entrada. Contudo, à medida que mais amostras são processadas com o passar do tempo, o ruído do gradiente diminui gradualmente[3].

A expressão de adaptação dos coeficientes do filtro passa a ser:

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\hat{\nabla}j(n) \tag{2.33}$$

onde $\hat{\nabla}j(n)$ representa a mencionada estimativa do vetor gradiente da função de desempenho.

Na expressão (2.32) do $\nabla J(n)$, substitui-se a matriz $\mathbf{R} = E[\mathbf{u}(n)\mathbf{u}^T(n)]$ e o vetor $\mathbf{p} = E[\mathbf{u}(n)d^*(n)]$, respectivamente por $\hat{\mathbf{R}} = \mathbf{u}(n)\mathbf{u}^T(n)$ e $\hat{\mathbf{p}} = \mathbf{u}(n)d^*(n)$, resultando em:

$$\hat{\nabla}J(n) = -2\mathbf{u}(n)\mathbf{u}^T(n)\mathbf{w}(n) + 2\mathbf{u}(n)d^*(n). \tag{2.34}$$

Substituindo (2.34) em (2.35),

$$\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu\mathbf{u}(n)e^*(n). \tag{2.35}$$

A seguir, as três equações que resumem o algoritmo LMS:

1. Saída do filtro

$$y(n) = \mathbf{w}^T(n)\mathbf{u}(n) \tag{2.36}$$

2. Cálculo do erro

$$e(n) = d(n) - y(n) \tag{2.37}$$

3. Adaptação dos parâmetros do filtro

$$\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu\mathbf{u}(n)e^*(n) \tag{2.38}$$

Resultando na equação

A constante μ controla a velocidade de convergência e a estabilidade do processo de adaptação. Um μ muito alto resultará em instabilidade e oscilações. Por outro lado, em situações de não-estacionaridade, um μ muito baixo resultará em baixa capacidade de adaptação às mudanças nas estatísticas do sinal de entrada. A estabilidade e convergência do processo de adaptação são garantidas, desde que o valor de μ esteja de acordo com a inequação

$$0 < \mu < \frac{1}{\lambda_{max}} \tag{2.39}$$

2.5 Conclusão do Capítulo

Com os estudos apresentados neste capítulo, pode-se chegar às seguintes conclusões:

O filtro de Wiener é ótimo no sentido do erro quadrático médio, mas sua solução é numericamente mal-condicionada e computacionalmente dispendiosa. Este problema é contornado pelo algoritmo Steepest Descent que por sua vez calcula a solução de Wiener iterativamente.

Tanto o filtro de Wiener como o algoritmo Steepest Descent são impraticáveis para situações de não-estacionaridade dos sinais, o que leva a uma outra versão baseada na estimativa instantânea do gradiente, o algoritmo LMS, cujo desempenho é controlado pelo passo de adaptação μ[15].

O algoritmo LMS apresenta um problema de amplificação do ruído associado à estimação do gradiente quando a potência do sinal de entrada é alta. Este problema é contornado através da normalização do parâmetro de adaptação μ.

3 Derivação do Algoritmo LMS

3.1 Derivação do LMS

A fórmula mostra um filtro adaptativo de entrada desejada $\mathbf{x}(n)$, e saida d(n).

$$y(n) = \sum_{i=0}^{n-1} \mathbf{w}_i(n)\mathbf{x}(n-i) \tag{3.1}$$

São consideradas as sequências de valor real. Os pesos $\mathbf{w}_0(n), \mathbf{w}_1(n), ..., \mathbf{w}_{n-1}(n)$ são selecionados, de modo que a diferença do erro será.

$$e(n) = d(n) - y(n) \tag{3.2}$$

É minimizado em algum sentido. Podemos notar que os pesos são explicitamente indicado para ser funções do tempo com indice(n). O algoritmo LMS convencional é uma implementação do algoritmo estocástico com descida mais ingrime. Ele simplesmente substitui a função de custo $\xi = E[e^2(n)]$ por sua forma instantânea rudimentar e estima $\xi = e^2(n)$[8]. Substituindo $\xi = e^2(n)$ para ξ na recursão com descida mais ingreme e substiuindo o indice de interação k pelo indice de tempo (n), teremos.

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\nabla e^2(n) \tag{3.3}$$

onde $\mathbf{w}(n) = [\mathbf{w}_0(n), \mathbf{w}_1(n), ..., \mathbf{w}_{n-1}(n)]^T$, μ é o parâmetro do algoritmo com tamanho e passo, e ∇ é o gradiente do operador definido como vetor coluna.

$$\nabla = \left[\frac{\partial}{\partial \mathbf{w}_0}\frac{\partial}{\partial \mathbf{w}_1}.....\frac{\partial}{\partial \mathbf{w}_{n-1}}\right]^T \tag{3.4}$$

Notamos que o elemento i do gradiente do vetor $\nabla e^2(n)$ é

$$\frac{\partial e^2(n)}{\partial \mathbf{w}_i} = 2e(n)\frac{\partial e(n)}{\partial \mathbf{w}_i} \tag{3.5}$$

substituindo a equação(2) no ultimo fator do lado direito da equação(5) notando que d(n) é independente de $\mathbf{w}$; teremos

$$\frac{\partial e^2(n)}{\partial \mathbf{w}_i} = -2e(n)\frac{\partial y(n)}{\partial \mathbf{w}_i} \tag{3.6}$$

Substituindo y(n) a partir da equação(1), obtemos

$$\frac{\partial e^2(n)}{\partial \mathbf{w}_i} = -2e(n)\mathbf{x}(n-i) \tag{3.7}$$

substituindo a equação(4) na equação(7), obtemos

$$\nabla e^2(n) = -2e(n)\mathbf{x}(n-i) \tag{3.8}$$

Resumo do algoritmo LMS.

Entrada:

Vetor de Entrada, $\mathbf{x}(n)$,

Vetor Peso $\mathbf{w}(n)$,

Vetor de Saida desejado, $\mathbf{d}(n)$

Saida:

Saida do filtro, $y(n)$

Vetor peso atualizado, $\mathbf{w}(n+1)$

1. Filtragem

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

2. Erro Estimado

$e(n) = d(n) - y(n)$

3.Adaptação do vetor peso

$\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu e(n)\mathbf{x}(n)$

Onde $\mathbf{x}(n) = [\mathbf{x}(n), \mathbf{x}(n-1), ..., \mathbf{x}(n-M+1)]^T$, substituindo este resultado na equação(3), obtemos o comportamento da média do vetor peso no algoritmo LMS.

$$\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu e(n)\mathbf{x}(n) \tag{3.9}$$

3.2 Convergência do vetor peso

Consideremos a entrada do filtro x(n) e sua saída desejada d(n), são estacionários. neste caso o vetor peso ótimo $\mathbf{w}_0$ transversal do filtro de wiener é fixo e pode ser obtido de acordo com a equação de wiener-hopf. Subtraindo $\mathbf{w}_0$ de ambos os lados da equação(9), obtemos[11]

$$\mathbf{v}(n+1) = \mathbf{v}(n) + 2\mu e(n)\mathbf{x}(n) \tag{3.10}$$

Onde $\mathbf{v}(n) = \mathbf{w}(n) - \mathbf{w}_0$ é o vetor de ponderação de erro. Observamos tambèm que:

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

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

$$e(n) = d(n) - \mathbf{x}^T(n)\mathbf{w_0} - \mathbf{x}^T(n)(\mathbf{w}(n) - \mathbf{w_0})$$

$$e(n) = e_0(n) - \mathbf{x}^T(n)\mathbf{v}(n) \tag{3.11}$$

onde

$$e_0(n) = d(n) - \mathbf{x}^T(n)\mathbf{w_0} \tag{3.12}$$

é o erro de estimação quando os pesos do filtro são ótimos. Substituindo a equação(11) na equação(10) e reorganizando, obtém-se:

$$\mathbf{v}(n+1) = (\mathbf{I} - 2\mu\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{v}(n) + 2\mu e_0(n)\mathbf{x}(n)) \tag{3.13}$$

onde I é a matriz identidade. aplicando as esperanças em ambos os lados da equação(13), obtemos:

$$E[\mathbf{v}(n+1)] = E[(\mathbf{I} - 2\mu\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{v}(n))] + 2\mu E[e_0(n)\mathbf{x}(n)]$$

$$E[\mathbf{v}(n+1)] = E[(\mathbf{I} - 2\mu\mathbf{x}(n)\mathbf{x}^T(n))\mathbf{v}(n)] \tag{3.14}$$

onde a última iqualdade resulta no fator que $E[e_0(n)\mathbf{x}(n)] = 0$, de acordo com o princípio da ortogonalidade. Usando a suposição de independencia, pode-se argumentar que, $\mathbf{v}(n)$ depende apenas das útimas observações $[\mathbf{x}(n-1), d(n-1)], [\mathbf{x}(n-2), d(n-2)], ...$, é independente de $\mathbf{x}(n)$, asssim[8]:

$$E[\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{v}(n)] = E[\mathbf{x}(n)\mathbf{x}^T(n)]E[\mathbf{v}(n)] \tag{3.15}$$

podemos notar que na maioria dos casos, a suposição de independência é questionável. Por exemplo, no caso do comprimento M transversal do filtro, os vetores de entrada são:

$$\mathbf{x}(n) = [\mathbf{x}(n)\mathbf{x}(n-1), ..., \mathbf{x}(n-M)]^T$$

e

$$\mathbf{x}(n-1) = [\mathbf{x}(n-1)\mathbf{x}(n-2), ..., \mathbf{x}(n-M)]^T$$

ter M-1 termos em comum fora de M. No entanto as esperanças com o LMS mostrou que as simulações feitas pela suposição de independência se combinam as simulações no computador e o desempenho real do algoritmo LMS, na prática substituindo a equação(15) na equação(14), obtemos[4].

$$E[\mathbf{v}(n+1)] = (\mathbf{I} - 2\mu\mathbf{R})E[\mathbf{v}(n)]) \tag{3.16}$$

Onde $\mathbf{R} = E[\mathbf{x}(n)\mathbf{x}^T(n)]$é a matriz de correlação de entrada do vetor $\mathbf{x}(n)$. Fazendo uma simulação na equação(16) podemos mostrar que $E[\mathbf{v}(n)]$ converge para zero quando μ permanece dentro do intervalo.

$$0 < \mu < \frac{1}{\lambda_{max}} \tag{3.17}$$

onde λ_{max} é o autovalor máximo de $\mathbf{R}$. no entanto devemos considerar que o intervalo acima, não garante necessariamente a estabilidade do algoritmo LMS. A convergência do algoritmo LMS tem uma convergência da média de $\mathbf{w}(n)$ e também para $\mathbf{w_0}$ a convergência da variância dos elementos de $\mathbf{w}(0)$ para alguns valores limitados. O tamanho de μ é que vai definir a convergência do algoritmo[8].

3.2.1 Curva de aprendizagem

Na figura podemos ver a curva de aprendizagem resultante do uso do algoritmo LMS.

