Universidade Federal do Maranhão
Centro de Ciências Exatas e Tecnologia
Programa de Pós-graduação em Engenharia Elétrica

José de Ribamar Silva Fonseca

Análise e implementação em circuito eletrônico de algoritmos adaptativos de gradiente estocástico baseado em função contínua, par e não linear

São Luís
2022

José de Ribamar Silva Fonseca

Análise e implementação em circuito eletrônico de algoritmos adaptativos de gradiente estocástico baseado em função contínua, par e não linear

Tese apresentada ao Programa de Pós-Graduacão em Engenharia Elétrica da Universidade Federal do Maranhão - UFMA, como requisito para a obtenção do título de DOUTOR em Engenharia Elétrica na área de concentração de Automação e Controle.

Orientador: Prof. Dr. Allan Kardec Duailibe Barros Filho

Coorientador: Prof. Dr. Luis Cláudio Oliveira Silva

São Luís

2022

I

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Diretoria Integrada de Bibliotecas/UFMA

Silva Fonseca, José de Ribamar.
Análise e implementação em circuito eletrônico de algoritmos adaptativos de gradiente estocástico baseado em função contínua, par e não linear / José de Ribamar Silva Fonseca. - 2022.
84 p.

Coorientador(a): Luís Cláudio Oliveira Silva.
Orientador(a): Allan Kardec Duailibe Barros Filho.
Tese (Doutorado) - Programa de Pós-graduação em Engenharia Elétrica/ccet, Universidade Federal do Maranhão, São Luis-MA, 2022.

1. Algoritmos de Gradiente Estocástico. 2. Filtros Adaptativos. 3. FPGA. 4. Implementação em Hardware Digital. 5. Nanossatélite. I. Duailibe Barros Filho, Allan Kardec. II. Oliveira Silva, Luís Cláudio. III. Título.

José de Ribamar Silva Fonseca

Análise e implementação em circuito eletrônico de algoritmos adaptativos de gradiente estocástico baseado em função contínua, par e não linear

Tese apresentada ao Programa de Pós-Graduacão em Engenharia Elétrica da Universidade Federal do Maranhão - UFMA, como requisito para a obtenção do título de DOUTOR em Engenharia Elétrica na área de concentração de Automação e Controle.

Aprovado em 2022

BANCA EXAMINADORA

Prof. Dr. Vicente Leonardo Paucar Casas

Membro da Banca Examinadora

Prof. Dr. Ewaldo Santana

Membro da Banca Examinadora

Prof. Dr. Inocêncio Sanches dos Santos Neto

Membro da Banca Examinadora

Profa. Dra. Marta de Oliveira Barreiros

Membro da Banca Examinadora

Prof. Dr. Luís Cláudio Oliveira

Coorientador

Prof. Dr. Allan Kardec Duailibe Barros Filho

Orientador

Dedicatória

À minha esposa Raquel, à minha filha Cecília e ao meu filho Caio.

Resumo

Os filtros adaptativos têm resolvido muitos problemas de processamento de sinais, além de constituírem parte fundamental do aprendizado de máquina. O estudo da filtragem adaptativa foi impulsionado com o desenvolvimento do algoritmo Least Mean Square (LMS) em 1960. Desde então outros algoritmos adaptativos têm surgido com um desempenho superior ao algoritmo LMS em relação ao desajuste e à taxa de convergência. Vários algoritmos têm surgido de modo a melhorar o desempenho do algoritmo LMS. Entre eles, os algoritmos adaptativos baseados no gradiente estocástico com base em critérios de erro não quadrático médio, foram desenvolvidos e analisados por Douglas e Meng, estes algoritmos proporcionam um desempenho superior em relação aos seus equivalentes de gradiente estocástico quando se analisa em relação ao desajuste e a velocidade de convergência. Estes algoritmos adaptativos constituem uma importante família de algoritmos. Quando se implementa esses algoritmos em dispositivo de hardware, tais como DSP, Microcontroladores e FPGA, estes são implementados em precisão finita. Alguns efeitos nessa implementação podem afetar o seu desempenho. Em última análise podem levar à divergência devido aos erros de quantização. Essa importante família de algoritmos adaptativos de erro não quadrático médio, foi analisada em 1994 por Douglas e Meng. Analisou-se a constante de tempo e o desajuste. Os resultados encontrados dessa análise tem apresentado divergências quando comparamos com os resultados experimentais. Nesse sentido este trabalho tem o objetivo de analisar essa família de algoritmos em precisão finita, especificamente desenvolvemos uma nova metodologia para o cálculo do desajuste em ponto fixo, assim como também realizamos uma implementação em hardware do tipo FPGA para resolução de problema em nanossatélite.

Palavras-chaves: Filtros Adaptativos, Algoritmos de Gradiente Estocástico, Precisão Finita, Quantização, Implementação em Hardware digital, FPGA, Nanossatélite.

Abstract

Adaptive filters have solved many signal processing problems, in addition to being a fundamental part of machine learning. The study of adaptive filtering was boosted with the development of the Least Mean Square (LMS) algorithm in 1960. Since then, other adaptive algorithms have emerged with a superior performance to the LMS algorithm in terms of misfit and convergence rate. Several algorithms have emerged in order to improve the performance of the LMS algorithm. Among them, adaptive algorithms based on stochastic gradient based on non-square mean error criteria, were developed and analyzed by Douglas and Meng, these algorithms provide superior performance over their stochastic gradient counterparts when analyzed against misfit and the speed of convergence. These adaptive algorithms constitute an important family of algorithms. When implementing these algorithms in hardware devices such as DSPs, Microcontrollers and FPGAs, they are implemented in finite precision. Some effects in this implementation can affect its performance. Ultimately they can lead to divergence due to quantization errors. This important family of adaptive non-mean-square error algorithms was analyzed in 1994 by Douglas and Meng. The time constant and the maladjustment were analyzed. The results found from this analysis have presented divergences when compared with the experimental results. In this sense, this work aims to analyze this family of algorithms in finite precision, specifically we developed a new methodology for the misfit calculation, as well as an FPGA hardware implementation for problem solving and nanosatellite.

Agradecimentos

Ao professor Allan Kardec Duailibe Barros Filho, pela confiança, orientação e dedicação com que encaminhou este trabalho.

Ao professor Luís Cláudio Oliveira, pela empenho e disponibilidade na coorientação deste trabalho.

Agradeço aos professores e amigos do Laboratório da Informação Biológica-PIB.

À toda minha família e amigos.

Sumário

Lista de Figuras **8**

Lista de Tabelas **11**

1 Introdução **13**

1.1 Problema 15

1.2 Objetivos 16

1.3 Motivações 17

1.4 Estado da Arte 18

2 Processamento de Sinais e Filtragem Adaptativa **20**

2.1 Características dos Filtros Adaptativos 21

2.2 Aplicações 24

3 Análise do desajuste de uma família de algoritmos adaptativos de gradiente estocástico **27**

3.1 Introdução 27

3.2 Metodologia 31

4 Processamento de Sinal Digital e os Efeitos da Precisão Finita **35**

4.1 Erros de Quantização no processo de conversão para o sinal digital 36

4.2 Filtragem Adaptativa em Precisão Finita 38

4.3 Implementação em Dispositivo de Hardware Digital 40

5 Análise em Precisão Finita de uma Família de Algoritmos Adaptativos Baseado no Erro não quadrático médio **41**

5.1 Análise de Adaptação 41
5.1.1 Pressupostos 2 42
5.2 Análise da constante de tempo 43
5.3 Análise da Convergência 47

6 Simulações e Resultados **50**
6.1 Dados da Simulação 50
6.2 O algoritmo LMMN 50
6.3 O algoritmo WEM 52
6.4 O algoritmo SIGMOIDAL 54
6.5 O algoritmo LMF 56

7 Circuito Eletrônico baseado em algoritmo Sigmoidal de ponto fixo aplicado em Nanossatélite **59**
7.1 Fundamentos da implementação 59
7.1.1 Tecnologia FPGA 59
7.1.2 Nanossatélite na perspectiva do padrão CubeSat 60
7.1.3 O algoritmo Sigmoidal de Ponto Fixo 62
7.2 Realização do algoritmo Sigmoidal de Ponto Fixo em FPGA 62
7.2.1 Geração do circuito em VHDL 63
7.2.2 Implementação do algoritmo em FPGA 65
7.2.3 Resultados da implementação 70

8 Discussão **74**

9 Conclusões **77**

A Apêndice **78**

B Apêndice **79**

Referências Bibliográficas **80**

Lista de Figuras

2.1 Configuração geral de um filtro adaptativo. Em que k representa o número de iteração, $\mathbf{x}_k$ designa o sinal de entrada, y_k é o sinal de saída do filtro adaptativo, e d_k define o sinal desejado. O sinal de erro e_k é dado por $d_k - y_k$. O sinal de erro é então utilizado para formar uma função de desempenho (ou objetivo) que é exigida pelo algoritmo de adaptação para determinar a atualização apropriada dos coeficientes do filtro $\mathbf{w}_k$. 21

2.2 Modelo de um filtro adaptativo discreto aplicado ao problema de identificação de sistema desconhecido, onde z^{-I} representa uma unidade de atraso. 22

2.3 Porção da superfície quadrática tridimensional juntamente com alguns contornos. O erro quadrático médio está plotado na vertical, $w0$ e $w1$ variam de -1 a 1. 23

2.4 Identificação de sistema 24

2.5 Modelagem Inversa 25

2.6 Predição Linear 25

2.7 Cancelamento de Ruído, onde n_{1k} e n_{2k} são ruídos conhecidos. 26

4.1 Representação esquemática de níveis de quantização para o caso de $b = 2$ bits e $\triangle = L/4$. Consideramos um valor L limiar. Usando b bits, o intervalo de $[0, L]$ pode ser dividido em 2^b níveis de largura $\triangle = L/2^b$ cada. Se, além disso, usamos 1 (um) bit de sinal, os resultantes $(b + 1)$ bits divide o intervalo $[-L, L]$ em 2^{2b} níveis. Aqui assumimos operação de arredondamento. 37

4.2 Variância devido à quantização com aritmética de ponto fixo. Para $C = 1$ número de *bits* variando entre 7 bits à 32 bits 38

5.1 Identificação de sistema implementado em precisão finita. Onde $\mathbf{x}_k$ designa o sinal de entrada. Os blocos Q_d e Q_c são quantizadores de b bits. Assim, o quantizador, Q_d, usa b_d bits para os dados de entrada e Q_c quantiza os coeficientes em b_c bits. O sinal de saída y_{Qk} indica a forma quantizada. O sinal desejado definido por d_k do sistema desconhecido e d_{Qk} sua forma quantizada. O ruído de medição n_k e o sinal de erro quantizado e_{Qk}. Os coeficientes $\mathbf{w_k}$ do sistema desconhecido e $\mathbf{w_{Qk}}$ do filtro adaptativo. 41

6.1 Curva de Aprendizagem dos Algoritmos LMMN Quantizado Sob Critério de Erro Geral, $\mu = 0,01$ 52

6.2 Curva de Aprendizagem dos Algoritmos WEM Quantizado Sob Critério de Erro Geral, $\mu = 0,01$ 54

6.3 Curva de Aprendizagem dos Algoritmos SIGMOIDAL Quantizado Sob Critério de Erro Geral, $\mu = 0,01$ 56

6.4 Curva de Aprendizagem dos Algoritmos LMF Quantizado Sob Critério de Erro Geral, $\mu = 0,005$ 58

7.1 Subsistemas de um CubeSat 61

7.2 Função convertida em VHDL usando aritmética de ponto fixo 64

7.3 (a) Funcionamento básico do método proposto; (b) Quantidade de pinos empregada na comunicação e controle entre FPGA e o MCU 65

7.4 Diagrama esquemático do circuito proposto 66

7.5 Visão RTL do circuito gerado a partir do projeto criado 68

7.6 Placa de desenvolvimento para o experimento 69

7.7 Relatório da síntese fornecido pelo Altera Quartus II 70

7.8 Placa de prototipagem confeccionada em funcionamento 71

7.9 Resposta do filtro aplicado ao armazenamento 72

7.10 Placa de prototipagem confeccionada em funcionamento 73

A.1 Diagrama esquemático da placa de circuito impresso 78

B.1 Pedido de Registro de topografia de circuito eletrônico 79

Lista de Tabelas

6.1 Desajuste do algoritmo LMMN, com potência do ruído $n_k = 10^{-3}$, Passo de Adaptação μ, Desajuste Experimental M_{exp}, Desajuste proposto M_{p1} com Erro Relativo de $\Delta 1$ (%) e Desajuste proposto por [10] M_{p2} com Erro Relativo de $\Delta 2$ (%) 51

6.2 Desajuste do algoritmo LMMN em sua forma quantizada, com potência do ruído $n_k = 10^{-3}$, C=400, número de Bits (Nº de bits) implementado, passo de adaptação μ_Q, desajuste experimental quantizado M_{Q-exp} e desajuste proposto quantizado M_Q. 51