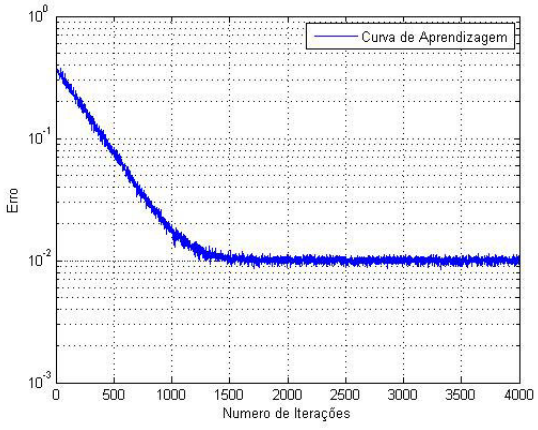

Figura 3.1: Curva de aprendizagem do algoritmo LMS. Na horizontal temos o número de iterações e na vertical o erro.

Vemos que esta curva é de natureza exponencial. podemos desta forma aproximá-la por um envelope exponencial dado por $e^{\frac{-t}{\tau}}$, onde t é o tempo e τ é uma grandeza chamada de

constante de tempo, de forma tal que uma iteração seja igual à uma unidade de tempo. A constante de tempo está relacionada com o n-ésimo autovalor da matriz $\mathbf{R}$ da seguinte forma[4]:

$$\tau_n = \frac{1}{2\mu\lambda_n}, \tag{3.18}$$

na qual μ é o tamanho do passo, que regula o processo adaptativo e λ_n é o enésimo autovalor de $\mathbf{R}$.

3.3 MSE comportamento do algoritmo LMS

Na curva de aprendizagem podemos ver que quando os peso não são iquais a $\mathbf{W}_*$, o erro quadrático médio (ξ) é maior que o erro quadrático médio mínimo (ξ_{min}). Temos assim, um excesso no erro final[3][15].

Definimos, então o excesso do erro quadrático médio, $Excesso_{MSE}$, como a diferença entre o erro quadrático médio atual (ξ_k) e o erro quadrático médio minimo:

$$Excesso_{MSE} = E[\xi_k - \xi_{min}]$$

$$Excesso_{MSE} = \mu E[n_k^2] tr[\mathbf{R}], \tag{3.19}$$

onde $tr[\mathbf{R}]$ é o traço da matriz $\mathbf{R}$ e n_k^2 é um sinal de ruído.

Definimos , também a diferença entre o erro quadrático médio atual e o erro quadrático médio mínimo, normalizado pelo erro quadrático médio mínimo, como o desajuste ($\mathfrak{M}$)[15].

$$\mathfrak{M} = \frac{E[\xi_k - \xi_{min}]}{\xi_{min}.} \tag{3.20}$$

Desta forma, temos que:

$$\mathfrak{M}_{\boldsymbol{LMS}} = \mu tr[\mathbf{R}] \tag{3.21}$$

3.4 Conclusão do Capítulo

Neste capítulo, realizamos uma revisão da superfície quadrática gerada quando se utiliza o erro quadrático médio como critério aplicado sobre o erro em um filtro adaptativo. Mostramos a derivação do algoritmo LMS e descrevemos as equações que determinam sua condição de convergência. A constante de tempo e o desajuste também são enfatizados, pois os mesmos são utilizados com referência comparativa de outros algoritmos adaptativos[15].

4 Algoritmo Adaptativo Tipo-LMS Com a Soma do Erro

4.1 Introdução

Em muitas aplicações de processamento de sinais utilizando a filtragem adaptativa, existe uma necessidade de algoritmos que miniminizam pequenos erros, com rápida convergência e baixa complexidade computacional [2].

Além disso; subjacente a estes métodos algumas simplificações sobre a estatística do sinal são assumidas, esse procedimento é importante na análise do algoritmo. Entretanto, pode introduzir grandes erros no sistema.

Entre os algoritmos propostos, podemos encontrar o proposto por Wallach e Widrow [3], o que é baseado no quarto momento do erro. Além disso, houve uma proposta por Chambers [5], que utiliza a soma do erro e as obras de Barros e colegas [13][4]. A idéia por trás da soma dos erros é que se pode ter um bom comportamento do momento de segunda ordem no estado estacionário aliada à rápida convergência de ordem superior até mesmo momento no entanto, a desvantagem desses trabalhos é que os coeficientes foram escolhidos, em vez empiricamente.

4.2 Estatísticas de Segunda Ordem e Estatística de alta Ordem

Entre os filtros adaptativos, o algoritmo (LMS) de Widrow e Hoff aparece como um dos mais amplamente utilizados. O LMS pertence a uma classe de algoritmos que pode ser designado como estatísticas de segunda ordem(SOS), em oposição a estatísticas de ordem superior(HSO). A utilização de métodos de (SOS) é suficiente quando se assume que os sinais envolvidos no processo têm uma distribuição Gaussiana, produzindo uma série de simplificações na análise do comportamento do algoritmo, bem como levando a métodos com menor custo computacional, em oposição aos métodos baseados em (HOS).

Curiosamente, devido ao aumento do poder computacional nas últimas décadas, métodos (HOS) têm atraído mais atenção da comunidade de pesquisa. Na verdade, em vez de lidar apenas com a potência do sinal (ou seja, as estatísticas de segunda ordem), (HOS) permite o acesso à informação contida em todos os momentos do sinal, originando, por conseguinte, uma melhor aproximação da distribuição efetiva do sinal em estudo. Por sua vez, pode-se esperar que os algoritmos projetados no âmbito do quadro (HOS) se comportam de forma mais eficiente [8].

No entanto, isso pode levar a grandes erros, no caso do tempo de convergência, uma vez que é uma indicação da rápidez com que o algoritmo iniciou a aprendizagem. Assim, propomos uma nova maneira de avaliar o tempo de convergência, através da análise do comportamento do algoritmo no início da aprendizagem.

Implementamos uma função de custo baseada na soma do erro. O resultado é muito semelhante ao algoritmo LMS, mais eficiente em termos de velocidade de convergência e erro de desadaptação, mesmo para os sinais que não possuem distribuição de Gaussiana [10]. Além disso, analisamos o tempo de convergência e desajuste do algoritmo proposto.

4.3 O Método de Aplicação

Consideremos a Fig.(4.1) com um sinal desejado d_j, $\mathbf{X}_j$ é a entrada do filtro. Definimos que $d_j = s_j + n_j$, onde s_j é o sinal que será extraido, e n_j é o ruido. Supondo que n_j é estatisticamente independente de s_j e de $\mathbf{X}_j$ [8].

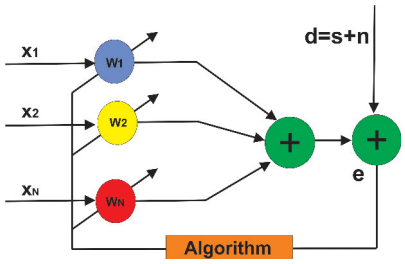

Figura 4.1: Diagrama de blocos do filtro com um sinal desejado d composto por um sinal s e um ruido n, com um vetor entrada $\mathbf{X}$ e o erro e, que é o retorno do algoritmo para atualização do vetor peso $\mathbf{W}$.

Desejamos recuperar s_j estimando um determinado sinal de saida $y_j = \mathbf{W}_j^T \mathbf{X}_j$, após calcular o erro $e_j = d_j - y_j$. O algoritmo de Widrow-Hoff usa uma estimativa instantânea do gradiente da função custo $E\left\{e_j^2\right\}$.

Estamos interessados em minimizar a sequinte função custo.

$$\xi_j = \sum_{k=1}^{p} \alpha_k^2 E\left\{e_j^{2k}\right\} \tag{4.1}$$

Na Fig.(4.2) vemos a superfície tridimensional gerada pela função $\xi_j = \sum_{k=1}^{p} \alpha_k^2 E\left\{e_j^{2k}\right\}$ considerando apenas dois pesos.

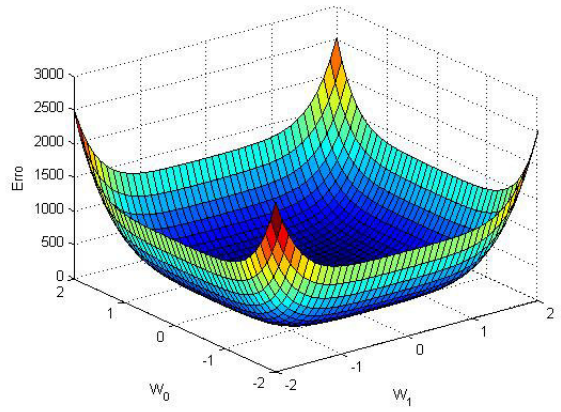

Figura 4.2: Gráfico da superfície gerada pela função $\xi_j = \sum_{k=1}^{p} \alpha_k^2 E\left\{e_j^{2k}\right\}$ e os pesos $\mathbf{W_0}$ e $\mathbf{W_1}$ variam de -2 a 2.

Fazendo algumas manipulações matemáticas obtemos o gradiente instantâneo da Eq.(4.1):

$$y_j = \mathbf{W}_j^T \mathbf{X}_j$$

$$e_j = d_j - y_j$$

$$e_j = d_j - \mathbf{W}_j^T \mathbf{X}_j$$

Calculando $\frac{\partial e_j}{\partial \mathbf{W}_j}$ temos:

$$\frac{\partial e_j}{\partial \mathbf{W}_j} = -\frac{\partial \mathbf{W}_j^T}{\partial \mathbf{W}_j}\mathbf{X}_j$$

Então o $\nabla_{\mathbf{W}_j} e_j = -\mathbf{X}_j$, Logo o gradiente instantâneo será:

$$\nabla \xi_j = \sum_{k=1}^{p} \alpha_k^2 2kE\left\{e_j^{2k-1}\right\} \nabla_{\mathbf{W}_j} e_j$$

$$\nabla \xi_j = \sum_{k=1}^{p} \alpha_k^2 2kE\left\{e_j^{2k-1}\right\} (-\mathbf{X}_j)$$

Então:

$$\nabla \xi_j = -2\left(\sum_{k=1}^{p} \alpha_k^2 k e_j^{2k-1}\right)\mathbf{X}_j$$

Conduzirá a seguinte regra de atualização simples para os pesos. E pela equação abaixo temos:

$$\mathbf{W}_{j+1} = \mathbf{W}_j - \mu\nabla\xi_j$$

$$\mathbf{W}_{j+1} = \mathbf{W}_j - \mu\left\{-2\left(\sum_{k=1}^{p}\alpha_k^2 k {e_j}^{2k-1}\right)\mathbf{X}_j\right\}$$

$$\mathbf{W}_{j+1} = \mathbf{W}_j + 2\mu\left(\sum_{k=1}^{p}\alpha_k^2 k {e_j}^{2k-1}\right)\mathbf{X}_j \tag{4.2}$$

e

$$\alpha_{k,j+1} = \alpha_{k,j} - \mu\hat{\nabla}\xi_j$$

$$\alpha_{k,j+1} = \alpha_{k,j} - 2\gamma\alpha_{k,j}{e_j}^{2k},\ k = 1, 2, \ldots, p, \tag{4.3}$$

onde μ e γ são constantes que controlam a estabilidade, a taxa de convergência e de aprendizagem do algoritmo[7]. Além disso, ${e_j}^{2k}$ é pequeno quando $j \to \infty$, portanto, $\alpha_{k,j+1}$ tende a ser igual a $\alpha_{k,j}$.

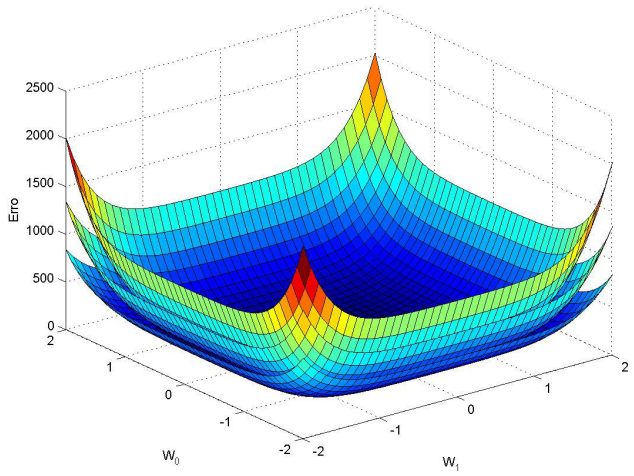

Figura 4.3: Gráfico das superfícies gerada pelo algoritmo Proposto com variação dos α_k e os pesos $\mathbf{W_0}$ e $\mathbf{W_1}$ variam de -2 a 2.

4.4 Comportamento do Vetor Peso

Para analisar o comportamento do vetor peso devemos verificar as condições sob as quais o algoritmo converge para a solução desejada, e como se comporta até atingir o estado estacionário. Isso pode ser realizado por meio da análise do desajuste do erro e o tempo de convergência[3].

Primeiro faremos uma mudança de variável, definindo o vetor $\mathbf{V}_j = \mathbf{W}_j - \mathbf{W}_*$, onde $\mathbf{W}_*$ é a solução ideal, ou seja, $s_j = \mathbf{W}_*^T \mathbf{X}_j$. Assim Eq.(4.2) torná-se:

$$\mathbf{W}_{j+1} - \mathbf{W}_* = \mathbf{W}_j - \mathbf{W}_* + 2\mu \left[\sum_{k=1}^{p} \alpha_k^2 k {e_j}^{2k-1} \right] \mathbf{X}_j$$

$$\mathbf{V}_{j+1} = \mathbf{V}_j + 2\mu \left(\sum_{k=1}^{p} \alpha_k^2 k {e_j}^{2k-1} \right) \mathbf{X}_j \tag{4.4}$$

A Eq.(4.4) pode ser reescrita como:

$$d_j = s_j + n_j$$

$$s_j = \mathbf{W}_*^T \mathbf{X}_j$$

$$e_j = d_j - \mathbf{W}_j^T \mathbf{X}_j$$

$$e_j = s_j + n_j - \mathbf{W}_j^T \mathbf{X}_j$$

$$e_j = \mathbf{W}_*^T \mathbf{X}_j + n_j - \mathbf{W}_j^T \mathbf{X}_j$$

$$e_j = n_j - \mathbf{W}_j^T \mathbf{X}_j + \mathbf{W}_*^T \mathbf{X}_j$$

$$e_j = n_j - \left(\mathbf{W}_j^T - \mathbf{W}_*^T \right) \mathbf{X}_j$$

Sabemos que $\mathbf{V}_j^T = \mathbf{W}_j^T - \mathbf{W}_*^T$, mais também sabemos que $\mathbf{V}_j^T \mathbf{X}_j = \mathbf{X}_j^T \mathbf{V}_j$ então temos [11]:

$$e_j = n_j - \mathbf{V}_j^T \mathbf{X}_j$$

ou

$$e_j = n_j - \mathbf{X}_j^T \mathbf{V}_j$$

Substituindo o erro $e_j = n_j - \mathbf{X}_j^T \mathbf{V}_j$ na Eq.(4.2) acima e com manipulações matemáticas temos a equação do comportamento do vetor peso [15].

$$\mathbf{W}_{j+1} = \mathbf{W}_j + 2\mu \left[\sum_{k=1}^{p} \alpha_k^2 k \left(n_j - \mathbf{X}_j^T \mathbf{V}_j \right)^{2k-1} \right] \mathbf{X}_j$$

$$\mathbf{W}_{j+1} - \mathbf{W}_* = \mathbf{W}_j - \mathbf{W}_* + 2\mu \left[\sum_{k=1}^{p} \alpha_k^2 k \left(n_j - \mathbf{X}_j^T \mathbf{V}_j \right)^{2k-1} \right] \mathbf{X}_j$$

$$\mathbf{V}_{j+1} = \mathbf{V}_j + 2\mu \left[\sum_{k=1}^{p} \alpha_k^2 k \left(n_j - \mathbf{X}_j^T \mathbf{V}_j \right)^{2k-1} \right] \mathbf{X}_j$$

$$\mathbf{V}_{j+1} = \mathbf{V}_j - 2\mu \left[\sum_{k=1}^{p} \sum_{i=0}^{2k-1} \alpha_k^2 k \binom{2k-1}{i} {n_j}^i (-\mathbf{X}_j^T \mathbf{V}_j)^{2k-1-i} \right] \mathbf{X}_j \tag{4.5}$$

4.5 Cálculo do Desajuste

Agora, podemos estudar o desajuste, que é uma medida de como é a saida da solução ideal. Para o cálculo do desajuste podemos concentrar a nossa atencão quando $\mathbf{V}_j \to 0$. Isso nos permite negligenciar termos de potência mais alta de $\mathbf{V}_j$[7].

Fazendo algumas manipulações matemáticas na Eq.(4.5) e lembrando que $e_j = n_j - \mathbf{X}_j^T \mathbf{V}_j$, temos:

$$\mathbf{V}_{j+1} = \mathbf{V}_j - 2\mu \left[\sum_{k=1}^{p} \sum_{i=0}^{2k-1} \alpha_k^2 k \binom{2k-1}{i} (-1)^{2k-1-i} {n_j}^i \left(\mathbf{X}_j^T \mathbf{V}_j \right)^{2k-1-i} \right] \mathbf{X}_j$$

Analizando e desenvolvendo a equação entre parênteses temos:

$$= -(\mathbf{X}_j^T \mathbf{V}_j) {n_j}^{2k-1} + (2k-1)(\mathbf{X}_j^T \mathbf{V}_j) {n_j}^{2k-2} - \frac{(2k-1)(2k-2)}{2} {n_j}^{2k-3} (\mathbf{X}_j^T \mathbf{V}_j) + ...$$

$$.... - (2k-1) {n_j}^{2k-2} (\mathbf{X}_j^T \mathbf{V}_j) + {n_j}^{2k-1}$$

$$\cong -(2k-1) {n_j}^{2k-2} (\mathbf{X}_j^T \mathbf{V}_j) + {n_j}^{2k-1}$$

O estudo do desajuste e a análise do erro acontece quando $j \to \infty$. Neste caso $E\{\mathbf{W}_j\} \to E\{\mathbf{W}_*\}$ pois temos um estimador não tendencioso. Como $\mathbf{V}_j = \mathbf{W}_j - \mathbf{W}_*$, $E\{\mathbf{V}_j\} \to 0$ quando $j \to \infty$[8]. Logo,

$$\mathbf{V}_{j+1} \cong \mathbf{V}_j + 2\mu \left\{ \sum_{k=1}^{p} \alpha_k^2 k \left[{n_j}^{2k-1} - (2k-1) {n_j}^{2k-2} (\mathbf{X}_j^T \mathbf{V}_j) \right] \mathbf{X}_j \right\} \tag{4.6}$$

Calculando as esperanças de ambos os lados da Eq.(4.6) e definindo $\mathbf{R} = E\left\{\mathbf{X}_j \mathbf{X}_j^T\right\}$, e assumindo que n_j é simétrico e iid de $\mathbf{X}_j$, podemos estudar o comportamento de $\mathbf{V_j}$ [11]:

$$E\{\mathbf{V}_{j+1}\} \cong E\{\mathbf{V}_j\} + 2\mu \sum_{k=1}^{p} \alpha_k^2 k E\left\{ {n_j}^{2k-1} - (2k-1) {n_j}^{2k-2} \mathbf{X}_j^T \mathbf{V}_j \right\} \mathbf{X}_j$$

$$E\{\mathbf{V}_{j+1}\} \cong E\{\mathbf{V}_j\} - 2\mu \sum_{k=1}^{p} \alpha_k^2 k E\left\{ {n_j}^{2k-1} \mathbf{X}_j \right\} - (2k-1) E\left\{ {n_j}^{2k-2} \mathbf{X}_j^T \mathbf{V}_j \mathbf{X}_j \right\}$$

$$E\{\mathbf{V}_{j+1}\} \cong E\{\mathbf{V}_j\} - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} E\left\{\mathbf{X}_j^T \mathbf{X}_j\right\} E\{\mathbf{V}_j\}$$

$$E\{\mathbf{V}_{j+1}\} \cong \left[I - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}\right] E\{\mathbf{V}_j\}$$

$$\frac{E\{\mathbf{V}_{j+1}\}}{E\{\mathbf{V}_j\}} \cong \left[1 - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}\right] \tag{4.7}$$

Desenvolvendo a sequência na Eq.(4.7), temos,

Se $j = 0$, temos:

$$E\{V_1\} \cong \left(1 - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}\right) E\{V_0\}$$

Se $j = 1$, temos:

$$E\{V_2\} \cong \left(1 - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}\right) E\{V_1\}$$

$$E\{V_2\} \cong \left(1 - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}\right) \left(I - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}\right) E\{V_0\}$$

$$E\{V_2\} \cong \left(1 - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}\right)^2 E\{V_0\}$$

Se $j = 2$, temos:

$$E\{V_3\} \cong \left(1 - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}\right)^3 E\{V_0\}$$

Se $j = 3$, temos:

$$E\{V_4\} \cong \left(1 - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}\right)^4 E\{\mathbf{V_0}\}$$

$$\frac{E\{\mathbf{V}_{j+1}\}}{E\{\mathbf{V}_j\}} \cong \left[1 - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}\right]$$

Como o estimador é não tendencioso, $\frac{E\{\mathbf{V}_{j+1}\}}{E\{\mathbf{V}_j\}} \to 1$, logo,

$$\left|1 - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}\right| \leq 1$$

$$-1 \leq 1 - 2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R} \leq 1$$

$$-2 \leq -2\mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R} \leq 0$$

$$0 \leq \mu \sum_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R} \leq 1$$

converge se:

$$0 \leq \mu \leq \frac{1}{\sum\limits_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\} \mathbf{R}}$$

$$0 \leq \mu \leq \frac{1}{\left(\sum\limits_{k=1}^{p} \alpha_k^2 k(2k-1) m_{2k-2} \lambda_{max}\right)} < \frac{1}{\sum\limits_{k=1}^{p} \alpha_k^2 k(2k-1) m_{2k-2}[\mathbf{R}]} \tag{4.8}$$

4.5.1 Comportamento da convergência do α_k

Calculando as experanças de ambos os lados da Eq.(4.3) e assumindo que n_j é simétrico e iid de $\mathbf{X}_j$, podemos estudar o comportamento de $\alpha_{k,j+1}$.