6.3 Desajuste do algoritmo WEM, com potência do ruído $n_k = 10^{-3}$, Passo de Adaptação μ, Desajuste Experimental M_{exp}, Desajuste proposto M_{p1} com Erro Relativo de $\Delta 1$ (%) e Desajuste proposto por [10] M_{p2} com Erro Relativo de $\Delta 2$ (%) 53

6.4 Desajuste do algoritmo WEM em sua forma quantizada com potência do ruído $n_k = 10^{-3}$, C=400, número de Bits (Nº de bits) implementado, passo de adaptação μ_Q, desajuste experimental quantizado M_{Q-exp} e desajuste proposto quantizado M_Q. 53

6.5 Desajuste do algoritmo SIGMOIDAL, com potência do ruído $n_k = 10^{-3}$, Passo de Adaptação μ, Desajuste Experimental M_{exp}, Desajuste proposto M_{p1} com Erro Relativo de $\Delta 1$ (%) e Desajuste proposto por [10] M_{p2} com Erro Relativo de $\Delta 2$ (%) 55

6.6 Desajuste do algoritmo SIGMOIDAL em sua forma quantizada, com potência do ruído $n_k = 10^{-3}$, C=400, número de Bits (Nº de bits) implementado, passo de adaptação μ_Q, desajuste experimental quantizado M_{Q-exp} e desajuste proposto quantizado M_Q. 55

6.7 Desajuste do algoritmo LMF, com potência do ruído $n_k = 10^{-1}$, Passo de Adaptação μ, Desajuste Experimental M_{exp}, Desajuste proposto M_{p1} com Erro Relativo de $\Delta 1$ (%) e Desajuste proposto por [10] M_{p2} com Erro Relativo de $\Delta 2$ (%) 57

6.8 Desajuste do algoritmo LMF em sua forma quantizada, com potência do ruído $n_k = 10^{-1}$, C=8,número de Bits (Nº de bits) implementado, passo de adaptação μ_Q, desajuste experimental quantizado M_{Q-exp} e desajuste proposto quantizado M_Q. 57

1 Introdução

O desenvolvimento no campo de processamento de sinais, nos últimos trinta anos, tem recebido contribuições significativas, possuindo diversos objetos de interesse que podem incluir sons, imagens, sinais de telecomunicações, sinais biológicos etc. Na maioria dos problemas em situações práticas, envolve o processamento de dados contaminados por ruído, de forma a extrair alguma informação sobre o sinal de interesse, estas implementações podem ser na forma analógica e/ou na digital. Especificamente o processamento de sinal digital tem avançado significativamente a partir do desenvolvimento tecnológico de circuitos digitais, que apresentam vários atrativos devido ao baixo custo, confiabilidade, precisão, tamanhos físicos pequenos e flexibilidade. Atualmente existem diversos dispositivos que podem ser usados no processamento de sinal digital, como DSP, microcontroladores e FPGA [8], [12], [20], [41], [23] e [15].

Existem diversas técnicas para processamento de sinais, entre elas a filtragem adaptativa tem sido fortemente utilizada, geralmente formado por circuitos eletrônicos e com o avanço desta teoria e da tecnologia dos circuitos integrados torna-se cada vez mais sofisticada. Filtros adaptativos são dispositivos auto-ajustáveis, baseados em algoritmos recursivos, que modificam seus parâmetros de acordo com critérios pré-estabelecidos. Em ambiente estacionário, acompanham a solução ótima do problema. Em ambiente não estacionário, acompanham as modificações do sinal envolvido. [8], [20], [15] e [12].

A filtragem adaptativa constitui atualmente uma ferramenta importante no processamento de sinais, além de formarem os exemplos mais simples de algoritmos no campo de aprendizado de máquinas. Tem conseguido êxito em numerosas aplicações, tais como modelagem, equalização, controle, cancelamento de ruído e outras, especialmente quando é necessário processar sinais de ambientes com estatísticas desconhecidas que variam com o tempo [33]. O estudo da filtragem adaptativa foi impulsionado com o desenvolvimento do algoritmo de gradiente estocástico Least Mean Square (LMS) em 1960, baseado na função de custo do erro quadrático médio [8]. Desde então, novos algoritmos foram desenvolvidos, baseados em outras técnicas, entre eles, alguns bem conhecidos,

como o algoritmo de Kalman [5] e o Recursive Least Square (RLS) [8].

Novos algoritmos de filtragem adaptativa de gradiente estocástico, baseados em critérios do erro não quadrático médio tem sido sugerido por outros autores, como forma de melhorar o desempenho desses algoritmos em ambientes estatísticos particulares [36], [45], [10]. Todavia, pouca atenção foi dada a análise de algoritmos de filtragem adaptativa de gradiente estocásticos, baseados em critérios de erro não quadrático médio. Talvez devido à complexidade computacional e dificuldade matemática associada à sua análise, não descartando seu potencial na resolução de problemas práticos de filtragem adaptativas. No entanto, a exploração de suas propriedades levou a importantes descobertas nesta área, por exemplo, o algoritmo LMS limita-se a uma relação difícil entre o erro final de desajuste e o tempo de convergência, um fato que forçou alguns pesquisadores a recorrer a métodos dispendiosos em termos computacionais [5].

Assim, usando funções não lineares do erro, em algoritmos de gradiente estocástico, alguns autores mostraram que eles produzem desempenho geral aprimorado [10] e [4]. Desde então [10] e [4] tem sido referencial teórico para diversos trabalhos na área [38], [18], [3] e [35] e também citado por diversos autores [1], [49],[44] e [31]. Nas funções lineares e não-lineares do erro, os parâmetros de adaptação analisados para o comportamento de convergência foram a constante de tempo e o desajuste. E. Santana et al (2017), após minuciosa análise matemática para descrever as condições de convergência dos algoritmos alcançaram um resultado surpreendente superando a análise realizada por [10] e [4] na aproximação da constante de tempo. Desde então novas análises e classes de algoritmos baseados nesta técnica vem surgindo como mostrado em [21], [11], [40] e [48].

O desajuste teórico analisado e proposto por [10], em termos experimentais apresenta resultados aproximados, quando o tamanho de passo de adaptação é muito pequeno e resultados divergentes quando aumentasse o tamanho de passo de adaptação destes algoritmos, tão logo, como forma de enriquecer a discussão nesse assunto, propomos uma metodologia alternativa para o cálculo do desajuste e uma análise em precisão finita tanto para constante de tempo como para o desajuste para esses tipos de algoritmos.

Neste trabalho, analisamos o efeito da precisão finita em algoritmos adaptativos de gradiente estocástico baseado em função contínua, par e não linear, desenvolvemos uma expressão para o desajuste. Também embarcamos um algoritmo em dispositivo de hardware digital.

1.1 Problema

Em termos práticos, os algoritmos adaptativos são implementados nos dispositivos de hardware digital em precisão finita, e muitas vezes com aritmética de ponto fixo. A penalidade no desempenho de alguns algoritmos adaptativos incorridos com o resultado da implementação em precisão finita tem sido analisadas em estudos relacionados a filtragem adaptativa. O seu comportamento pode obter resultados ou efeitos indesejáveis devido aos erros de quantização introduzidos nos cálculos envolvidos no processamento adaptativo desses filtros [20].

Uma etapa tão importante quanto desenvolver os algoritmos na teoria é conhecer os erros de quantização que podem degradar o desempenho de um filtro adaptativo de várias maneiras. Por exemplo, eles podem afetar a estabilidade do filtro e em última análise conduzir à sua divergência. Também podem degradar o desempenho em estado estacionário do filtro fazendo-o atingir um erro quadrático médio maior do que o esperado a partir de uma análise de precisão infinita [19].

Outra etapa tão importante é a implementação e realização desses algoritmos em hardware, onde os desenvolvedores de circuitos podem esbarrar em diversos problemas, como por exemplo, em problemas de transformação do algoritmo para linguagem de hardware. É frequente a linguagem de hardware recusar muitas funções além de outras dificuldades que podem ocorrer, fazendo a implementação não atingir seu objetivo na resolução de problemas [29] e [41].

Devido à crescente utilização de sistemas reconfiguráveis em nanossatélites do tipo cubesat, bem como a necessidade de implementação de filtros adaptativos para aplicações aeroespaciais, haja visto que os computadores de bordo dos nanossatélites possuem recursos computacionais limitados. Desta forma, uma placa de circuitos baseada em FPGA pode auxiliar em tarefas de filtragem adaptativas por ser mais rápidas e capazes de executar operações em paralelo, um produto deste tipo é relevante, desde que superando as limitações de implementações e aplicação [47].

1.2 Objetivos

- **Objetivo Geral:**
  - Desenvolver uma nova metodologia para o cálculo do desajuste em ponto fixo de uma família de algoritmos adaptativos de gradiente estocástico. Desenvolver um algoritmo Sigmoidal de Ponto Fixo. Implementar o algoritmo de Ponto Fixo em hardware.

- **Objetivo Específico:**
  - Analisar uma família de algoritmos adaptativos de gradiente estocástico baseado no erro não quadrático médio, recorrendo a um novo método de análise para o desajuste;
  - Propor uma expressão matemática para o desajuste dos algoritmos de gradiente estocástico;
  - Derivar expressões aproximadas para o desajuste e observar o comportamento desses algoritmos em precisão finita observando a degradação desse filtro adaptativo em ambientes de desempenho quantizados.
  - Analisar a constante de tempo e o desajuste em precisão finita;
  - Simular em ambiente de teste os algoritmos LMMN, WEM, SIGMOIDAL e LMF em ponto fixo;
  - Comparar os resultados teóricos com os simulados;
  - Implementar o algoritmo adaptativo Sigmoidal de ponto Fixo em hardware do tipo FPGA;
  - Desenvolver e implementar uma arquitetura de Hardware utilizando FPGA para filtragem adaptativa em sinal de rádio de nanossatélite do tipo Cubesat, utilizando o algoritmo Sigmoidal de Ponto Fixo.

1.3 Motivações

Atualmente a filtragem adaptativa é uma ferramenta importante para o processamento de sinais e são fundamentais na aprendizagem de máquinas. Analisar uma família de algoritmos de gradiente estocástico, através de uma nova metodologia para o desajuste pode enriquecer a discussão na área contribuindo para a disseminação desses algoritmos. Os algoritmos adaptativos quando implementados em dispositivos de hardware para o processamento de sinais digitais, são implementados em precisão finita. Nestas condições o comportamento e o desempenho podem obter resultados ou efeitos indesejáveis devido aos erros de quantização introduzidos nos cálculos envolvidos no processamento adaptativo desses filtros ou arredondamento. Assim, é consideravelmente relevante estudar o desempenho desses algoritmos adaptativos de gradiente estocástico baseado em erro geral não médio quadrático em precisão finita, sendo que estes algoritmos abrangem uma importante família de algoritmos adaptativos.

A necessidade desse desenvolvimento analítico se dá pela importância de criar um modelo teórico e experimental que poderá ser utilizado na resolução de diversos problemas de processamento de sinais. Deste modo, a ideia de produzir um produto final para resolução de problema baseado neste estudo, ou seja, embarcar um sistema adaptativo em hardware, motiva o prosseguimento desse estudo. Nesse aspecto de sistemas adaptativos embarcados e devido à demanda de desempenho dos algoritmos de processamento de sinais, os sistemas embarcados baseados em microcontroladores podem apresentar gargalos de processamento. O uso de ASICs (application specific integrated circuits) pode atender os requerimentos de processamento, contudo, a arquitetura fixa e alto custo de implementação os tornam menos flexíveis para implementar sistemas de aprendizado customizáveis. Como uma alternativa, as FPGAs tornam-se a opção mais balanceada entre desempenho e flexibilidade para implementarem sistemas embarcados inteligentes.

A exploração espacial é um campo de interesse crescente por parte de instituições de pesquisa, universidades e empresas ao redor do mundo, impulsionado pela criação de padrões de nanossatélites, vem possibilitando o desenvolvimento de novas aplicações para os satélites de pequeno porte à um custo cada vez mais reduzido. Essa área de nanossatélites devido a sua importância e outras características, se torna pertinente para o desenvolvimento e aplicação do trabalho.

1.4 Estado da Arte

Os trabalhos pioneiros na filtragem adaptativa em precisão finita analisam alguns algoritmos adaptativos, mostrando o desempenho e a degradação desses algoritmos, como, por exemplo, no algoritmo RLS (*Recursive Least Square*) convencional, este pode ser sensível a erros de precisão finita e divergir como resultado no seu desempenho [37] em oposição ao algoritmo LMS (*Least Mean Square*) [19]. Isto porque para filtros do tipo LMS, erros de precisão finita em algoritmos baseados no gradiente da função de custo, são mais significativos do que os erros de precisão finita durante a fase transitória em algoritmos não baseado nesta técnica. No entanto, em estado estacionário, um significativo excesso de erro quadrático médio pode ser formado [42].