$$E\left\{\alpha_{k,j+1}\right\} = E\left\{\alpha_{k,j}\right\} - 2\gamma E\left\{\alpha_{k,j}\right\} E\left\{e_j^{2k}\right\}$$

$$\frac{E\left\{\alpha_{k,j+1}\right\}}{E\left\{\alpha_{k,j}\right\}} = \left[1 - 2\gamma E\left\{{e_j}^{2k}\right\}\right]. \tag{4.9}$$

Uma vez que o estimador da Eq.(4.9) é não tendencioso, logo $\frac{E\left\{\alpha_{k,j+1}\right\}}{E\left\{\alpha_{k,j}\right\}} \to 1$.
Assim,

$$|1 - 2\gamma E\left\{m_{2k}\right\}| \leq 1$$

converge se

$$0 \leq \gamma \leq \frac{1}{E\left\{m_{2k}\right\}}. \tag{4.10}$$

Onde m_q é o $q-th$ momento de n_j e λ_{max} é o máximo autovalor de $\mathbf{R}$. Como não temos, em princípio, o acesso às informações do ruído, uma condição mais natural seria $0 \leq \mu \leq \frac{1}{\sum\limits_{k=1}^{p} \alpha_k^2 k(2k-1) E\left\{{n_j}^{2k-2}\right\}[\mathbf{R}]}$ e $0 \leq \gamma \leq \frac{1}{E\{m_{2k}\}}$.
O desajuste é definido por $\mathfrak{M} = \frac{\xi_{ex}}{m_2}$, onde $\xi_{ex} = [\mathbf{R}\mathbf{Z}_j]$ é o excesso do erro quadrado médio, e $\mathbf{Z}_j = E\left\{\mathbf{V}_j \mathbf{V}_j^T\right\}$. Para analisá-los, vamos primeiro descobrir o valor do estado estacionário para $\mathbf{Z}_j$ [8]. Usando a Eq(5.6), e definindo que $a_1 = \sum\limits_{i=1}^{p}\sum\limits_{k=1}^{p} \alpha_i^2 \alpha_k^2 ik m_{2(i+k)-2}$, $a_2 = \sum\limits_{k=1}^{p} \alpha_k^2 k(2k-1) m_{2k-2}$ *e* $a_3 = \sum\limits_{i=1}^{p}\sum\limits_{k=1}^{p} \alpha_i^2 \alpha_k^2 ik(2i-1)(2k-1) m_{2(i+k)-4}$, encontramos:
Desenvolvendo o produto da Eq.(4.6) através de manipulações matemáticas e aplicando a transposta temos as sequintes equações:

$$\mathbf{V}_{j+1} \cong \mathbf{V}_j + 2\mu \left\{ \sum_{k=1}^{p} \alpha_k^2 k \left[{n_j}^{2k-1} - (2k-1){n_j}^{2k-2} (\mathbf{X}_j^T \mathbf{V}_j)\right] \mathbf{X}_j \right\}$$

$$\mathbf{V}_{j+1}\mathbf{V}_{j+1}^T \cong \left(\mathbf{V}_j + 2\mu\left\{\sum_{k=1}^{p}\alpha_k^2 k\left[{n_j}^{2k-1} - (2k-1){n_j}^{2k-2}(\mathbf{X}_j^T\mathbf{V}_j)\right]\mathbf{X}_j\right\}\right).$$

$$\left(\mathbf{V}_j + 2\mu\left\{\sum_{k=1}^{p}\alpha_k^2 k\left[{n_j}^{2k-1} - (2k-1){n_j}^{2k-2}(\mathbf{X}_j^T\mathbf{V}_j)\right]\mathbf{X}_j\right\}\right)^T$$

$$\mathbf{V}_{j+1}\mathbf{V}_{j+1}^T \cong \mathbf{V}_j\left(\mathbf{V}_j + 2\mu\left\{\sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j{n_j}^{2k-1} - \sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j(2k-1){n_j}^{2k-2}(\mathbf{X}_j^T\mathbf{V}_j)\right\}\right)^T +$$

$$+2\mu\sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j{n_j}^{2k-1}\left(\mathbf{V}_j + 2\mu\left\{\sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j n_j^{2k-1} - \sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T\mathbf{V}_j\right\}\right)^T -$$

$$-2\mu\sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T\mathbf{V}_j.$$

$$.\left(\mathbf{V}_j + 2\mu\left\{\sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j{n_j}^{2k-1} - \sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T\mathbf{V}_j\right\}\right)^T$$

Definimos A como:

$$A \cong \mathbf{V}_j\left(\mathbf{V}_j + 2\mu\left\{\sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j{n_j}^{2k-1} - \sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j(2k-1){n_j}^{2k-2}(\mathbf{X}_j^T\mathbf{V}_j)\right\}\right)^T$$

$$A \cong \mathbf{V}_j\mathbf{V}_j^T + 2\mu\sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j^T\mathbf{V}_j{n_j}^{2k-1} - 2\mu\sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j^T(2k-1){n_j}^{2k-2}\mathbf{X}_j\mathbf{V}_j\mathbf{V}_j^T$$

$$E\{A\} \cong E\left\{\mathbf{V}_j\mathbf{V}_j^T\right\} + 2\mu\sum_{k=1}^{p}\alpha_k^2 k E\left\{\mathbf{X}_j^T\mathbf{V}_j\right\}E\left\{{n_j}^{2k-1}\right\} - 2\mu\sum_{k=1}^{p}\alpha_k^2 k(2k-1)E\left\{{n_j}^{2k-2}\right\}E\left\{\mathbf{V}_j\mathbf{V}_j^T\right\}.$$

$$.E\left\{\mathbf{X}_j\mathbf{X}_j^T\right\}$$

$E\left\{{n_j}^{2k-1}\right\} = 0$ pois $2k-1$ é impar para todo k inteiro positivo e n_j é uma distribuição simétrica.

$$E\{A\} \cong \mathbf{Z}_j - 2\mu\sum_{k=1}^{p}\alpha_k^2 k(2k-1)E\left\{{n_j}^{2k-2}\right\}E\left\{\mathbf{V}_j\mathbf{V}_j^T\right\}E\left\{\mathbf{X}_j\mathbf{X}_j^T\right\}$$

$$E\{A\} \cong \mathbf{Z}_j - 2\mu\sum_{k=1}^{p}\alpha_k^2 k(2k-1)m_{2k-2}\mathbf{Z}_j\mathbf{R}$$

Onde

$$a_2 = \sum_{k=1}^{p}\alpha_k^2 k(2k-1)m_{2k-2}$$

Portanto;

$$E\{A\} \cong \mathbf{Z}_j - 2\mu a_2\mathbf{Z}_j\mathbf{R}$$

Definimos B como:

$$B \cong 2\mu \sum_{k=1}^{p} \alpha_k^2 k \mathbf{X}_j {n_j}^{2k-1} \left(\mathbf{V}_j + 2\mu \left\{ \sum_{k=1}^{p} \alpha_k^2 k \mathbf{X}_j {n_j}^{2k-1} - \sum_{k=1}^{p} \alpha_k^2 k \mathbf{X}_j (2k-1) {n_j}^{2k-2} \mathbf{X}_j^T \mathbf{V}_j \right\} \right)^T$$

$$B \cong 2\mu \sum_{k=1}^{p} \alpha_k^2 k \mathbf{X}_j {n_j}^{2k-1} \left(\mathbf{V}_j^T + 2\mu \left\{ \sum_{k=1}^{p} \alpha_k^2 k \mathbf{X}_j^T {n_j}^{2k-1} - \sum_{k=1}^{p} \alpha_k^2 k \mathbf{V}_j^T \mathbf{X}_j (2k-1) {n_j}^{2k-2} \mathbf{X}_j^T \right\} \right)$$

$$B \cong 2\mu \sum_{k=1}^{p} \alpha_k^2 k \mathbf{X}_j \mathbf{V}_j^T {n_j}^{2k-1} + 4\mu^2 \sum_{k=1}^{p} \alpha_k^2 k \mathbf{X}_j {n_j}^{2k-1} \sum_{k=1}^{p} \alpha_k^2 k \mathbf{X}_j^T {n_j}^{2k-1} -$$

$$-4\mu^2 \sum_{k=1}^{p} \alpha_k^2 k \mathbf{X}_j {n_j}^{2k-1} \sum_{k=1}^{p} \alpha_k^2 k \mathbf{V}_j^T \mathbf{X}_j (2k-1) {n_j}^{2k-2} \mathbf{X}_j^T$$

Trocando os indices da primeira somatória de k por i ,temos:

$$B \cong 2\mu \sum_{i=1}^{p} \alpha_i^2 i \mathbf{X}_j \mathbf{V}_j^T {n_j}^{2i-1} + 4\mu^2 \sum_{i=1}^{p} \alpha_i^2 i \mathbf{X}_j {n_j}^{2i-1} \sum_{k=1}^{p} \alpha_k^2 k \mathbf{X}_j^T {n_j}^{2k-1} -$$

$$-4\mu^2 \sum_{i=1}^{p} \alpha_i^2 i \mathbf{X}_j {n_j}^{2i-1} \sum_{k=1}^{p} \alpha_k^2 k \mathbf{V}_j^T \mathbf{X}_j (2k-1) {n_j}^{2k-2} \mathbf{X}_j^T$$

$$B \cong 2\mu \sum_{i=1}^{p} \alpha_i^2 i \mathbf{X}_j \mathbf{V}_j^T {n_j}^{2i-1} + 4\mu^2 \sum_{i=1}^{p} \sum_{k=1}^{p} \alpha_i^2 \alpha_k^2 ik \mathbf{X}_j \mathbf{X}_j^T {n_j}^{2i-1} {n_j}^{2k-1} -$$

$$-4\mu^2 \sum_{i=1}^{p} \sum_{k=1}^{p} \alpha_i^2 \alpha_k^2 ik(2k-1) \mathbf{X}_j \mathbf{X}_j^T \mathbf{V}_j {n_j}^{2i-1} {n_j}^{2k-2} \mathbf{X}_j^T$$

Aplicando a esperança em ambos os lados, temos:

$$E\{B\} \cong 2\mu \sum_{i=1}^{p} \alpha_i^2 i E\left\{\mathbf{X}_j \mathbf{V}_j^T\right\} E\left\{{n_j}^{2i-1}\right\} + 4\mu^2 \sum_{i=1}^{p} \sum_{k=1}^{p} \alpha_i^2 \alpha_k^2 ik E\left\{\mathbf{X}_j \mathbf{X}_j^T\right\} E\left\{{n_j}^{2i-1} {n_j}^{2k-1}\right\} -$$

$$-4\mu^2 \sum_{i=1}^{p} \sum_{k=1}^{p} \alpha_i^2 \alpha_k^2 ik(2k-1) E\left\{\mathbf{X}_j X_j^T\right\} E\left\{\mathbf{V}_j \mathbf{X}_j^T\right\} E\left\{{n_j}^{2i-1} {n_j}^{2k-2}\right\}$$