O trabalho de Douglas e Meng [10], na filtragem adaptativa, tem sido referencial teórico para diversos trabalhos na área de processamento de sinais. Principalmente devido a vantagem desse tipo de superfície, quando comparada a quadrática, é que ela naturalmente produz uma convergência mais rápida com menor desajuste, sendo indicada por alguns autores para utilização na resolução de diversos problemas de processamento de sinais [36], [3], [10], [45] e [17]. A partir deste trabalho novos algoritmos surgiram com expectativa de possuírem um desempenho melhor em relação ao pioneiro LMS. Algoritmos de filtragem adaptativa com base em erro não quadrático médio tem sido sugeridos por vários autores, quer como uma forma de melhorar o desempenho de adaptação destes algoritmos para ambientes estatísticos particulares, ou como forma de aprimorar o cálculo de processamento em hardware [10].

Apesar da filtragem adaptativa ter sido fortemente utilizada como técnicas de processamento de sinais, e com os avanços desta teoria e da tecnologia dos circuitos integrados. A utilização desses novos algoritmos tem caminhado a passos lentos, principalmente pela dificuldade em tornar estes algoritmos proveitosos na resolução de problemas práticos, muitas vezes devido a problemas teóricos e outras vezes devido a problemas de implementação em hardware [12]. Com isso, os algoritmos do tipo LMS auferem espaços nos sistemas embarcados, principalmente devido ao melhoramento em suas técnicas e estruturas quando implementado em precisão finita, ganhando novas modelagens e arquiteturas eficientes em hardware [25] e [27].

O esforço para construção e testes de sistemas adaptativos embarcados que apresentam melhores resultados, necessita de circuitos especialmente construído para uma finalidade adequada, como está demonstrado na literatura [28]. Uma arquitetura eficiente, rápida e barata é a FPGA, devido ao seu paralelismo, torna-se um ambiente ideal para implementação de sistemas adaptativos em processamento de sinais [28]. Uma aplicação dedicada para construção do circuito adaptativo é a filtragem de sinal de rádio, utilizada por satélite miniaturizado. Esta aplicação normalmente utiliza componentes eletrônicos e materiais acessíveis aos pesquisadores e a toda comunidade acadêmica um meio prático para o desenvolvimento de nanossatélite. Esse padrão, denominado Cubesat, surgiu do esforço colaborativo dos pesquisadores Jordi Puig-Suari e Bob Twiggs da Universidade Politécnica do Estado da Califórnia (Cal Poly) e da Universidade de Stanford, respectivamente [43].

O problema do sinal de rádio com ruído em nanossatélites, consiste em uma alteração de alguma característica do sinal transmitido por efeito de um outro sinal exterior ao sistema de transmissão, ou gerado pelo próprio sistema de transmissão [24] e [13]. Estes sinais são de natureza aleatória, não sendo possível prevê o seu valor num instante de tempo futuro. Os efeitos indesejados dos ruídos no desempenho do sistema de transmissão podem ser minimizados através da utilização de técnicas de projetos dos circuitos e/ou através da filtragem, [2] e [24].

2 Processamento de Sinais e Filtragem Adaptativa

A filtragem adaptativa tem sido fortemente utilizada como técnica para o processamento de sinais digitais, e com os avanços desta teoria e da tecnologia dos circuitos integrados torna-se cada vez mais inovadoras e aplicadas em diversos problemas de filtragem [27]. Filtros adaptativos são dispositivos auto-ajustáveis, baseados em algoritmos recursivos, que modificam seus parâmetros de acordo com critérios pré-estabelecidos. Em ambiente estacionário, acompanham a solução ótima do problema. Em ambiente não estacionário, acompanham as modificações do sinal envolvido [46]. Os filtros adaptativos são classificados em dois grupos principais: lineares e não lineares. Filtros adaptativos lineares computam um valor estimado da resposta desejada utilizando uma combinação linear dos sinais de entrada, de outra forma, diz-se que os filtros adaptativos são não lineares [12].

Um trabalho pioneiro no campo de projetos de filtros foi desenvolvido por Norbert Wiener em 1949. Seus esforços tornaram possível o projeto de filtros lineares para eliminação de ruído para predição e suavizamento de sinais estatisticamente estacionários. Outro trabalho mais recente de Kalman e Bucy (1961) conduziu à concepção de filtros lineares de tempo variável para sinais não estacionários. Para tais sinais, filtros de Kalman-Bucy podem entregar um desempenho melhor do que os filtros de Wiener [9].

As aplicações típicas dos filtros adaptativos são: predição, modelamento, cancelamento de ruído, identificação de sistema, dentre outras [8]. Duas medidas se destacam na análise do desempenho dos algoritmos adaptativos: o desajuste e a taxa de convergência [6]. O algoritmo adaptativo o LMS é referência e é de longe o mais conhecido [20]. Novos algoritmos adaptativos têm surgido com um desempenho superior ao LMS em relação ao desajuste e à taxa de convergência, entre esses, tem-se o algoritmo RLS [20]. Outro algoritmo, que tem como base este trabalho é o Sigmoidal (SA), onde [35] demonstrou a sua superioridade em relação ao LMS comparando o desajuste e a taxa de convergência.

2.1 Características dos Filtros Adaptativos

A configuração geral de um ambiente de filtragem adaptativa é ilustrada na Figura 2.1.

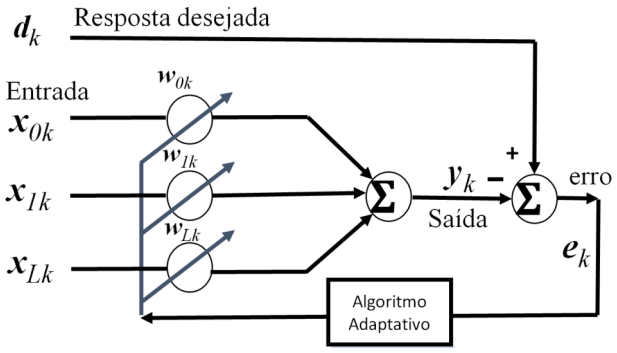


Figura 2.1: Configuração geral de um filtro adaptativo. Em que k representa o número de iteração, $\mathbf{x}_k$ designa o sinal de entrada, y_k é o sinal de saída do filtro adaptativo, e d_k define o sinal desejado. O sinal de erro e_k é dado por $d_k - y_k$. O sinal de erro é então utilizado para formar uma função de desempenho (ou objetivo) que é exigida pelo algoritmo de adaptação para determinar a atualização apropriada dos coeficientes do filtro $\mathbf{w}_k$.

A especificação completa de um sistema adaptativo, mostrado na Figura 2.1, consiste em três itens: Estrutura do filtro adaptativo, aplicação e algoritmo adaptativo [15].

A estrutura de um filtro digital pode ser dada de duas formas: *Finite Impulse Response* (FIR) ou *Infinite Impulse Response* (IIR). Em nosso trabalho consideramos apenas a realização em estrutura FIR [15].

Filtros adaptativos podem ser contínuo ou discretos. Uma forma particular de filtro adaptativo a ser considerada é um tipo discreto (dados amostrados). O filtro discreto consiste de um combinador linear adaptativo (CLA), formado por um sinal de entrada de linhas de atraso e pesos variáveis (ganhos variáveis) cuja entrada de sinais são os sinais na linha de entrada atrasados, uma soma para adicionar os sinais ponderados, e um mecanismo para ajustar automaticamente os pesos. A resposta de impulso de um sistema tal discreto é completamente controlada pelas configurações de peso. O processo de adaptação procura automaticamente uma resposta ao impulso do filtro ideal, ajustando os pesos. A Figura 2.2, ilustra esquematicamente a maioria dos componentes de um filtro adaptativo, que é utilizado neste caso para a modelação de um sistema dinâmico

desconhecido [15].

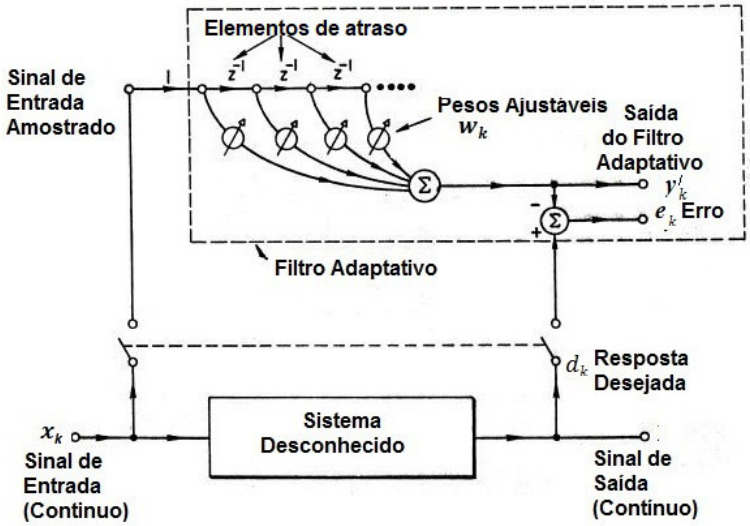


Figura 2.2: Modelo de um filtro adaptativo discreto aplicado ao problema de identificação de sistema desconhecido, onde z^{-I} representa uma unidade de atraso.

O sinal de saída y_k é uma combinação linear dos coeficientes do filtro $\mathbf{w}_k^t \mathbf{x}_k$, que produz um MSE (*Erro Quadrático Médio*) representado por $E[(e_k)^2)]$ como função de uma solução ótima original [15].

Muitos algoritmos adaptativos utilizam-se do MSE como função de custo aplicada sobre o erro, no qual deseja-se minimizar. O MSE é uma função convexa dos componentes do vetor peso e gera uma superfície hiperparabolóide que garante a existência de um mínimo global, ver Figura 2.1, para o caso de um filtro com 2 pesos. O problema é como determinar procedimentos de forma tal a encontrar esse mínimo, o mais rápido possível e com o menor erro final [46].

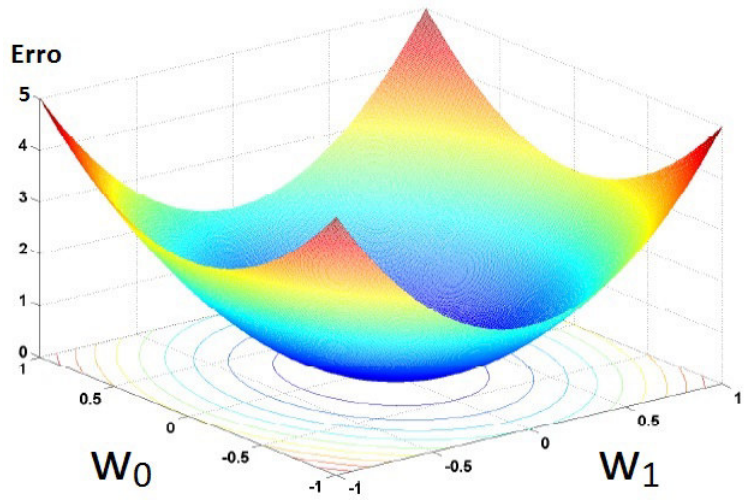


Figura 2.3: Porção da superfície quadrática tridimensional juntamente com alguns contornos. O erro quadrático médio está plotado na vertical, $w0$ e $w1$ variam de -1 a 1.

Dois tipos de processos ocorrem na filtragem adaptativa: treinamento e operação. O processo de treinamento verifica a adaptação das ponderações na linha de atraso aproveitado. O processo de operação consiste em formar os sinais de saída ponderando os sinais da linha de atraso, utilizando os pesos resultantes do processo de adaptação [37].

Existe uma grande variedade de algoritmos adaptativos, a escolha de um ou outro algoritmo é determinada pelos seguintes fatores [37]:

- Taxa de convergência: definida como o número de iterações necessárias para que o algoritmo consiga ajustar os coeficientes do filtro a valores próximos da solução ótima.
- Desajuste: medida da diferença entre os parâmetros obtidos após um certo número de iterações e os parâmetros ótimos.
- Rastreamento: capacidade do algoritmo de acompanhar as variações das propriedades estatísticas de sinais não estacionários.
- Robustez: habilidade do algoritmo para operar satisfatoriamente com um sinal de entrada mal condicionado.
- Complexidade computacional: número de operações aritméticas requeridas para fazer uma iteração completa do algoritmo; tamanho da memória requerida e/ou flexibilidade da programação.

- Estrutura: refere-se à estrutura do fluxo da informação no algoritmo e a maneira de ser implementada em hardware.

- Propriedades numéricas: a implementação do algoritmo adaptativo em um processador digital pode fazer com que o algoritmo se desvie da operação ideal, determinada em precisão infinita, produzindo acumulação de erros de quantização. É possível que os erros se acumulem sem limite. Desta forma o algoritmo pode ser numericamente instável.