Observando que n_j é uma distribuição simétrica e $E\{{n_j}^{2t-1}\} = 0$, $\forall t$ inteiro positivo, temos;

$$E\{B\} \cong 2\mu \sum_{i=1}^{p} \alpha_i^2 i E\left\{\mathbf{X}_j \mathbf{V}_j^T\right\} E\left\{{n_j}^{2i-1}\right\} + 4\mu^2 \sum_{i=1}^{p} \sum_{k=1}^{p} \alpha_i^2 \alpha_k^2 ik E\left\{\mathbf{X}_j \mathbf{X}_j^T\right\} E\left\{{n_j}^{2(i+k)-2}\right\} -$$

$$-4\mu^2 \sum_{i=1}^{p} \sum_{k=1}^{p} \alpha_i^2 \alpha_k^2 ik(2k-1) E\left\{\mathbf{X}_j \mathbf{X}_j^T\right\} E\left\{\mathbf{V}_j \mathbf{X}^T\right\} E\left\{{n_j}^{2(i+k)-3}\right\}$$

$$E\{B\} \cong 4\mu^2 \sum_{i=1}^{p} \sum_{k=1}^{p} \alpha_i^2 \alpha_k^2 ik E\left\{\mathbf{X}_j \mathbf{X}_j^T\right\} E\left\{{n_j}^{2(i+k)-2}\right\}$$

$$E\{B\} \cong 4\mu^2 \sum_{i=1}^{p}\sum_{k=1}^{p} \alpha_i^2\alpha_k^2 ikm_{2(i+k)-2}\mathbf{R}$$

Onde

$$a_1 = \sum_{i=1}^{p}\sum_{k=1}^{p} \alpha_i^2\alpha_k^2 ikm_{2(i+k)-2}$$

Logo,

$$E\{B\} \cong 4\mu^2 a_1\mathbf{R}$$

Definimos C como:

$$C \cong -2\mu \sum_{k=1}^{p} \alpha_k^2 k\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T\mathbf{V}_j$$

$$\left(\mathbf{V}_j + 2\mu\left\{\sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j{n_j}^{2k-1} - \sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T\mathbf{V}_j\right\}\right)^T$$

$$C \cong -2\mu \sum_{k=1}^{p} \alpha_k^2 k\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T\mathbf{V}_j$$

$$\left(\mathbf{V}_j^T + 2\mu\left\{\sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j^T{n_j}^{2k-1} - \sum_{k=1}^{p}\alpha_k^2 k\mathbf{V}_j^T\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T\right\}\right)$$

$$C \cong -2\mu \sum_{k=1}^{p} \alpha_k^2 k\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T\mathbf{V}_j\mathbf{V}_j^T - 4\mu^2 \sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T\mathbf{V}_j \sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j^T{n_j}^{2k-1} +$$

$$+4\mu^2 \sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T\mathbf{V}_j \sum_{k=1}^{p}\alpha_k^2 k\mathbf{V}_j^T\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T$$

Trocando os indices da primeira somatória de k por i ,temos:

$$C \cong -2\mu \sum_{i=1}^{p} \alpha_i^2 i(2i-1){n_j}^{2i-2}\mathbf{X}_j\mathbf{X}_j^T\mathbf{V}_j\mathbf{V}_j^T - 4\mu^2 \sum_{i=1}^{p}\alpha_i^2 i\mathbf{X}_j(2i-1){n_j}^{2i-2}\mathbf{X}_j^T\mathbf{V}_j \sum_{k=1}^{p}\alpha_k^2 k\mathbf{X}_j^T{n_j}^{2k-1} +$$

$$+4\mu^2 \sum_{i=1}^{p}\alpha_i^2 i\mathbf{X}_j(2i-1){n_j}^{2i-2}\mathbf{X}_j^T\mathbf{V}_j \sum_{k=1}^{p}\alpha_k^2 k\mathbf{V}_j^T\mathbf{X}_j(2k-1){n_j}^{2k-2}\mathbf{X}_j^T$$

$$C \cong -2\mu \sum_{i=1}^{p} \alpha_i^2 i(2i-1){n_j}^{2i-2}\mathbf{X}_j\mathbf{X}_j^T\mathbf{V}_j\mathbf{V}_j^T - 4\mu^2 \sum_{i=1}^{p}\sum_{k=1}^{p}\alpha_i^2 i\alpha_k^2 k\mathbf{X}_j(2i-1)\mathbf{X}_j^T\mathbf{V}_j\mathbf{X}_j^T{n_j}^{2i-2}{n_j}^{2k-1} +$$

$$+4\mu^2 \sum_{i=1}^{p}\sum_{k=1}^{p}\alpha_i^2 i\alpha_k^2 k\mathbf{X}_j(2i-1)(2k-1)\mathbf{X}_j^T\mathbf{V}_j\mathbf{V}_j^T\mathbf{X}_j{n_j}^{2i-2}{n_j}^{2k-2}\mathbf{X}_j^T$$

Aplicando o operador Esperança Matemática e aceitando as suposições a respeito das distribuições, temos:

$$E\{C\} \cong -2\mu \sum_{i=1}^{p} \alpha_i^2 i(2i-1)E\left\{\mathbf{X}_j\mathbf{X}_j^T\right\}\left\{\mathbf{V}_j\mathbf{V}_j^T\right\}\left\{{n_j}^{2i-2}\right\} -$$

$$-4\mu^2\sum_{i=1}^{p}\sum_{k=1}^{p}\alpha_i^2\alpha_k^2 ik(2i-1)E\left\{\mathbf{X}_j\mathbf{X}_j^T\right\}E\left\{\mathbf{V}_j\mathbf{X}_j^T\right\}E\left\{{n_j}^{2i-2}{n_j}^{2k-1}\right\}+$$

$$+4\mu^2\sum_{i=1}^{p}\sum_{k=1}^{p}\alpha_i^2\alpha_k^2 ik(2i-1)(2k-1)E\left\{\mathbf{X}_j\mathbf{X}_j^T\right\}E\left\{\mathbf{V}_j\mathbf{V}_j^T\right\}E\left\{\mathbf{X}_j\mathbf{X}_j^T\right\}E\left\{{n_j}^{2i-2}{n_j}^{2k-2}\right\}$$

$$E\left\{C\right\}\cong -2\mu\sum_{i=1}^{p}\alpha_i^2 i(2i-1)E\left\{\mathbf{X}_j\mathbf{X}_j^T\right\}E\left\{\mathbf{V}_j\mathbf{V}_j^T\right\}E\left\{{n_j}^{2i-2}\right\}-$$

$$-4\mu^2\sum_{i=1}^{p}\sum_{k=1}^{p}\alpha_i^2\alpha_k^2 ik(2i-1)E\left\{\mathbf{X}_j\mathbf{X}_j^T\right\}E\left\{\mathbf{V}_j\right\}E\left\{\mathbf{X}_j^T\right\}E\left\{{n_j}^{2(i+k)-3}\right\}+$$

$$+4\mu^2\sum_{i=1}^{p}\sum_{k=1}^{p}\alpha_i^2\alpha_k^2 ik(2i-1)(2k-1)E\left\{\mathbf{X}_j\mathbf{X}_j^T\right\}E\left\{\mathbf{V}_j\mathbf{V}_j^T\right\}E\left\{\mathbf{X}_j\mathbf{X}_j^T\right\}E\left\{{n_j}^{2(i+k)-4}\right\}$$

$$E\left\{C\right\}\cong -2\mu\sum_{i=1}^{p}\alpha_i^2 i(2i-1)m_{2i-2}\mathbf{R}\mathbf{Z}_j+4\mu^2\sum_{i=1}^{p}\sum_{k=1}^{p}\alpha_i^2\alpha_k^2 ik(2i-1)(2k-1)m_{2(i+k)-4}\mathbf{R}\mathbf{Z}_j\mathbf{R}$$

Definindo

$$a_2=\sum_{i=1}^{p}\alpha_i^2 i(2i-1)m_{2i-2}$$

e

$$a_3=\sum_{i=1}^{p}\sum_{k=1}^{p}\alpha_i^2\alpha_k^2 ik(2i-1)(2k-1)m_{2(i+k)-4}$$

Temos:

$$E\left\{C\right\}\cong -2\mu a_2\mathbf{R}\mathbf{Z}_j+4\mu^2 a_3\mathbf{R}\mathbf{Z}_j\mathbf{R}$$

Somando as esperanças, encontramos o valor de $\mathbf{Z}_{j+1}$ [10]:

$$\mathbf{Z}_{j+1}\cong E\left\{A\right\}+E\left\{B\right\}+E\left\{C\right\}$$

$$\mathbf{Z}_{j+1}\cong \mathbf{Z}_j-2\mu a_2\mathbf{Z}_j\mathbf{R}+4\mu^2 a_1\mathbf{R}-2\mu a_2\mathbf{R}\mathbf{Z}_j+4\mu^2 a_3\mathbf{R}\mathbf{Z}_j\mathbf{R}$$

$$\mathbf{Z}_{j+1}\cong \mathbf{Z}_j+4\mu^2 a_1\mathbf{R}-2\mu a_2\mathbf{Z}_j\mathbf{R}-2\mu a_2\mathbf{R}\mathbf{Z}_j+4\mu^2 a_3\mathbf{R}\mathbf{Z}_j\mathbf{R}$$

$$\mathbf{Z}_{j+1}=\mathbf{Z}_j+4\mu^2 a_1\mathbf{R}-2\mu a_2\left(\mathbf{Z}_j\mathbf{R}+\mathbf{R}\mathbf{Z}_j\right)+4\mu^2 a_3\mathbf{R}\mathbf{Z}_j\mathbf{R} \tag{4.11}$$

Usando a propriedade do traço, podemos encontrar a relação $\xi_{ex}=[\mathbf{R}\mathbf{Z}_j]=[\mathbf{\Lambda}\mathbf{\Psi}_j]$, onde $\mathbf{\Lambda}$ e $\mathbf{\Psi}_j$ são matrizes diagonais e seus elementos não nulos são os autovalores de $\mathbf{R}$ e $\mathbf{Z}_j$ e o traço de uma matriz $\mathbf{A}$ é definido como $[\mathbf{A}]$, respectivamente.

Agora, vamos encontrar o valor de $\mathbf{\Psi}_j$ em estado estacionário. Isto pode ser encontrado a partir da Eq.(4.10). Pelo cálculo dos valores de $\mathbf{\Psi}_j$ quando j é suficientemente grande.