2.2 Aplicações

A diferença essencial entre as várias aplicações é a forma de extrair a resposta desejada, assim as aplicações dos filtros adaptativos podem ser divididas em [8] e [15]:

- **Identificação de sistemas**: o filtro adaptativo é usado para encontrar o modelo linear que melhor represente uma planta desconhecida. A planta e o filtro adaptativo são alimentados pelo mesmo sinal de entrada e a saída da planta é a resposta desejada para o filtro adaptativo. A identificação de sistemas adaptativa é frequentemente utilizada em sistemas de comunicações, sistemas de controle e em identificação de estruturas, ver Figura 2.4 [15];

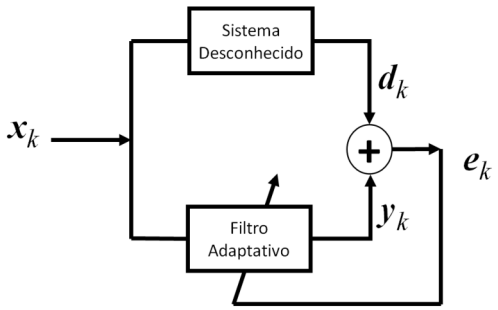


Figura 2.4: Identificação de sistema

- **Modelagem inversa**: o filtro adaptativo é usado para encontrar o modelo inverso de um sistema desconhecido. O sinal desejado para o filtro adaptativo é a entrada atrasada do sistema. Uma aplicação da modelagem inversa é a equalização de canal, em que o objetivo é reduzir a interferência simbólica através de um filtro adaptativo que inverta a resposta do canal, ver Figura 2.5 [15];

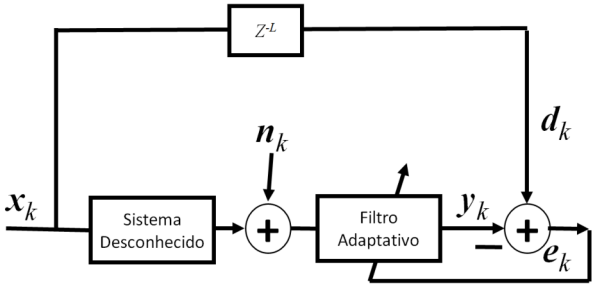


Figura 2.5: Modelagem Inversa

- **Predição Linear**: este filtro adaptativo é utilizado para prover a predição do valor presente de um sinal aleatório. O valor presente serve como sinal desejado no filtro adaptativo e os valores passados alimentam o filtro. Dependendo da aplicação, pode-se ter duas saídas: y_k, denominada saída do filtro preditor ou e_k, denominada erro de predição da saída do filtro. Uma de suas aplicações é na análise espectral, em que utiliza-se o modelamento preditivo para estimar a potência espectral de um sinal de interesse, ver Figura 2.6 [15];

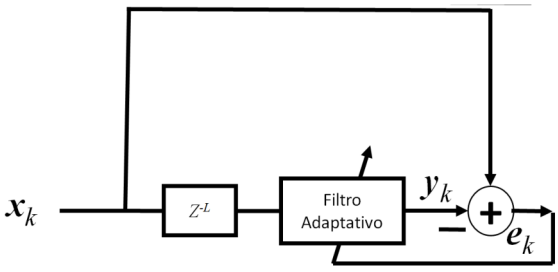


Figura 2.6: Predição Linear

- **Cancelamento de interferência**: nesta aplicação, o filtro adaptativo é utilizado para cancelar uma interferência desconhecida contida em um sinal primário. Este sinal primário serve como sinal desejado para o filtro adaptativo. Um sinal correlacionado com a interferência alimenta a entrada do filtro. Utiliza-se em áreas tão diversas como o cancelamento de interferência de $50Hz$ ou $60Hz$ da rede elétrica em eletrocardiogramas, cancelamento de ruído em sinais de fala, dentre outros, ver Figura 2.7 [15].

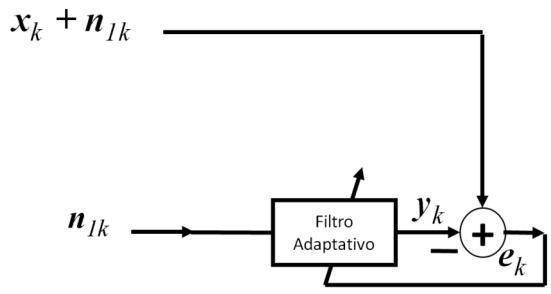


Figura 2.7: Cancelamento de Ruído, onde n_{1k} e n_{2k} são ruídos conhecidos.

3 Análise do desajuste de uma família de algoritmos adaptativos de gradiente estocástico

3.1 Introdução

A filtragem adaptativa tem conseguido uma grande quantidade de sucesso em numerosas aplicações. Muito deste sucesso é devido ao desenvolvimento de conhecidos e robustos algoritmos adaptativos, tais como o algoritmo LMS, que tenta encontrar os parâmetros do filtro adaptativo para minimizar o critério do erro quadrático médio. Característica de desempenho deste algoritmo pode ser aproximadamente prevista dadas informações de covariância da entrada estocástica. Além disso, dado que a entrada é estacionária estocástica, a média convergente de pesos do algoritmo é a solução de Wiener na teoria dos mínimos quadrados de estimativa lineares. Esses algoritmos adaptativos baseados no critério de mínimo erro quadrático médio representam uma referência com características bem conhecidas que podem ser comparados a outros algoritmos [46]. Uma outra família de algoritmos adaptativos de gradiente estocástico, baseados em critérios de erro não quadrático médio, em oposição ao LMS, têm sido sugeridos por vários autores, quer como uma forma de melhorar o desempenho de adaptação destes algoritmos para ambientes estatísticos particulares, como velocidade e desajuste [45], [17].

Douglas e Meng comentando a respeito de filtragem adaptativa baseadas em critérios de erro não quadrático médio, a fim de unificar a discussão desses algoritmos modificados realizaram um série de análise desse algoritmo em 1994 no artigo "Stochastic gradient adaptation under general error criteria. IEEE Transactions on Signal Proc., Vol. 42, nº 6.".

Considere o problema de identificação de sistema, ver Figura 2.2. Em que o objetivo é encontrar uma planta desconhecida com função de transferência polinomial dada por $\mathbf{w}_k^*(z)$ para ser modelado por um filtro adaptativo de tempo discreto $\mathbf{w}_k(z)$. A função de transferência da planta e do filtro adaptativo pode ser representada na forma de resposta ao impulso finita (FIR) dado por [15]:

$$\mathbf{w}(z) = w_1 + w_2 z^{-1} + w_3 z^{-2} + ... + w_N z^{-N+1}, \tag{3.1}$$

onde definimos o vetor $\mathbf{w} = [w_1 w_2 ... w_N]^T$ tal que $\mathbf{w} = \mathbf{w}^*$ para o sistema desconhecido e $\mathbf{w} = \mathbf{w}_k$ para o filtro adaptável a cada iteração k. Desejamos obter estimativas precisas dos coeficientes do modelo da planta $\mathbf{w}^*$ dadas as observações ruidosas da saída da planta d_k, o que pode ser considerado como a resposta desejada do modelo adaptativo, quando acionado pela entrada estocástica $\mathbf{x}_k$ e perturbado com amostras de ruído n_k. Para derivar a família de algoritmos de adaptação de gradiente estocástico com base em critérios de erro não quadrático médio, que em primeiro lugar formar o erro de valor escalar e_k como a diferença entre a resposta desejada e a saída do modelo adaptativo y_k. Nós usamos esse erro para ajustar o vetor de coeficientes de filtro $\mathbf{w}_k$ em cada iteração no algoritmo dada da forma [10]:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mu \hat{\bigtriangledown}_k, \tag{3.2}$$

onde μ é um parâmetro de tamanho do passo e $\hat{\bigtriangledown}_k$ é o gradiente do vetor peso $\mathbf{w}_k$ de uma função de erro (e_k), o chamado critério de erro. Este critério de erro é ainda simétrico e geralmente satisfaz as condições seguintes [10]:

$$0 \leqslant \theta(e_k) = \theta(-e_k) \tag{3.3}$$

$$0 \leqslant e_1 \leqslant e_2 \rightarrow \theta(-e_k) \leqslant \theta(e_2). \tag{3.4}$$

Avaliando o gradiente $\hat{\bigtriangledown}_k$, descobrimos que [10]:

$$\hat{\bigtriangledown}_k = \frac{\partial \theta(e_k)}{\partial \mathbf{w}_k} \tag{3.5}$$

$$= \frac{\theta'(e_k)\partial e_k}{\partial \mathbf{w}_k} \tag{3.6}$$

$$= -\theta'(e_k)\mathbf{x}_k, \tag{3.7}$$

onde o primeiro funcional denota diferenciação. Definindo a função $F(e_k) = \theta(e_k)$, podemos expressar o algoritmo de gradiente estocástico pelo seguinte conjunto de Equações [10]:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mu F'(e_k)\mathbf{x}_k, \tag{3.8}$$

$$e_k = d_k - y_k, \tag{3.9}$$

$$y_k = \mathbf{w}_k^t \mathbf{x}_k. \tag{3.10}$$

Comparando a Equação 3.5 e 7.2 com equações semelhantes definidas para a adaptação do algoritmo LMS [46], percebemos que um critério de erro geral leva a uma modificação do erro no algoritmo LMS pela função ímpar-simétrica não-linear $f(.)$ em estimativa instantâneas do gradiente $\hat{\bigtriangledown}_k$.

Algoritmos adaptativos formulado por [46] foram abordados dentro da comunidade de processamento de sinal com menos frequência, principalmente por causa de várias dificuldades matemáticas associadas à sua análise. Estas dificuldades não descarta a sua utilidade potencial na filtragem adaptativa. No entanto, onde foi indicado que o uso de um critério de erro não quadrático médio pode melhorar o desempenho de um algoritmo adaptativo quando a distribuição do ruído de interferência não é gaussiana [17]. Esta negligência de critérios de erro não quadrático médio ilumina um viés predominante na comunidade de processamento de sinal em direção estatísticas gaussianas, um viés que é motivada mais pela tratabilidade matemática ao invés de experiências práticas. Estatísticas gaussianas são muitas vezes justificados através do Teorema do Limite Central (TLC); No entanto, essas referências à TLC são questionáveis, como o número de sinais de interferência independentes, deve ser muito grande para alcançar gaussianidade aproximada. Além disso, há uma série de comunicações práticas e ambiente de processamento de sinal, tais como canais eletromagnéticos de transmissão, ambientes acústicos submarinos, e canais telefônicos, em que a interferência não-Gaussiana existe, é conhecida e tem sido com precisão modelada. Algoritmos usando critérios de erro não quadráticos médios são úteis para potenciais melhorias de desempenho nessas situações. Para interferências gaussianas, esses algoritmos também levam a uma estimativa imparcial dos coeficientes do filtro sob as restrições dadas na Equação 3.3 [22].

A escolha de um critério de erro dada uma informação a priori sobre a distribuição de ruído foi abordada na análise de filtragem adaptativa como uma procura de parâmetro em que uma estrutura adequada do critério é assumida, e o algoritmo é otimizado para encontrar os parâmetros que permitam atingir o máximo desempenho melhorado em relação ao algoritmo LMS. Tal método de concepção geralmente leva a

um algoritmo sub-ótimo como o critério é limitado a uma classe restrita de funções. No campo de identificação de sistema, os critérios de erro não quadráticos médios foram usados para resolver o problema de outliers de dados, que tendem a causar grandes erros nas estimativas dos parâmetros [45].

Outros autores tem sugerido novos algoritmos de filtragem adaptativa de gradiente estocástico, baseados em critérios do erro não quadrático médio, como forma de simplificar o cálculo do algoritmo em hardware [45] ou para melhorar o desempenho de adaptação destes algoritmos em ambientes estatísticos particulares [36], [45], [10]. Todavia, pouca atenção foi dada a análise de algoritmos de filtragem adaptativa de gradiente estocásticos, baseados em critérios de erro não quadrático médio. Talvez devido à complexidade computacional e dificuldade matemática associada à sua análise, não descartando seu potencial na resolução de problemas práticos de filtragem adaptativas. No entanto, a exploração de suas propriedades levou a importantes descobertas nesta área, por exemplo, o algoritmo LMS limita-se a uma difícil relação entre o erro final de desajuste e o tempo de convergência, um fato que forçou alguns pesquisadores a recorrer a métodos dispendiosos em termos computacionais [5].

Assim, usando funções não lineares do erro, em algoritmos de gradiente estocástico, alguns trabalhos mostraram que eles produzem desempenho geral aprimorado [10] e [4]. Desde então [10] e [4] tem sido referencial teórico para diversos trabalhos na área [38], [18], [3] e [35] e também citado por diversos autores [1], [49],[44] e [31]. Em ambos os casos de funções lineares e não-lineares do erro, os parâmetros de adaptação analisados para o comportamento de convergência foram a constante de tempo e o desajuste. A constante de tempo foi analisada por [36], [10] e [4], os resultados téoricos proposto por [36] superaram [10] e [4] na aproximação da constante de tempo experimental.

O desajuste teórico analisado e proposto por [10], em termos experimentais apresenta resultados aproximados, quando o tamanho de passo de adaptação é muito pequeno e resultados divergentes quando aumentasse o tamanho de passo de adaptação destes algoritmos, tão logo, como forma de enriquecer a discussão nesse assunto, propomos uma metodologia alternativa para o cálculo do desajuste para esses tipos de algoritmos [15].