Assim, o desajuste será:

Calculando o desajuste $\mathfrak{M}$. Quando $j \to \infty$, $\mathbf{V}_j \to 0$, pois $\mathbf{V}_j = \mathbf{W}_j - \mathbf{W}_*$ como $\mathbf{Z}_j = E\left\{\mathbf{V}_j\mathbf{V}_j^T\right\}$. Temos que $\mathbf{Z}_j \to 0$ quando $j \to \infty$, temos.

$$\mathbf{Z}_{j+1} = \mathbf{Z}_j + 4\mu^2 a_1\mathbf{R} - 2\mu a_2\left(\mathbf{Z}_j\mathbf{R} + \mathbf{R}\mathbf{Z}_j\right) + 4\mu^2 a_3\mathbf{R}\mathbf{Z}_j\mathbf{R}$$

Pela propriedade do traço $[\mathbf{Z}_j\mathbf{R}] = [\mathbf{R}\mathbf{Z}_j]$, logo,

$$+4\mu^2 a_1\mathbf{R} - 2\mu a_2\left(\mathbf{R}\mathbf{Z}_j + \mathbf{R}\mathbf{Z}_j\right) + 4\mu^2 a_3\mathbf{R}\mathbf{Z}_j\mathbf{R} = 0$$

$$+4\mu^2 a_1\mathbf{R} - 2\mu a_2\mathbf{R}\mathbf{Z}_j - 2\mu a_2\mathbf{R}\mathbf{Z}_j + 4\mu^2 a_3\mathbf{R}\mathbf{Z}_j\mathbf{R} = 0$$

Como $\mathfrak{M} = \frac{\mathbf{R}\mathbf{Z}_j}{m_2}$ e substituindo na equação acima $\mathbf{R}\mathbf{Z}_j = \mathfrak{M}m_2$, teremos:

$$4\mu^2 a_1\mathbf{R} - 2\mu a_2\mathfrak{M}m_2 - 2\mu a_2\mathfrak{M}m_2 + 4\mu^2 a_3\mathfrak{M}m_2\mathbf{R} = 0$$

Dai

$$\mathfrak{M} = \frac{a_1\mu[\mathbf{R}]}{a_2 m_2 - \mu a_3 m_2[\mathbf{R}]} \tag{4.12}$$

4.6 Convergência no Tempo

Além do desajuste é importante estudar o tempo de convergência. Em geral, a constante de tempo é definida para circuitos lineares de primeira ordem e mede o tempo em que o algoritmo atinge cerca de 38% do erro inicial. A constante de tempo foi considerada $\tau_{i,LMS} = \frac{1}{2\mu_i\lambda_i}$. O mesmo valor pode ser facilmente encontrado traçando uma linha, em primeiro lugar passando por $\mathbf{W_0}$ e depois por $\mathbf{W_1}$, e encontrar a hora em que esta linha cruza o eixo de tempo [2].

Podemos usar a mesma lógica para encontrar uma constante de tempo que pode nos dar uma idéia do comportamento do algoritmo. No entanto, analisamos este comportamento só para o caso de uma única entrada no vetor de referência, embora conclusões semelhantes pode ser obtida no caso em que os elementos de $\mathbf{X_j}$ são mutuamente estatisticamente independentes. Depois de esboçar a linha, podemos ver que a constante de tempo é dada por $\tau_i = \frac{\mathbf{W}_*}{\mathbf{W_1}}$ [4]. Uma vez que temos o valor de $\mathbf{W}_*$, o que precisamos fazer é encontrar $\mathbf{W_1}$, e sabendo que $d_j = s_j + n_j$. Usamos a Eq.(4.2), e encontramos a seguinte equação:

$$a_4 = \sum_{k=1}^{p} k\alpha_k^2 \sum_{i=0}^{2k-1} \binom{2k-1}{i} E\left\{{s_j}^{2k-i}\right\} E\left\{{n_j}^{i}\right\} \tag{4.13}$$

$$\tau_{i,proposed} = \frac{1}{2\mu a_4},$$

Onde assumimos que todos os sinais são estacionários e portanto $s_p = s_q$ ou $n_p = n_q$ $\forall p, q$[5].

4.6.1 Comparação dos Algoritmos

Uma figura importante de mérito seria comparar os diferentes algoritmos, verificando o equilíbrio entre o tempo de convergência e o desajuste. Nós podemos realizar isso como Walach e Widrow fez, usando um índice $\beta(K)$, que mediu o tempo de convergência para os algoritmos, para o mesmo desajuste[14]. Usando o algoritmo Proposto como padrão, podemos definir $\beta(K) = \tau_{i,Proposed}/\tau_{i,LMS}$. Em que $\tau_{proposto}$ é o tempo de convergência do algoritmo *proposto*, e $\tau_{i,LMS}$ é o tempo de convergência do algoritmo a ser comparado. É importante notar que seria vantajoso utilizar o algoritmo proposto, e não os algoritmos quando $\beta(K) > 1$.

Podemos realizar esta comparação igualando o desajuste para a família de algoritmos LMS dado por $\chi_{LMS} = \lambda_i \mu$ e aquela dada por $\mathfrak{M}$, e encontrar uma taxa entre as duas constantes de aprendizagem diferentes, que é do $\mu_{proposto} = \frac{a_2 \mu_i}{2a_1 - a_3 \mu_i}$. A partir disso, nós encontramos,

$$\beta(K) = \frac{\mu_i \lambda_i}{\mu_{Proposto} a_4}. \tag{4.14}$$

Tabela 4.1: Desajuste teórico e experimental encontrado para dois tipos de ruído: Gaussiano e Uniforme.

μ	0.0010	0.0020	0.0030	0.0040	0.0050	0.0060	0.0070	0.0080	0.0090	0.0010
Teoretical	0.0082	0.0165	0.0261	0.0324	0.0533	0.0483	0.0722	0.0857	0.1019	0.1057
Experimental	0.0057	0.0220	0.264	0.0225	0.0256	0.0007	0.0390	0.0079	0.0066	0.0187

Desajuste e taxa de aprendizagem, quando os algoritmos estimam parâmetros escalares multiplicado por um sinal sinusoidal embutido no ruído Gaussiano. Todos os algoritmos são ajustados para o mesmo desajuste.

4.7 Resultados e Discussões

A fim de verificar a validade dos resultados teóricos encontrados aqui, em primeiro lugar, examinamos o desajuste. Há uma série de motivações para realizar essa aplicação, seja porque o número de parâmetros no algoritmo é grande ou porque há diferentes tipos de distribuição de probabilidade para o modelo do ruído na planta. Assim, para o caso de seis algoritmos que mostramos no exemplo, enquanto foram utilizados dois tipos de *pdf*: Gaussiana e uniforme. Os resultados são mostrados na Tabela 4.1 e implementado na Figura.

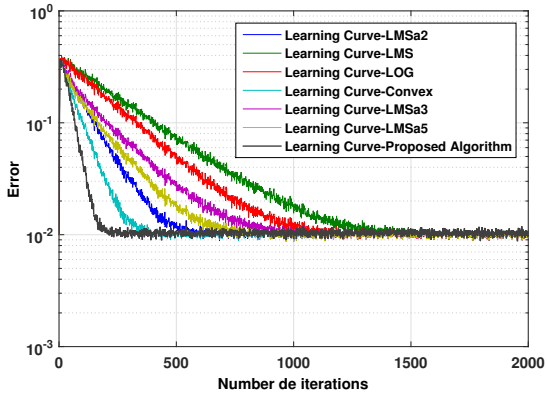

Figura 4.4: A diferença entre a estimativa de parâmetros reais com uma função do número de iterações. Os parâmetros foram calculados pelos algoritmos LMSa2, LMS, LOG, Convex, LMSa3 e LMSa5 e o algoritmo proposto quando um sinal sinusoidal está embutido no ruído Gaussiano. Todos os algoritmos são ajustados para o mesmo desajuste.

Os sinais s_k é um sinal senoidal e n_k é um sinal Gaussiano de média zero e variância unitária, enquanto $d_k = \eta \cdot s_k + n_k$. Em que η é um escalar, e a entrada de referência $x_k = s_k$ [6].

É importante comparar os diferentes tipos de algoritmos. Para este efeito, foi utilizado o algoritmo de Vitor Nascimento H.[7](que chamamos de LMSa2), algoritmo de Walach e Widrow [3](que chamamos de LMS), o algoritmo de Muhammed O.Sayin [12](que chamamos de LOG), algoritmo de Haiquan Zhao e Lu [9](que chamamos de LMSa3) e algoritmo de Bijit Kumar(que chamamos de LMSa5). Podemos realizar esta forma semelhante a Walach e Widrow, usando um índice $\beta(K)$ que mede o tempo de convergência dos algoritmos para o mesmo desajuste.

Nós executamos o algoritmo proposto com N = 5 e P = 3, e encontramos desajuste para diferentes taxas de aprendizagem, como mostrado na Figura.

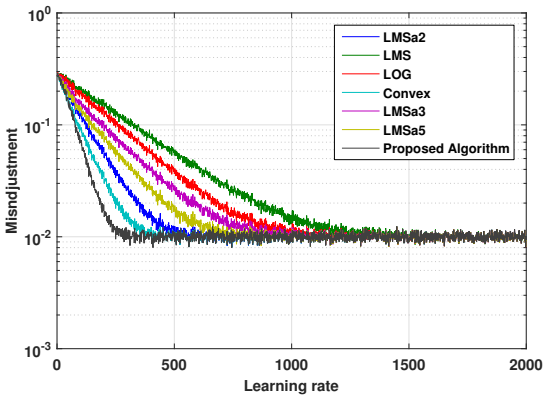

Figura 4.5: Desajuste e taxa de aprendizagem, quando os algoritmos estimam parâmetros escalares multiplicado por um sinal sinusoidal embutido no ruído Gaussiano.

Nós podemos fazer essa comparação igualando os desajuste dado por $\mu_{proposto} = \frac{a_2\mu_i}{2a_1 - a_3\mu_i}$, onde μ_i representa μ do algoritmo a ser comparado, e para encontrar uma taxa de aprendizagem entre diferentes constantes.

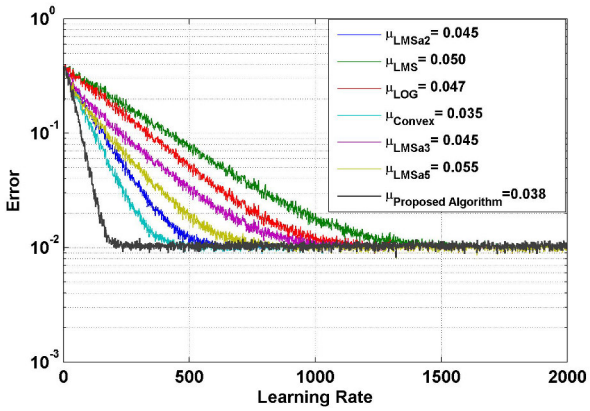

Figura 4.6: Simulando valores de μ dos diferentes algoritmos em comparação com o algoritmo Proposto, quando um escalar multiplicando por um sinal sinusoidal embutido no ruído Gaussiano para o mesmo desajuste.

Com a variação do μ, o algoritmo Proposto aumentou a sua convergência, em comparação com os outros algoritmos, que sofreram perdas e ganhos. No entanto, o algoritmo Proposto teve um pequeno aumento no seu erro. Ainda assim, o algoritmo Proposto é mais rápido quando comparado com os outros algoritmos [9].

Note que quando $\beta(K) > 1$, o algoritmo Proposto tem o melhor custo em comparação com os outros algoritmos. O algoritmo LOG [12] convergiu melhor do que o algoritmo

LMS em todas as simulações e o algoritmo convergiu com maior robustez e melhor velocidade de convergência.

Ao acessar as estatísticas do sinal de entrada d_j, ou simplesmente controlando o λ, que é independentemente da distribuição do ruído na Eq.(5.13), pode-se ver que podemos obter um melhor desempenho para o algoritmo Proposto do que o LMSa2, LMS, LOG, Convex, LMSa3 e LMSa5.

A fim de ter um resultado real para mostrar a eficácia do algoritmo Proposto, comparou-se o tempo de convergência para os mesmos dados do exemplo acima, usando o raciocínio acima para encontrar as constantes de aprendizagem para o algoritmo Proposto e os algoritmos LMSa2, LMS, LOG, Convex, LMSa3 e LMSa5, mantendo o mesmo desajuste.