Uma expressão para o desajuste, especificamente análise de uma família de algoritmos que tenham funções de custo não-lineares de erro não quadrático médio, e que

tenham como gradiente uma função não linear, ímpar, que pode ser expressa em séries de Taylor como uma combinação do erro tem como vantagem uma superfície, quando comparada a quadrática naturalmente produz uma convergência mais rápida com menor desajuste, sendo indicada por alguns autores para utilização na resolução de diversos problemas de processamento de sinais [36], [3], [10], [45], [17].

3.2 Metodologia

Para desenvolvermos um algoritmo usando o método de descida mais íngreme, em vez de utilizar o $\varepsilon = E[e_k^2]$ como função de custo ou critério a ser aplicado sobre o erro e_k. Utilizaremos uma função $F(e_k)$ como função de custo, onde essa função é contínua, não linear, par, simétrica, aplicada sobre o erro a cada iteração, a qual queremos minimizar. Então, a cada k - ésima iteração , nós teremos uma estimação do gradiente $\hat{\Delta}_k = -F'(e_k)\mathbf{x}_k$ [10] e [45], onde $\mathbf{x}_k$ é o sinal de entrada e $F'(.)$ representa a diferenciação de F. Definindo $f(e_k) = F'(e_k)$ podemos especificar um algoritmo adaptativo do tipo descida mais ígreme [10], [45], [36]:

$$\mathbf{w}_{(k+1)} = \mathbf{w}_k + \mu f(e_k)\mathbf{x}_k, \tag{3.11}$$

onde $\mathbf{w}_k$ são os vetores de pesos, μ é o passo de adaptação, a saída do sistema adaptativo é dada por $d_k = \mathbf{w}^{*t}\mathbf{x} + n_k$, sendo $\mathbf{w}^*$ o vetor de peso ótimo e n_k o ruído do sinal de saída, o erro é dado por $e_k = d_k - \mathbf{w}_k^t\mathbf{x}_k$, o vetor de desvio dos pesos é dado por $\mathbf{v}_k = \mathbf{w}_k - \mathbf{w}^*$, logo $e_k = n_k - \mathbf{v}_k^t\mathbf{x}_k$.

Expressando $f(e_k)$ em expansão de série de taylor em torno de $-\mathbf{v}_k^t\mathbf{x}_k$, obtemos:

$$\begin{aligned} f(e_k) &= \sum_{i=0}^{\infty} \frac{f^i}{i!}(-\mathbf{v}_k^t\mathbf{x}_k)n_k^i \\ &= f(-\mathbf{v}_k^t\mathbf{x}_k) + f^{'}(-\mathbf{v}_k^t\mathbf{x}_k)(n_k) \\ &+ \frac{f^{''}(-\mathbf{v}_k^t\mathbf{x}_k)}{2}(n_k^2) + O^3, \end{aligned} \tag{3.12}$$

onde f^i denota a i-ésima derivada da função f e parcela O^3 representa os termos restantes da série de taylor.

Substituindo a Equação 3.12 em 3.11, obtemos:

$$\begin{aligned}\mathbf{v}_{(k+1)} &= \mathbf{v}_k + \mu[f(-\mathbf{v}_k^t\mathbf{x}_k) + f^{'}(-\mathbf{v}_k^t\mathbf{x}_k)(n_k) \\ &+ \frac{f^{''}(-\mathbf{v}_k^t\mathbf{x}_k)}{2}(n_k^2) + O^3]\mathbf{x}_k. \end{aligned} \tag{3.13}$$

Para determinar a equação para o desajuste, utilizamos a Equação 3.13, multiplicamos ambos os lados dela por sua transposta:

Pressupostos 1

- o vetor de entrada $\mathbf{x}_k$ é aleatório limitado em um intervalo $-1 < \mathbf{x}_k \leq 1$, onde cada vetor de entrada $\mathbf{x}_k$ é estatisticamente independente de todos os vetores de entrada anteriores $\mathbf{x}_j$, $j < k$.
- todos os ruído são estatisticamente independente de $\mathbf{x}_k$.
- todas as variáveis tenha distribuição de probabilidade não necessariamente Gaussiana.
- o vetor de peso $\mathbf{w}_k$ é estatisticamente independente de $\mathbf{x}_k$

Continuamos assumindo que a norma do vetor de desvio dos pesos são pequenos, no cálculo da covariância, assim, podemos considerar apenas os dois primeiros termos da Equação 3.13. Para o cálculo da covariância multiplicamos 3.13 pela sua transposta.

$$\begin{aligned}\mathbf{v}_k\mathbf{v}_k^t &= [\mathbf{v}_k + \mu f(-\mathbf{v}_k^t\mathbf{x}_k)\mathbf{x}_k + \mu f^{'}(-\mathbf{v}_k^t\mathbf{x}_k)(n_k)\mathbf{x}_k] \\ &[\mathbf{v}_k^t \quad +\mu f(-\mathbf{v}_k^t\mathbf{x}_k)\mathbf{x^t}_k + \mu f^{'}(-\mathbf{v}_k^t\mathbf{x}_k)(n_k)\mathbf{x^t}_k] \\ &= \mathbf{v}_k\mathbf{v}_k^t + \mu f(-\mathbf{v}_k^t\mathbf{x}_k)\mathbf{v}_k\mathbf{x^t}_k \\ &+ \mu f^{'}(-\mathbf{v}_k^t\mathbf{x}_k)(n_k)\mathbf{v}_k\mathbf{x}_k + \mu f(-\mathbf{v}_k^t\mathbf{x}_k)\mathbf{v}_k\mathbf{x^t}_k \\ &+ \mu^2 f^2(-\mathbf{v}_k^t\mathbf{x}_k)\mathbf{x}_k\mathbf{x^t}_k \\ &+ \mu^2 f(-\mathbf{v}_k^t\mathbf{x}_k)f^{'}(-\mathbf{v}_k^t\mathbf{x}_k)n_k\mathbf{x}_k\mathbf{x}_k^t \\ &+ \mu f^{'}(-\mathbf{v}_k^t\mathbf{x}_k)(n_k)\mathbf{v}_k\mathbf{x^t}_k \\ &+ \mu^2 f(-\mathbf{v}_k^t\mathbf{x}_k)f^{'}(-\mathbf{v}_k^t\mathbf{x}_k)n_k\mathbf{x}_k\mathbf{x}_k^t \\ &+ \mu^2 f^{'2}(-\mathbf{v}_k^t\mathbf{x}_k)n_k^2\mathbf{x}_k\mathbf{x}_k^t. \end{aligned} \tag{3.14}$$

Em seguida tomamos esperança da Equação 3.14 e encontramos a sua covariância e aplicamos os Pressupostos 1, ficaremos com a Equação 3.15:

$$\begin{aligned} E \quad & [\mathbf{v}_{(k+1)}\mathbf{v}^{\mathbf{t}}{}_{(k+1)}] = E[\mathbf{v}_k\mathbf{v}_k^t] + 2\mu E[f(-\mathbf{v}_k^t\mathbf{x}_k)]E[\mathbf{v}_k\mathbf{x}_k^t] \\ + \quad & 2\mu E[f^{'}(-\mathbf{v}_k^t\mathbf{x}_k)]n_k E[\mathbf{v}_k\mathbf{x}_k^t] \\ + \quad & 2\mu^2 E[f(-\mathbf{v}_k^t\mathbf{x}_k)f^{'}(-\mathbf{v}_k^t\mathbf{x}_k)]n_k E[\mathbf{x}_k\mathbf{x}_k^t] \\ + \quad & \mu^2 E[f^2(-\mathbf{v}_k^t\mathbf{x}_k) + f^{'2}(-\mathbf{v}_k^t\mathbf{x}_k)n_k^2]E[\mathbf{x}_k\mathbf{x}_k^t]. \end{aligned} \tag{3.15}$$

Definindo $E[\mathbf{x}_k\mathbf{x}_k^t] = \mathbf{R}$, e $E[n_k^2] = \sigma_n^2$, onde usamos o fato de que a função densidade de probabilidade (pdf) do ruído é simétrica e os momentos ímpares do ruído são iguais a zero. Adotando que $E[\mathbf{v}_{(k+1)}\mathbf{v}^{\mathbf{t}}{}_{(k+1)}] = E[\mathbf{v}_k\mathbf{v}_k^t]$, logo obteremos:

$$\begin{aligned} E[\mathbf{v}_k\mathbf{v}_k^t] \quad &= \quad E[\mathbf{v}_k\mathbf{v}_k^t] + 2\mu E[f(-\mathbf{v}_k^t\mathbf{x}_k)]E[\mathbf{v}_k\mathbf{x}_k^t] \\ &+ \quad \mu^2 E[f^{'2}(-\mathbf{v}_k^t\mathbf{x}_k)\sigma_n^2]\mathbf{R}, \end{aligned} \tag{3.16}$$

pois $f^2(-\mathbf{v}_k^t\mathbf{x}_k) \to 0$ com $\mathbf{v} \to 0$. Utilizando o Teorema apresentado por [26], seja f uma função não linear, ímpar, definida e contínua em um intervalo $(-\delta, \delta)$, onde δ é um número positivo suficiente pequeno [36]. Desta forma, $f(\alpha\beta) \approx \alpha f(\beta)$, para todo α, $\beta \in (-\delta, \delta)$, podemos reescrever como:

$$\begin{aligned} E[\mathbf{v}_k\mathbf{v}_k^t] \quad &= \quad E[\mathbf{v}_k\mathbf{v}_k^t] + 2\mu E[f(-\mathbf{x}_k^t\mathbf{x}_k)]E[\mathbf{v}_k\mathbf{v}_k^t] \\ &+ \quad \mu^2 E[f^{'2}(-\mathbf{v}_k^t\mathbf{R}\mathbf{x}_k)]I\sigma_n^2. \end{aligned} \tag{3.17}$$

É mostrado em [10] que, se $\mathbf{x}_k$ for assumido conjuntamente gaussiano, então o excesso de erro condicionado no vetor de erro do peso também é gaussiano com variância dada por $\sigma_{ex,k}^2 = \mathbf{v}_k^t\mathbf{R}\mathbf{x}_k$.

$$\begin{aligned} E[\mathbf{v}_k\mathbf{v}_k^t] \quad &= \quad E[\mathbf{v}_k\mathbf{v}_k^t] - 2\mu E[f(\mathbf{x}_k^t\mathbf{x}_k)]E[\mathbf{v}_k\mathbf{v}_k^t] \\ &+ \quad \mu^2 E[f^{'2}(\sigma_{ex,k}^2)]I\sigma_n^2. \end{aligned} \tag{3.18}$$

A solução resultante em estado estacionário, será dada por:

$$E[\mathbf{v}_k\mathbf{v}_k^t] = (\frac{\mu^2 E[f'^2(\sigma_{ex,k}^2)]\sigma_n^2}{2\mu E[f(\mathbf{x}_k^t\mathbf{x}_k)]})I. \tag{3.19}$$

Sabe-se que o excesso de erro médio quadrático na saída do filtro é causado por flutuações aleatórias (i) nos coeficientes de peso durante a adaptação e no sinal de entrada. Como assumimos a independência de $\mathbf{x}_k$ e $\mathbf{v}_k$, bem como a natureza não correlacionada do vetor de erro de peso, onde ficamos com:

$$E[(\mathbf{v}_k^t\mathbf{x}_k)^2] = E[(\sum_{i=1}^{N} v_{i,k}x_{i,k})^2]. \tag{3.20}$$

$$= \sum_{i=1}^{N} E[v_{i,k}^2]E[x_{i,k}^2] \tag{3.21}$$

$$= \frac{\mu^2 E[f'^2(\sigma_{ex,k}^2)]\sigma_n^2}{2\mu E[f(\mathbf{x}_k^t\mathbf{x}_k)]}E[x_{i,k}^2]. \tag{3.22}$$

O desajuste é definido como o excesso de erro quadrático médio sobre o erro mínimo médio quadrático, dado por:

$$M = \frac{E[(\mathbf{v}_k^t\mathbf{x}_k)^2]}{E[n_k^2]} \tag{3.23}$$

$$M = \frac{\mu E[f'^2(\sigma_{ex,k}^2)]}{2E[f(\mathbf{x}_k^t\mathbf{x}_k)]}tr[\mathbf{R}], \tag{3.24}$$

sendo $E[x_{i,k}^2] = tr[\mathbf{R}]$ e $tr[.]$ denota a operação traço.

4 Processamento de Sinal Digital e os Efeitos da Precisão Finita

O mundo real consiste numa infinidade de sinais analógicos que são representados em precisão infinita ou analógica. Os sinais analógicos que não são ou não podem ser entendidos em linguagem de máquina, são geralmente observados na natureza e nos seres humanos na forma analógica. Para que nossos computadores, celulares e outros dispositivos de hardware possam trabalhar, é necessário que haja um processamento desses sinais analógicos na forma digital que são representados em precisão finita. Alguns dispositivos de hardware podem interagir diretamente com programas, a fim de agilizar e dinamizar a execução. Dependendo do tipo e modelo do dispositivo de hardware pode-se encontrar placas de circuitos eletrônicos específicas para processamento de áudio, vídeo e imagem. Para tal são necessários dispositivos especiais de codificação e decodificação dos sinais, que realiza as conversões necessárias para que ocorra o processamento [15].

O sinal digital é justamente uma representação numérica dos sinais analógicos. Para que haja essa representação, são utilizados os conversores A/D (*Conversores Analógicos Digitais*), que processam os sinais analógicos e os transformam em sinais digitais, dado numa sequência de 0s e 1s, podendo assim ser analisados por computadores e programas [15].

A representação dos números em dispositivos de hardware digital, números (de valor real ou valores complexos, inteiros ou frações) são representados usando dígitos binários (bits), que levam o valor de 0 ou 1. Para realizar as operações arimética e lógicas necessárias para o processamento desses números, pode ser implementada duas abordagens diferentes, dependendo da facilidade de implementação da precisão, bem como da faixa dinâmica necessárias no processamento. A aritmética de ponto fixo que pode ser fácil de implementar, mas tem apenas uma faixa dinâmica e precisão fixa. A aritmética de ponto flutuante, por outro lado, tem uma vasta gama de faixa dinâmica e uma precisão variável (em relação à amplitude de um número), sendo portanto mais complicado de implementar e analisar [41] e [15].

Uma vez que um computador pode operar apenas em um binário (por exemplo, 0 ou 1), números positivos podem diretamente ser representados utilizando números binários. O problema é quanto ao modo de representar os números negativos. Os três formatos diferentes usados em cada uma dessas representações de sinal são: formato de 'sinal magnitude', formato 'complemento de um' e formato 'complemento de dois' [15].

4.1 Erros de Quantização no processo de conversão para o sinal digital

Quantização é o processo de atribuição de valores discretos para um sinal cuja amplitude varia entre infinitos valores entre dois valores finitos, ver Figura 4.1 [30].

Quando aproximamos um sinal original por um sinal digital aproximado, podem ocorrer os erros de quantização, que é a diferença entre o sinal original e o sinal digital aproximado. Os sinais analógicos sempre terão esse erro de quantização nessa aproximação, que diminui ou aumenta de acordo com a resolução de bits, quanto maior número de bits menor será o erro. Os sinais digitais originais podem ter ou não o erro de quantização, o erro de quantização será nulo quando a resolução do sinal digital aproximado for igual ou maior que a resolução do sinal digital original [30]..

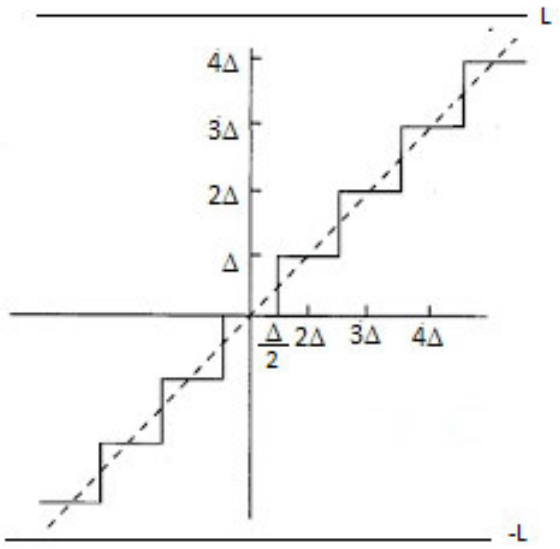


Figura 4.1: Representação esquemática de níveis de quantização para o caso de $b = 2$ bits e $\triangle = L/4$. Consideramos um valor L limiar. Usando b bits, o intervalo de $[0, L]$ pode ser dividido em 2^b níveis de largura $\triangle = L/2^b$ cada. Se, além disso, usamos 1 (um) bit de sinal, os resultantes $(b+1)$ bits divide o intervalo $[-L, L]$ em 2^{2b} níveis. Aqui assumimos operação de arredondamento.

Dessa forma podemos quantificar o erro de aproximação de seus valores originais, devido aos efeitos da quantização, dada por [15]:

$$n_\theta = d_k - d_{Qk}, \tag{4.1}$$

onde n_θ indica o erro ou ruído devido a quantização e θ representa a quantidade ou variável quantizada.

Utilizando aritmética de ponto fixo, com a operação de arredondamento assumido para a quantização, modelo mostrado na Figura 4.1, o erro pode ser modelado em processo estocástico com média zero e variância dada por [15]:

$$\sigma^2 = C\frac{2^{-2b}}{12}, \tag{4.2}$$

onde σ^2 é a variância, C é uma constante e b é o número de bits utilizado. Para filtragem adaptativa dada em precisão finita afim de facilitar os cálculos, supõe-se que os sinais internos são adequadamente dimensionados, de modo que nenhum excesso ocorre durante os cálculos, e que os valores dos sinais devam situar-se entre $[-1, 1]$ [15].

Na Figura 4.2, podemos observar o comportamento da variância dada pela Equação 4.2, para o número de bits variando entre 7 bits e 32 bits [15].

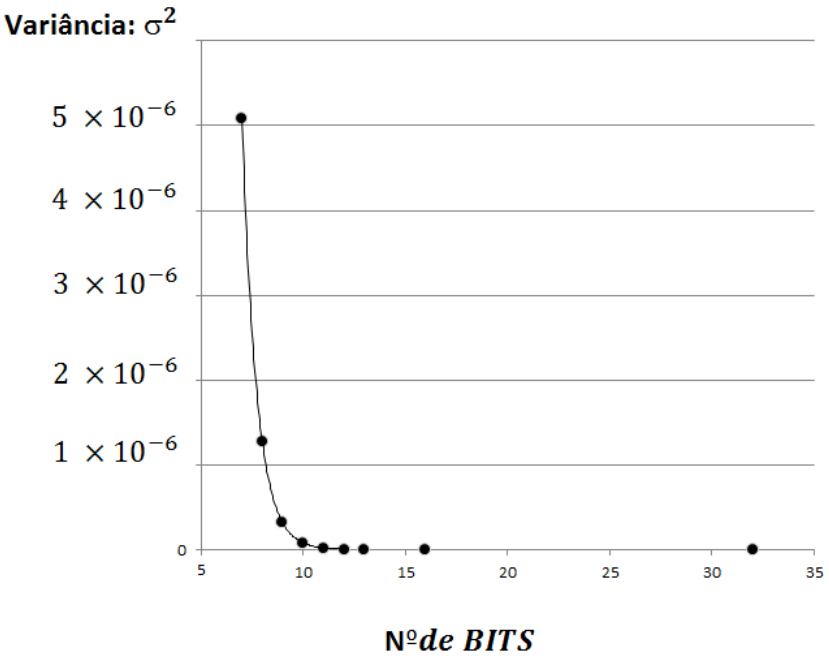


Figura 4.2: Variância devido à quantização com aritmética de ponto fixo. Para $C = 1$ número de *bits* variando entre 7 bits à 32 bits

4.2 Filtragem Adaptativa em Precisão Finita

Os efeitos numérico da precisão finita em algoritmos adaptativos, onde até agora nós consideramos modelos de filtros e implementações em que ambos os coeficientes do filtro e as operações do filtro, como adições e multiplicações foram expressos usando números em precisão infinita. Quando os sistemas de tempo discreto são implementados em hardware ou software, todos os parâmetros e operações aritméticas são implementados usando números de precisão finita e, portanto, seus efeitos são inevitáveis. Considere-se um filtro típico implementado como uma estrutura dada em precisão finita. Quando a representação em precisão finita é utilizado na sua aplicação, existem três possíveis considerações que afetam o desempenho global da sua implementação. Consideramos como as principais [30]:

1. A quantização dos coeficientes filtro, w_k, para obter sua representação de

comprimento finito, w_{Qk},

2. A quantização das variáveis e quantidades e da sequência de entrada, x_k para se obter x_{Qk}, e

3. Toda aritmética interna deve ser convertido em suas representações em precisão finita.

Assim, a saída, y_k, é também um valor quantizado y_{Qk}. Isso nos da uma nova percepção do filtro. Espera-se dessa forma uma nova análise do filtro dado os novos sinais e coeficientes encontrados.

Uma vez que a operação de quantização é uma operação não-linear, a análise global que leva em consideração todos os três efeitos descritos acima é de alta complexidade [37]. Portanto, faz-se necessário estudar cada um desses efeitos separadamente como se fosse o único agindo no momento. Isto torna mais fácil a análise e os resultados mais interpretáveis. Podemos definir os erros ou ruídos de quantização nas variáveis e quantidades relacionadas com a filtragem adaptativa de forma geral como segue:

$$n_d = d_k - d_{Qk}, \tag{4.3}$$

$$n_y = y_k - y_{Qk}, \tag{4.4}$$

$$n_e = e_k - e_{Qk}, \tag{4.5}$$

$$\mathbf{n_w} = \mathbf{w}_k - \mathbf{w}_{Qk}, \tag{4.6}$$

onde n_d é o ruído de quantização do sinal desejado, n_y o ruído de quantização do sinal de saída, n_e o ruído de quantização do sinal de erro e n_w o ruído devido a quantização, do coeficientes ou vetor peso. Supondo que o sinal de entrada $\mathbf{x}_k$ não sofre quantização. Os efeitos de quantização nos sinais de entrada e desejado pode ser facilmente tomado em consideração separadamente, a partir de outras fontes de erro de quantização. No caso do sinal desejado, o erro de quantização pode ser adicionado ao ruído de medição, enquanto para o sinal de entrada, o efeito de base na saída do filtro é um ruído adicional [8].

4.3 Implementação em Dispositivo de Hardware Digital

Muitos projetos em filtragem adaptativa são implementados em DSP ou FPGA, que apresentam-se muitas vezes em precisão finita. O uso desses dispositivos de hardware digital hoje em dia é muito grande. Vários itens como: celulares, equipamentos de laboratórios, automóveis e computadores fazem parte destes dispositivos de hardwares. Estas implementações tem o potencial de gerar efeitos indesejados na resposta do filtro, além de instabilidade numérica. Como resultado, surgem dúvidas, na fase de implementação quanto à eficácia do filtro digital e a quantidades de bits necessária para a sua representação de modo que os parâmetros de projetos ainda sejam satisfeitos [15].

Estes por serem dispositivos programáveis necessitam de uma plataforma para desenvolvimento de projetos. Estes dispositivos trabalham com linguagem de programação, em sua grande maioria com as principais linguagens, como Assembly, C, VHDL e outras. Cada fabricante fornece junto ao produto uma plataforma, que foi feita para trabalhar com as funções pré-definidas de cada dispositivos de hardware. Estas plataformas fornecem um ambiente de trabalho necessário para a realização de um projeto. Estes software são na verdade compiladores que trabalham ligados aos dispositivos, gerenciando toda a execução, simulação e depuração do código [41].

Para a realização de um projeto utilizando-se um dispositivo de hardware digital podemos destacar três etapas gerais para execução do projeto: pesquisa, simulação e emulação [30].

- 1º Etapa: A pesquisa envolve basicamente o entendimento do problema, onde serão buscadas informações se o dispositivo a ser utilizado envolve necessidade de cálculos em ponto fixo ou em ponto flutuante [15].
- 2º Etapa: A etapa de simulação ocorrerá na modelagem inicial do projeto, onde é realizada, através dos ambientes de desenvolvimento de cada dispositivo[15].
- 3º Etapa: Por fim, temos a etapa de emulação, que envolve a realização de testes do projeto já dentro de seu ambiente de uso. Nesta etapa o dispositivo de hardware é submetido as análises detalhadas, visando-se obter um equipamento o mais preciso possível, que não apresente falhas [15].

5 Análise em Precisão Finita de uma Família de Algoritmos Adaptativos Baseado no Erro não quadrático médio

5.1 Análise de Adaptação

Considere o diagrama em blocos mostrado na Figura 5.1, referente ao problema de identificação de sistemas com filtragem adaptativa em um hardware [15].

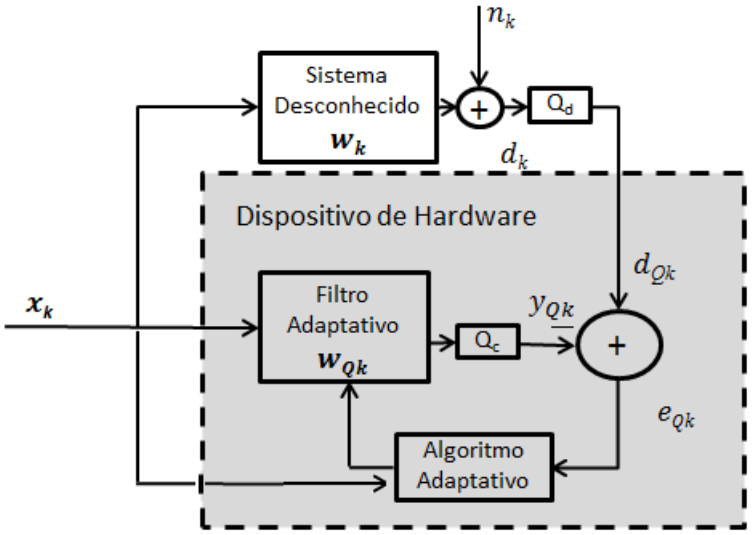


Figura 5.1: Identificação de sistema implementado em precisão finita. Onde $\mathbf{x}_k$ designa o sinal de entrada. Os blocos Q_d e Q_c são quantizadores de b bits. Assim, o quantizador, Q_d, usa b_d bits para os dados de entrada e Q_c quantiza os coeficientes em b_c bits. O sinal de saída y_{Qk} indica a forma quantizada. O sinal desejado definido por d_k do sistema desconhecido e d_{Qk} sua forma quantizada. O ruído de medição n_k e o sinal de erro quantizado e_{Qk}. Os coeficientes $\mathbf{w_k}$ do sistema desconhecido e $\mathbf{w_{Qk}}$ do filtro adaptativo.