Realizamos simulações onde o ruído tiveram diferentes distribuições, tais como bimodal(sinal sinusoidal) e uniforme. No entanto, como a família de algoritmos propostos por Walach e Widrow [15] é mostrado a comportar-se pior do que o LMSa2, LMS, LOG, Convex, LMSa3 e LMSa5 principalmente no caso em que o ruido é Gaussiano, mostramos na Figuras.(4.2) e (4.3) apenas o resultado para este caso.

Taxa de aprendizagem dos α_k no algoritmo Proposto. Quando $j \to \infty$ o erro $e_j^{2k} \to 0$ portanto:

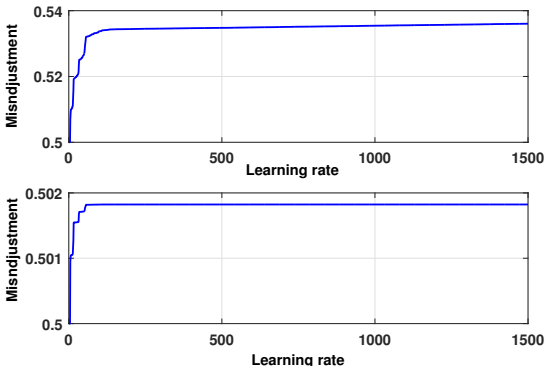

Figura 4.7: Taxa de aprendizagem dos α_k do algoritmo Proposto. com um escalar multiplicando por um sinal senoidal embutido no ruído Gaussiano para o mesmo desajuste.

Nós também podemos ver como o coeficiente α_k se comporta examinando a Fig.(4.5). Podemos ver que previu corretamente seu comportamento, como na Eq.(5.9). Com efeito, como o erro diminui, os coeficientes α_k convergem rapidamente.

4.8 Conclusão do Capítulo

Neste trabalho foi proposto um algoritmo que usa a função $\xi_j = \sum_{k=1}^{p} \alpha_k^2 E\left\{{e_j}^{2k}\right\}$ com uma função de custo age sobre o erro no filtro adaptativo. Pode ser visto nas figuras que podemos obter um melhor desempenho para o algoritmo proposto que os algoritmos LMSa2, LMS, LOG, Convex, LMSa3 e LMSa5 controlar α_k que é independente da distribuição de ruído.

Para ter resultados reais para mostrar a eficiência do algoritmo proposto, comparamos o tempo de convergência para o mesmo desajuste. Realizamos simulações de "Monte Carlo"onde o ruído tinha diferentes distribuições como Gaussiano e uniforme. No entanto, como a família de algoritmos proposta por Walach e Widrow foram mostradas a comportar-se pior do que os algoritmos LMSa2, LMS, LOG[12], Convex, LMSa3 e LMSa5 principalmente no caso em que o ruido é Gaussiano, mostramos nas figuras.(4.4),(4.5) e (4.6) apenas o resultado para este caso.

A utilização das estatisticas de alta ordem, como forma de mais informações sobre sinais, tem-se demostrado de grande importância em sistemas adaptativos. Apesar disto poucos pesquisadores tem utilizado tais técnicas, provalvelmentate devido às dificuldades matemáticas das não linearizações.

Neste trabalho, mostramos uma análise matemática para descrever a aplicação de funções não lineares, como critério aplicado sobre o erro. As equações obtidas mostram-se adequadas e através de simulações obtemos a indicação de sua veracidade.

Nas simulações o algoritmo Proposto mostrou-se mais eficiente quando comparado com os outros algoritmos da familia do LMS. Esta eficiência acentua-se devido a otimização dos parâmetros α_k. Com a implementação deste algoritmo obtivemos melhor estabilidade e maior velocidade de convergência com isso tivemos um menor erro.

5 O Algoritmo Proposto

5.1 Introdução

Neste capítulo mostraremos a teoria desenvolvida no capítulo anterior, o desenvolvimento do algoritmo Proposto, na qual escolhemos a função $\xi_j = \sum_{k=1}^{p} \alpha_k^2 E\left\{e_j^{2k}\right\}$ com aplicação sobre o erro. o nosso objetivo é que o algoritmo Proposto ajuste os pesos do combinador linear de forma tal a minimizar esta função. Mostraremos também que a superfície de desempenho oferece maior velocidade de convergência, bem como um menor desajuste [8].

5.2 A Função $\xi_j = \sum_{k=1}^{p} \alpha_k^2 E\left\{e_j^{2k}\right\}$

A função $\xi_j = \sum_{k=1}^{p} \alpha_k^2 E\left\{e_j^{2k}\right\}$ é uma função não linear, par, continua e simétrica, como está representado na figura (4.2). Esta função como podemos ver, não possui mínimo local, apenas mínimo global.

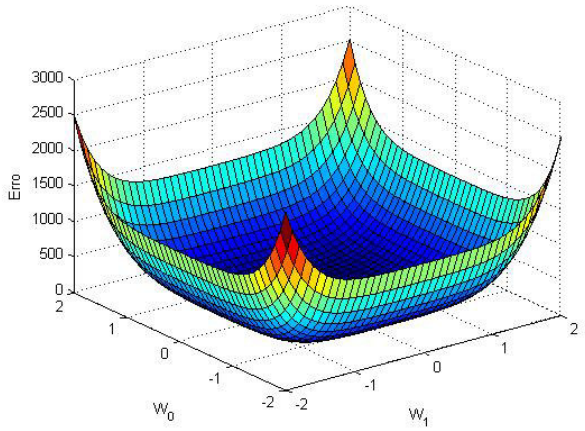

Figura 5.1: Gráfico da superfície gerada pela função $\xi_j = \sum_{k=1}^{p} \alpha_k^2 E\left\{e_j^{2k}\right\}$ e os pesos $\mathbf{W_0}$ e $\mathbf{W_1}$ variam de -2 a 2.

Uma outra característica desta função é que, para um valor fixo de α_k, podemos determinar intervalos $[-\delta, \delta]$ onde as curvas da função tem inclinações maiores que a curva da função quadrática, podemos observar esta característica na figura.(4.3).

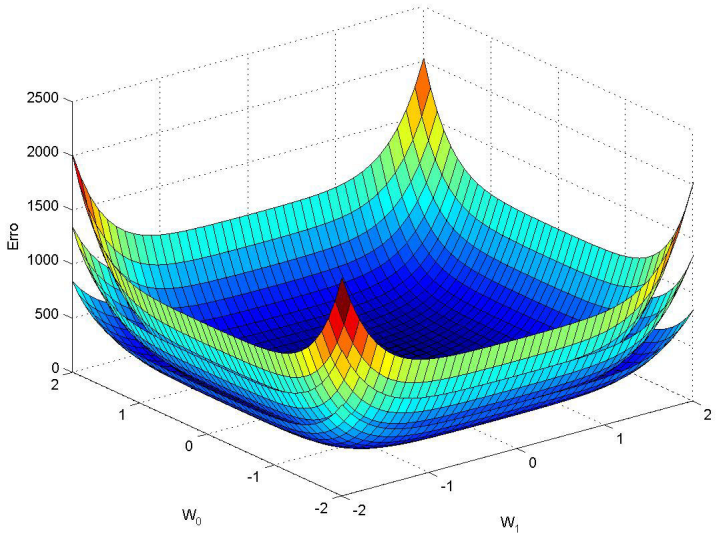

Figura 5.2: Gráfico das superfícies gerada pelo algoritmo Proposto com variação dos α_k e os pesos $\mathbf{W_0}$ e $\mathbf{W_1}$ variam de -2 a 2.

5.3 Derivação do Algoritmo Proposto

Para desenvolver o algoritmo Proposto, otimizamos o parâmetro α_k da função de custo [15].

$$\xi_j = \sum_{k=1}^{p} \alpha_k^2 E\left\{e_j^{2k}\right\}. \tag{5.1}$$

Então a cada iteração, no processo adaptativo teremos uma estimação do gradiente da forma

$$\hat{\nabla}_\alpha \xi_j = 2\left[\alpha_1 E\left\{e_j^2\right\} \alpha_2 E\left\{e_j^4\right\} .....\alpha_p E\left\{e_j^{2p}\right\}\right]^T \tag{5.2}$$

Com esta simples estimação do gradiente, podemos especificar o algoritmo Proposto que é dado por:

$$\alpha_{k,j+1} = \alpha_{k,j} - \mu\hat{\nabla}\xi_j$$

$$\alpha_{k,j+1} = \alpha_{k,j} - 2\gamma\alpha_{k,j}{e_j}^{2k},\ k = 1, 2, \ldots, p. \tag{5.3}$$

Este é o algoritmo Proposto.

O γ é uma constante que regula a velocidade e a estabilidade de adaptação.

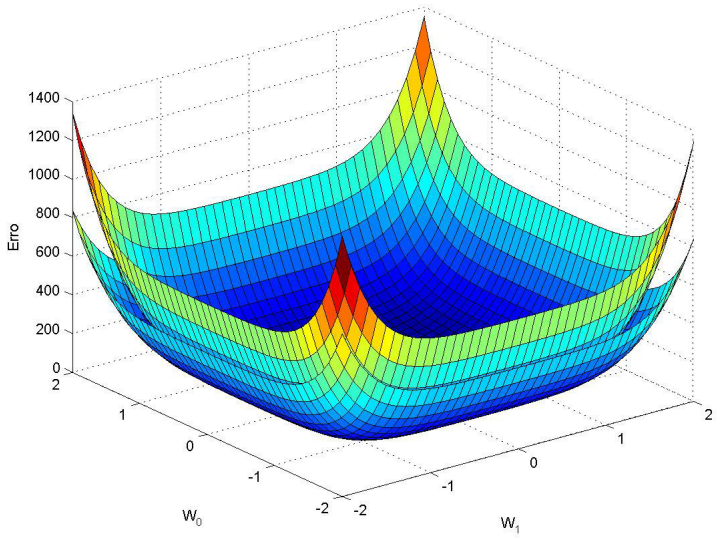

Figura 5.3: Gráfico da função $\xi_j = \sum_{k=1}^{p} \alpha_k^2 E\left\{e_j^{2k}\right\}$, onde podemos ver a maior inclinação da primeira e os pesos $\mathbf{W_0}$ e $\mathbf{W_1}$ variam de -2 a 2.

5.4 Convergência do Vetor Peso, constante de tempo e Desajuste

De acordo com a Eq.(4.10), o limite de γ para garantir a convergência do algoritmo Proposto é dado por [10]:

$$0 \leq \gamma \leq \frac{1}{E\left\{m_{2k}\right\}}. \tag{5.4}$$

A constante de tempo do algoritmo Proposto , dericadas apartir de Eq.(4.13), é

$$\tau_{i,proposto} = \frac{1}{2\mu a_4}. \tag{5.5}$$

O desajuste, de acordo com a Eq.(4.12), é determinado por:

$$\mathfrak{M} = \frac{a_1 \mu[\mathbf{R}]}{a_2 m_2 - \mu a_3 m_2[\mathbf{R}]} \tag{5.6}$$

Para fazer a comparação entre o algoritmo Proposto com o LMS e os demais algoritmos, inicialmente determinamos a relação entre os parâmetros do passo dos algoritmos, considerando desajustes iguais [13].

$$\mu_i tr[\mathbf{R}] = \frac{a_1 \mu_{Proposto} tr[\mathbf{R}]}{a_2 - \mu_{Proposto} a_3 tr[\mathbf{R}]}$$

$$\mu_{proposto} = \frac{a_2 \mu_i}{2a_1 - a_3 \mu_i}, \tag{5.7}$$

onde μ_i é o μ do algoritmo a ser comparado.