5.1.1 Pressupostos 2

Nesta seção assumimos os seguintes pressupostos [16] e [15]:

- assume-se que $\mathbf{x}_k$ é um sinal aleatório discreto com $-1 < \mathbf{x}_k \leq 1$, não sofre quantização e cada $\mathbf{x}_k$ vetor de dados de entrada é estatisticamente independente de todos os vetores de dados anteriores $\mathbf{x}_k$ para $j < k$ [15].
- a sequência de ruído amostrado (por medição ou erros de quantização gerado em qualquer etapa no algoritmo) é considerado uma variável aleatória com média zero, que é estatisticamente independente de quaisquer outros erros e/ou quantidades relacionadas.
- o parâmetro de tamanho do passo μ deve ser escolhido suficientemente pequeno de tal modo que o excesso de erro médio quadrático na iteração k seja muito menor do que o mínimo erro médio quadrático perto de convergência.
- as variações dos erros dependem do tipo da quantização e aritmética que vai ser utilizada na implementação do algoritmo [8].

Expressando a saída do sistema desconhecido como [14]:

$$d_k = \mathbf{w}^{*t}\mathbf{x} + n_k, \tag{5.1}$$

onde $\mathbf{w}^*$ é o coeficiente ótimo. Supõe-se que o sinal de entrada e o sinal desejado não sofre quantização, de modo que apenas a quantização interna de computação são tidas em conta. Assim, podemos expressar o erro como [14]:

$$e_k = (\mathbf{w}^{*t}\mathbf{x}_k - n_k) - \mathbf{w}_k^t\mathbf{x}_k. \tag{5.2}$$

Em sua forma quantizada como [14]:

$$\begin{aligned} e_{Qk} &= ((\mathbf{w}^{*t}\mathbf{x}_k - n_k) - \mathbf{w}_k^t\mathbf{x}_k)_Q \\ &= n_k + n_e - \mathbf{v}_{Qk}^t\mathbf{x}_k, \end{aligned} \tag{5.3}$$

onde definimos n_e como o ruído devido a quantização e $\mathbf{v}_{Qk}^t$ é o vetor desvio do peso dado por [14]:

$$\mathbf{v}_{Qk} = \mathbf{w}_{Qk} - \mathbf{w}^*. \tag{5.4}$$

Subtraindo $\mathbf{w}^*$ de ambos os lados da Equação 3.12, o algoritmo de adaptação de erro torna-se [14]:

$$(\mathbf{v}_{k+1})_Q = (\mathbf{v}_k + \mu f(n_k + n_e - \mathbf{v}_k^t \mathbf{x}_k)\mathbf{x}_k)_Q. \tag{5.5}$$

Que agora é uma equação não linear iterativo no vetor de desvio $\mathbf{v}_k$.

5.2 Análise da constante de tempo

Expressando a não linearidade $f(e_k)$ em uma expansão em série de Taylor sobre termo $-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk}$ obtemos:

$$\begin{aligned} f(n_k - \mathbf{v}_{Qk}^t \mathbf{x}_{Qk}) &= \sum_{i=0}^{\infty} \frac{f^i}{i!}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})(n_k - n_e)^i \\ &= f(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk}) + f^{'}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})(n_k - n_e) \\ &+ \frac{f^{''}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})}{2}(n_k - n_e)^2 \\ &+ \frac{f^{'''}(\delta)}{6}(n_k - n_e)^3, \end{aligned} \tag{5.6}$$

onde $f^i(.)$ denota a derivada (i) da função $f(.)$. Temos utilizado a forma derivada para representar o termo restante de Taylor tal que δ encontra-se no intervalo $[0, \mathbf{v}_k^t \mathbf{x}_k]$. Tecnicamente esta representação de série de Taylor limita a não linearidade $f(.)$ para uma classse restrita de funções que são diferenciáveis até a terceira ordem ao longo de toda extensão do erro e_k.

Agora vamos examinar o comportamento da Equação 5.6 no caso em que o vetor de desvio do peso é pequeno e o termo de O^3 pode ser desprezado[16]:

$$\mathbf{v}_{Q(k+1)} = \mathbf{v}_{Qk} + \mu[f(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})\mathbf{x}_{Qk} + f^{'}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})(n_k + n_e)\mathbf{x}_{Qk} + \frac{f^{''}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})}{2}(n_k + n_e)^2 \mathbf{x}_{Qk}] + \mathbf{n}_v, \tag{5.7}$$

onde n_v é um vetor de ruído devido à quantização do vetor peso $\mathbf{v}_k$.

Tomando a média de ambos os lados, obtemos [16]:

$$\begin{aligned} E[\mathbf{v}_{Q(k+1)}] &= E[\mathbf{v}_{Qk}] + \mu E[f(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})\mathbf{x}_{Qk}] \\ &+ \mu E[f^{'}(-\mathbf{v}_Q k^t \mathbf{x}_{Qk})\mathbf{x}_{Qk} n_k] \\ &+ \mu E[f^{'}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})\mathbf{x}_{Qk} n_e] \\ &+ \mu E[\frac{f^{''}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})}{2}\mathbf{x}_{Qk} n_k^2] \\ &+ \mu E[\frac{f^{''}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})}{2}\mathbf{x}_{Qk} 2 n_k n_e] \\ &+ \mu E[\frac{f^{''}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})}{2}\mathbf{x}_{Qk} n_e^2] - E[\mathbf{n}_v] \end{aligned} \tag{5.8}$$

E utilizando os pressupostos 2 apresentados ficamos:

$$\begin{aligned} E[\mathbf{v}_{Q(k+1)}] &= E[\mathbf{v}_{Qk}] + \mu E[f(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})\mathbf{x}_{Qk}] \\ &+ \mu E[\frac{f^{''}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})}{2}\mathbf{x}_{Qk}] E[n_k^2] \\ &+ \mu E[\frac{f^{''}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})}{2}\mathbf{x}_{Qk}] E[n_e^2] \end{aligned} \tag{5.9}$$

Sabendo que $E[n_k^2] = \sigma_n^2$ e $E[n_e^2] = \sigma_e^2$ [14],

$$\begin{aligned} E[\mathbf{v}_{Q(k+1)}] &= E[\mathbf{v}_{Qk}] + \mu E[f(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})\mathbf{x}_{Qk}] \\ &+ \mu E[\frac{f^{''}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})}{2}\mathbf{x}_{Qk}] \sigma_n^2 \\ &+ \mu E[\frac{f^{''}(-\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})}{2}\mathbf{x}_{Qk}] \sigma_e^2 \end{aligned} \tag{5.10}$$

Considerando o Teorema apresentado por [36] podemos reescrever como:

$$\begin{aligned} E[\mathbf{v}_{Q(k+1)}] &= E[\mathbf{v}_{Qk}] - \mu E[f(\mathbf{x}_{Qk}\mathbf{x}_{Qk}^t)\mathbf{v}_{Qk}] \\ &- \mu E[\frac{f^{''}(\mathbf{x}_{Qk}\mathbf{x}_{Qk}^t)}{2}\mathbf{v}_{Qk}] \sigma_n^2 \\ &- \mu E[\frac{f^{''}(\mathbf{x}_{Qk}\mathbf{x}_{Qk}^t)}{2}\mathbf{v}_{Qk}] \sigma_e^2 \end{aligned} \tag{5.11}$$

Aqui consideramos $\mathbf{R} = E[\mathbf{x}_k \mathbf{x}_k^t]$, como matriz de autocorrelação do sinal de entrada.

$$\begin{aligned} E[\mathbf{v}_{Q(k+1)}] &= E[\mathbf{v}_{Qk}] - \mu f(\mathbf{R}) E[\mathbf{v}_{Qk}] \\ &- \mu \frac{f^{''}(\mathbf{R})}{2} E[\mathbf{v}_{Qk}] \sigma_n^2 \\ &- \mu \frac{f^{''}(\mathbf{R})}{2} E[\mathbf{v}_{Qk}] \sigma_e^2 \end{aligned} \tag{5.12}$$

$$= \left(\mathbf{I} - \mu [f(\mathbf{R}) - \frac{f^{''}(\mathbf{R})}{2} \sigma_k^2 - \frac{f^{''}(\mathbf{R})}{2} \sigma_e^2] \right) E[\mathbf{v}_k] \tag{5.13}$$

Esta equação pode ser utilizada para determinar um limite no parâmetro μ para garantir convergência. Definindo $\mathbf{Q}$ como sendo a matriz de autovetores de $\mathbf{R}$, ou seja, as colunas de $\mathbf{Q}$ são formadas pelos autovetores de $\mathbf{R}$, e $\mathbf{\Lambda}$ uma matriz diagonal, cujos elementos da diagonal principal são os autovalores de $\mathbf{R}$, escrevemos a matriz de correlação de entrada como:

$$\mathbf{R} = \mathbf{Q} \Lambda \mathbf{Q}^{-1}. \tag{5.14}$$

Definindo, também $\tilde{\mathbf{v}}_{Qk} = \mathbf{Q}^{-1} \mathbf{v}_{Qk}$ como uma rotação nos vetores pesos, reescrevemos Equação 5.13 da seguinte maneira:

$$\begin{aligned} \mathbf{Q} E[\tilde{\mathbf{v}}_{Q(k+1)}] &= \left(\mathbf{I} - \mu [f(\mathbf{R}) - \frac{f^{''}(\mathbf{R})}{2} \sigma_n^2 - \frac{f^{''}(\mathbf{R})}{2} \sigma_e^2] \right) \mathbf{Q} E[\tilde{\mathbf{v}}_{Qk}] \\ &= \mathbf{Q}^{-1} \left(\mathbf{I} - \mu [f(\mathbf{R}) - \frac{f^{''}(\mathbf{R})}{2} \sigma_n^2 - \frac{f^{''}(\mathbf{R})}{2} \sigma_e^2] \right) \mathbf{Q} E[\tilde{\mathbf{v}}_{Qk}] \\ &= \left(\mathbf{Q}^{-1}\mathbf{Q} - \mu [\mathbf{Q}^{-1} f(\mathbf{R}) \mathbf{Q} - \mathbf{Q}^{-1} \frac{f^{''}(\mathbf{R})}{2} \mathbf{Q} \sigma_n^2 - \mathbf{Q}^{-1} \frac{f^{''}(\mathbf{R})}{2} \mathbf{Q} \sigma_e^2] \right) E[\tilde{\mathbf{v}}_{Qk}] \\ &= \left(\mathbf{I} - \mu [f(\mathbf{\Lambda}) - \frac{f^{''}(\mathbf{\Lambda})}{2} \sigma_n^2 - \frac{f^{''}(\mathbf{\Lambda})}{2} \sigma_e^2] \right) E[\tilde{\mathbf{v}}_{Qk}]. \end{aligned} \tag{5.15}$$

O que temos é o valor esperado:

$$\tilde{\mathbf{v}}_{Q(k+1)} = \left(\mathbf{I} - \mu [f(\mathbf{\Lambda}) - \frac{f^{''}(\mathbf{\Lambda})}{2} \sigma_n^2 - \frac{f^{''}(\mathbf{\Lambda})}{2} \sigma_e^2] \right) \tilde{\mathbf{v}}_{Qk}, \tag{5.16}$$

a qual pode ser resolvida por indução da seguinte forma: iniciando com um peso inicial arbitrário, $\tilde{\mathbf{v}}_0$, temos, para as primeiras três iterações:

$$\begin{aligned}
\tilde{\mathbf{v}}_1 &= \left(\mathbf{I} - \mu[f(\mathbf{\Lambda}) - \frac{f''(\mathbf{\Lambda})}{2}\sigma_n^2 - \frac{f''(\mathbf{\Lambda})}{2}\sigma_e^2]\right)\tilde{\mathbf{v}}_0 \\
\tilde{\mathbf{v}}_2 &= \left(\mathbf{I} - \mu[f(\mathbf{\Lambda}) - \frac{f''(\mathbf{\Lambda})}{2}\sigma_n^2 - \frac{f''(\mathbf{\Lambda})}{2}\sigma_e^2]\right)\tilde{\mathbf{v}}_0 \\
\tilde{\mathbf{v}}_3 &= \left(\mathbf{I} - \mu[f(\mathbf{\Lambda}) - \frac{f''(\mathbf{\Lambda})}{2}\sigma_n^2 - \frac{f''(\mathbf{\Lambda})}{2}\sigma_e^2]\right)\tilde{\mathbf{v}}_0
\end{aligned} \tag{5.17}$$

Generalizando para k-ésima iteração, obtemos:

$$\tilde{\mathbf{v}}_k = \left(\mathbf{I} - \mu[f(\mathbf{\Lambda}) - \frac{f''(\mathbf{\Lambda})}{2}\sigma_n^2 - \frac{f''(\mathbf{\Lambda})}{2}\sigma_e^2]\right)^k \tilde{\mathbf{v}}_0 \tag{5.18}$$

Da Equação 5.18, vemos que o algoritmo será convergente se

$$\lim_{k\to\infty}\left(\mathbf{I} - \mu[f(\mathbf{\Lambda}) - \frac{f''(\mathbf{\Lambda})}{2}\sigma_n^2 - \frac{f''(\mathbf{\Lambda})}{2}\sigma_e^2]\right)^k = 0 \tag{5.19}$$

O termo entre parenteses na Equação 5.19 é uma matriz diagonal, cujos elementos da diagonal principal são da forma:

$$\lim_{k\to\infty}\left(1 - \mu[f(\lambda_\mathbf{i}) - \frac{f''(\lambda_\mathbf{i})}{2}\sigma_n^2 - \frac{f''(\lambda_\mathbf{i})}{2}\sigma_e^2]\right)^k, \tag{5.20}$$

com i=0, ..., L.