Usamos o índice β_K que tem como razão as constantes de tempo do algoritmo proposed e do LMS, substituimos estas equações e obtemos:

$$\beta(K) = \frac{\tau_{i,Proposto}}{\tau_{i,LMS}}$$

.

$$\beta(K) = \frac{\frac{1}{2\mu a_4}}{\frac{1}{2\mu_i \lambda_i}}$$

$$\beta(K) = \frac{\mu_i \lambda_i}{\mu a_4}. \tag{5.8}$$

5.5 Simulações do algoritmo Proposto

O objetivo é verificar a exatidão das equações derivadas no capitulo anterior, fizemos simulações onde comparamos o desempenho do algoritmo Proposto com o algoritmo LMS e os outros algoritmos [3].

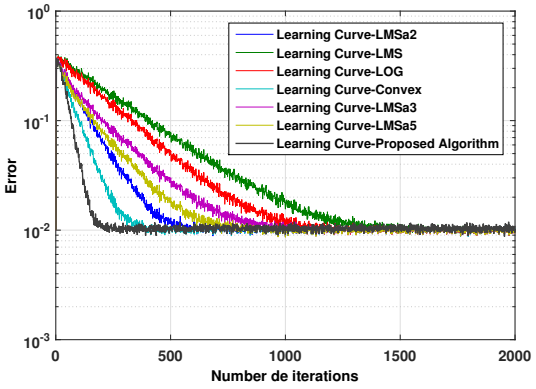

Figura 5.4: A diferença entre a estimativa de parâmetros reais com uma função do número de iterações. Os parâmetros foram calculados pelos algoritmos LMSa2, LMS, LOG, Convex, LMSa3 e LMSa5 e o algoritmo proposto quando um sinal sinusoidal está embutido no ruído Gaussiano. Todos os algoritmos são ajustados para o mesmo desajuste.

Nós executamos o algoritmo proposto com N = 5 e P = 3, e encontramos desajuste para diferentes taxas de aprendizagem, como mostrado na Figura.

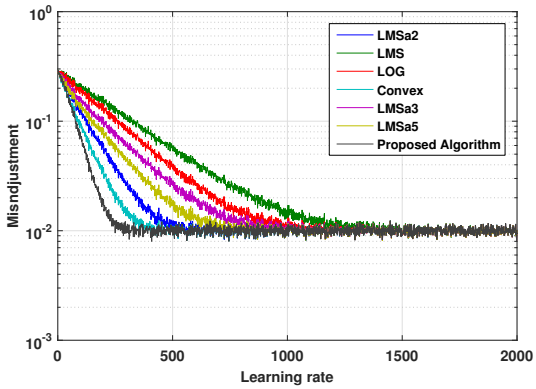

Figura 5.5: Desajuste e taxa de aprendizagem, quando os algoritmos estimam parâmetros escalares multiplicado por um sinal sinusoidal embutido no ruído Gaussiano.

Nós podemos fazer essa comparação igualando os desajuste dado por $\mu_{proposto} = \frac{a_2\mu_i}{2a_1 - a_3\mu_i}$, onde μ_i representa μ do algoritmo a ser comparado, e para encontrar uma taxa de aprendizagem entre diferentes constantes [7].

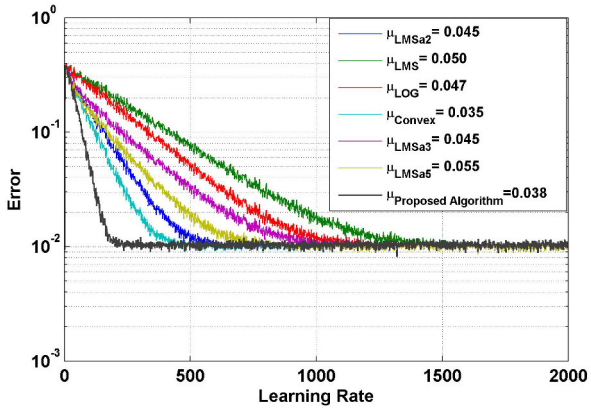

Figura 5.6: Simulando valores de μ dos diferentes algoritmos em comparação com o algoritmo Proposto, quando um escalar multiplicando por um sinal sinusoidal embutido no ruído Gaussiano para o mesmo desajuste.

Onde μ e γ são constantes que controlam a estabilidade, a taxa de convergência e de aprendizagem do algoritmo Proposto.

Taxa de aprendizagem com a otimização dos α_k do algoritmo Proposto. Quando $j \to \infty$ o erro $e_j^{2k} \to 0$ portanto:

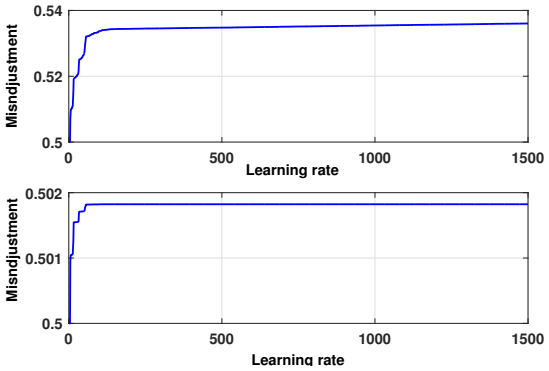

Figura 5.7: Taxa de aprendizagem dos α_k do algoritmo Proposto com um escalar multiplicando por um sinal senoidal embutido no ruído Gaussiano para o mesmo desajuste.

5.6 Conclusão do Capítulo

Neste capítulo mostramos o desenvolvimento da implementação do algoritmo Proposto que utiliza a soma do erro da função $\xi_j = \sum_{k=1}^{p} \alpha_k^2 E\left\{e_j^{2k}\right\}$, a qual queremos minimizar. Isto origina o algoritmo proposto.

Este algoritmo mostrou uma melhora no seu desempenho e diminuição do erro, quando comparado com o LMS e os demais algoritmos. Melhora esta que mostrou-se dependente do parâmetro α_k que foram otimizados, ou seja, ao aumentarmos o valor do α_k, consequentemente aumenta a inclinação da superfície de desempenho. O algoritmo Proposto aumenta a velocidade de convergência dos pesos e proporciona um menor erro, mantendo o mesmo desajuste. Dando mais ênfase à esta característica do algoritmo Proposto, temos a Tabela 4.1, os valores do índice de desempenho $\beta(K)$, obtidos para vários valores de α_k, em cada uma das duas distribuições do ruído [8].

Desajuste teórico e experimental encontrado para dois tipos de ruído: Gaussiano e Uniforme.

μ	0.0010	0.0020	0.0030	0.0040	0.0050	0.0060	0.0070	0.0080	0.0090	0.0010
Teoretical	0.0082	0.0165	0.0261	0.0324	0.0533	0.0483	0.0722	0.0857	0.1019	0.1057
Experimental	0.0057	0.0220	0.264	0.0225	0.0256	0.0007	0.0390	0.0079	0.0066	0.0187

Desajuste e taxa de aprendizagem, quando os algoritmos estimam parâmetros escalares multiplicado por um sinal sinusoidal embutido no ruído Gaussiano. Todos os algoritmos são ajustados para o mesmo desajuste.

6 Conclusão e Proposta de Continuidade

6.1 Conclusão

A utilização de estatística de alta ordem, como uma forma de obtenção de mais informações sobre sinais, tem-se demostrado de grande valia em sistemas adaptativos. Apesar disso, poucos pesquisadores tem-se utilizado de tais técnicas, provavelmente devido às dificuldades matemáticas advindas das não linearizações.

Neste trabalho, mostramos uma análise matemática para descrever a aplicação de funções não lineares, pares e continuas, as quais admitem expansão em série de Taylor, como critério aplicado sobre o erro. As equações obtidas mostraram-se adequadas e através de simulações obtemos a indicação de sua veracidade.

Nas simulações o algoritmo Proposto mostrou-se mais eficiente quando comparado com o LMS e outros algoritmos. Esta eficiência acentua-se ao aumentarmos a inclinação da superfície de desempenho devido os parâmtros α_k terem sido otimizados.

6.2 Proposta de Continuidade

Para trabalhos futuros podemos utilizar o algoritmo Proposto melhorando sua superfície de desempenho para determinação das componentes determinisísticas de um sinal de eletrocardiograma (ECG) ou de um sinal de eletroencefalograma (EEG), podemos obter um melhor desempenho quando comparamos os resultados advindos da utilização da familia de algoritmos do LMS para realizar a mesma função.

Referências Bibliográficas

[1] S. Haykin, Adaptive Filter Theory, Prentice Hall, 1991.

[2] D. Childers, J.C. Principe, Y. Ting, K. Lee, Adaptive WRLS-VFF for speech analysis, Trans. Signal Process. 3 (3) (1995) 209-213.

[3] E. Walach, B. Widrow, The least mean fourth (LMF) adaptive algorithm and its family, IEEE Trans. Inf. Theory 30 (1984).

[4] C.C.S. Silva, E.C. Santana, E. Aguiar, M. Araújo, A.K. Barros, An adaptive recursive algorithm based on non-quadratic function of the error, Signal Process. 92 (4) (2012) 853-856.

[5] J.A. Chambers, O. Tanrikulu, A.G. Constantinides, Least mean mixed- norm adaptive filtering, Electron. Lett. 30 (19) (1994) 1574-1575.

[6] T.J. Shan, T. Kailath, Adaptive algorithm with automatic gain control feature, IEEE Trans. Circuits Syst. 35 (1) (1988).

[7] Anis-ur-Rehman Anum Ali, R. Liaqat Ali, An improved gain vector to enhance convergence characteristics of recursive least squared algorithms, Int. J. Hybrid Inf. Technol. 4 (2) (2011).

[8] A.H. Sayed, Adaptive Filters, John Wiley Sons, 2008.

[9] J. Li, Y. Zheng, Z. Lin, Recursive identification of time-varying systems: the self-tuning RLS algorithm, Syst. Control Lett. 66 (2014) 104-110.

[10] A.H. Sayed, Adaptive Filters, John Wiley Sons, 2003.

[11] Behrouz Farhang-Boroujeny Adaptive Filters Theory and Applications, John Wiley Sons 2013.

[12] Muhammed O.Sayin, N. Denizean Vanli, e Suleyman Serdar Kozat , A Novel Family of Adaptative Filtering Algorithms Based on the Logarithmic Cost, IEEE Transactions on Signal Processing 2014.

[13] A. K. Barros, Jose Principe, Yoshinori Takeuchi, Carlos H. Sales, Noboru Ohnishi, An algorithm based on the even moments of the error, 8th. Workshop on Neural Networks for Signal Processing, pp. 879-885, 2003.

[14] Papoulis, A.. Probability, random variables, and stochastic processes. McGraw-Hill, 1991

[15] B. Widrow and S. D. Stearns, Adaptive Signal Processing, Engle- wood Cliffs, NJ: Prentice-Hall, 1985.