A condição de convergência será satisfeita com

$$\left|1 - \mu[f(\lambda_\mathbf{i}) - \frac{f''(\lambda_\mathbf{i})}{2}\sigma_n^2 - \frac{f''(\lambda_\mathbf{i})}{2}\sigma_e^2]\right| \leq 1, \tag{5.21}$$

que obtemos se tomarmos:

$$0 < \mu < \frac{2}{f(\lambda_\mathbf{max}) + \frac{f''(\lambda_\mathbf{max})}{2}\sigma_n^2 + \frac{f''(\lambda_\mathbf{max})}{2}\sigma_e^2}, \tag{5.22}$$

onde $\lambda_\mathbf{max}$ é o maior autovalor da matriz $\mathbf{R}$.

Seja:

$$\tilde{v}_i = (1 - \mu[f(\lambda_\mathbf{i}) - \frac{f''(\lambda_\mathbf{i})}{2}\sigma_n^2 - \frac{f''(\lambda_\mathbf{i})}{2}\sigma_e^2])^k \tilde{v}_0. \tag{5.23}$$

Definamos:

$$r_i \cong \left|1 + \mu[f(\lambda_\mathbf{i}) + \frac{f''(\lambda_\mathbf{i})}{2}\sigma_n^2 + \frac{f''(\lambda_\mathbf{i})}{2}\sigma_e^2]\right|. \tag{5.24}$$

Onde r_i é a razão geométrica da sequência de amostras de $\tilde{v}_i$. Considerando uma unidade de tempo igual a uma iteração temos que

$$e^{-1/\tau_i} = r_i, \tag{5.25}$$

a qual pode ser expandida como [36]:

$$r_i = e^{-1/\tau_i} = 1 - \frac{1}{\tau_i} + \frac{1}{2!\tau_i^2} - \frac{1}{3!\tau_i^3} + ... \tag{5.26}$$

Visto que em muitas aplicações τ_i é maior ou igual a 10 e r_i é menor que 1, façamos a seguinte aproximação

$$r_i \approx e^{-1/\tau_i} = 1 - \frac{1}{\tau_i}. \tag{5.27}$$

Igualando a Equação 5.25 e 5.27, obtemos

$$\tau_i = \frac{1}{\mu\left(f(\lambda_\mathbf{i}) + \frac{f''(\lambda_\mathbf{i})}{2}\sigma_n^2 + \frac{f''(\lambda_\mathbf{i})}{2}\sigma_e^2\right)}. \tag{5.28}$$

5.3 Análise da Convergência

A fim de determinar equações para o desajustes, utilizamos apenas os dois primeiros termos da Equação 5.7 e multiplicando ambos os lados por sua transposta, em

seguida tomando a experança, invocando os pressupostos 2 e assumindo que o vetor de erro de peso é pequeno, [14].

$$\begin{aligned} E[\mathbf{v}_{Q(k+1)}\mathbf{v^t}_{Q(k+1)}] &= E[\mathbf{v}_{Qk}\mathbf{v}_{Qk}^t] + 2\mu E[f(-\mathbf{v}_{Qk}^t\mathbf{x}_{Qk})]E[\mathbf{v}_{Qk}\mathbf{x}_{Qk}^t] \\ &+ \mu^2 E[f'^2(-\mathbf{v}_k^t\mathbf{x}_k)]\mathbf{R}(\sigma_n^2+\sigma_e^2) + \sigma_v^2 I. \end{aligned} \tag{5.29}$$

Para $\mathbf{R} = E[\mathbf{x}_k\mathbf{x}_k^t]$, $E[\mathbf{n}_v\mathbf{n}_v^t] = \sigma_v^2 I$ e $E[n_e^2] = \sigma_e^2$.

Usando o teorema apresentado por [36], ficamos:

$$\begin{aligned} E[\mathbf{v}_{Q(k+1)}\mathbf{v^t}_{Q(k+1)}] &= E[\mathbf{v}_{Qk}\mathbf{v}_{Qk}^t] + 2\mu E[f(-\mathbf{x}_{Qk}^t\mathbf{x}_{Qk})]E[\mathbf{v}_{Qk}\mathbf{v}_{Qk}^t] \\ &+ \mu^2 E[f'^2(-\mathbf{v}_k^t\mathbf{R}\mathbf{x}_k)](\sigma_n^2+\sigma_e^2) + \sigma_v^2 I. \end{aligned} \tag{5.30}$$

Para $\mathbf{v}_k^t\mathbf{R}\mathbf{x}_k = \sigma_{exc}^2$ [10]:

$$\begin{aligned} E[\mathbf{v}_{Q(k+1)}\mathbf{v^t}_{Q(k+1)}] &= E[\mathbf{v}_{Qk}\mathbf{v}_{Qk}^t] + 2\mu E[f(-\mathbf{x}_{Qk}^t\mathbf{x}_{Qk})]E[\mathbf{v}_{Qk}\mathbf{v}_{Qk}^t] \\ &+ \mu^2 E[f'^2(\sigma_{exc}^2)](\sigma_n^2+\sigma_e^2) + \sigma_v^2 I. \end{aligned} \tag{5.31}$$

Sabendo que $E[\mathbf{v}_{Qk}\mathbf{v}_{Qk}^t] = E[\mathbf{v}_{Qk}\mathbf{v}_{Qk}^t]$.

$$E[\mathbf{v}_{Qk}\mathbf{v}_{Qk}^t] = \frac{\mu^2 E[f'^2(\sigma_{exc}^2)](\sigma_n^2+\sigma_e^2) + \sigma_v^2 I}{2\mu E[f(-\mathbf{x}_{Qk}^t\mathbf{x}_{Qk})}. \tag{5.32}$$

Sabe-se que o excesso de erro médio quadrático na saída do filtro é causado por flutuações aleatórias (i) nos coeficientes de peso durante a adaptação, ver [10]. No qual assumimos a independência de $\mathbf{x}_k$ e $\mathbf{v}_k$, bem como a natureza não correlacionada do vetor de erro de peso:

$$\begin{aligned} E[(\mathbf{v}_{Qk}^t\mathbf{x}_{Qk})^2] &= E[(\sum_{i=1}^{N} v_{i,k}x_{i,k})^2] \\ &= \sum_{i=1}^{N} E[v_{i,k}^2]E[x_{i,k}^2], \end{aligned} \tag{5.33}$$

$$E[(\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})^2] = \frac{\mu^2 E[f'^2(\sigma_{exc}^2)](\sigma_n^2 + \sigma_e^2) + \sigma_v^2 I}{2\mu E[f(-\mathbf{x}_{Qk}^t \mathbf{x}_{Qk})} E[x_{i,k}^2]. \tag{5.34}$$

O desajuste é definido como excesso de erro quadrático médio sobre o erro mínimo quadrático médio e dado por [16]:

$$M = \frac{\xi excess}{\xi min}, \tag{5.35}$$

$$M = \frac{E[(\mathbf{v}_{Qk}^t \mathbf{x}_{Qk})^2]}{E[n_k^2]}, \tag{5.36}$$

onde $tr[.]$ denota a operação de traço $E[x_{i,k}^2] = tr[\mathbf{R}]$ e $E[n_k^2] = \sigma_n^2$ variância do ruído de medição. O desajuste pode ser dado por:

$$M = \frac{tr[\mathbf{R}]}{2\mu E[f(-\mathbf{x}_{Qk}^t \mathbf{x}_{Qk})]\sigma_n^2} (\mu^2 E[f'^2(\sigma_{exc}^2)](\sigma_n^2 + \sigma_e^2) + \sigma_v^2). \tag{5.37}$$

Para $\sigma_v^2 = 0$ e $\sigma_e^2 = 0$, não tem o ruído devido a quantização.

6 Simulações e Resultados

6.1 Dados da Simulação

Para determinar a precisão das equações derivadas, foram utilizados os seguintes algoritmos propostos na literatura: 1) O least mean mixed-norm (LMMN) [7], cuja função de custo é dada por $F(e) = \theta e^2 + (1-\theta)e^4$, com $\theta \in [0, 1]$ sendo o parâmetro de mistura; 2) o weighted even moment (WEM)[3], com função de custo $F(e) = \sum_{K=1}^{p} K^{p-1} e^{2K}$, onde K e p são inteiros e positivos; 3) O SIGMOIDAL [35], que usa a função de custo $F(e) = Ln(cosh(\alpha e))$ com α inteiro e positivo; 4) O least mean fourth (LMF) [45], com a função de custo dada por $F(e) = e^4$. Examinamos o desajuste experimental M_{exp}, o desajuste teórico M_{p1} proposto na equação (3.24), o M_{p2} proposto por [10] e o erro relativo dado por $\Delta = 100(M_{exp} - M_p)/M_{exp}$ para cada algoritmo desenvolvido em [7], [3], [35] e [45]. Em outra tabela analisamos também para cada algoritmo o Desajuste proposto em precisão finita de acordo com a Equação (5.37), o valor de C dado na Equação (4.2) é calculado de acordo com o experimento e especificado na descrição da tabela.

Para destacar os resultados, utilizamos um modelo ótimo representado pela planta: $P(z) = 0,0000z^{-1} + 0,2037z^{-2} + 0,5926z^{-3} + 0,2037z^{-4}$, cuja saída é corrompida por um ruído uniforme n_k de média zero, sinal de entrada x_k tem distribuição uniforme, limitado entre $[-1, 1]$. Nesta simulação a Relação Sinal-Ruído (SNR) será de 10^{-3}. Vamos escrever o sinal medido como $d_k = x_k + n_k$.

Os parâmetros de tamanhos de passos μ, assumiram os valores de acordo com as tabelas apresentadas. Para cada algoritmo e para cada passo μ realizamos simulações de Monte Carlo com 100 realizações cada uma. Mostramos os resultados em tabelas e gráficos para cada algoritmo analisado nas seções seguintes.

6.2 O algoritmo LMMN

Na tabela 6.1, apresentamos os resultados para o desajuste do algoritmo LMMN [7], calculamos a média do erro relativo de todas as simulações realizadas onde encontra-

mos o valor de $\Delta 1 = 4.11\%$para o desajuste teórico proposto já para o desajuste teórico proposto por [10] a média do erro relativo encontramos o valor de $\Delta 2 = 96.89\%$. Na tabela 6.2 e Figura 6.1, apresentamos os resultados para o desajuste do algoritmo LMMN [7] em sua forma quantizada.

Tabela 6.1: Desajuste do algoritmo LMMN, com potência do ruído $n_k = 10^{-3}$, Passo de Adaptação μ, Desajuste Experimental M_{exp}, Desajuste proposto M_{p1} com Erro Relativo de $\Delta 1$ (%) e Desajuste proposto por [10] M_{p2} com Erro Relativo de $\Delta 2$ (%)

μ	M_{exp}	M_{p1}	$\Delta 1$ (%)	M_{p2}	$\Delta 2$ (%)
0,020	$0,078234$	$0,071241$	$8,94$	$0,002331$	$97,02$
0,010	$0,036743$	$0,035621$	$3,06$	$0,001151$	$96,87$
0,009	$0,003409$	$0,032059$	$4,04$	$0,001009$	$96,98$
0,006	$0,022345$	$0,021372$	$4,35$	$0,000673$	$96,99$
0,003	$0,011037$	$0,010686$	$3,18$	$0,000348$	$96,84$
0,002	$0,007113$	$0,007126$	$0,19$	$0,000233$	$96,72$
0,001	$0,003586$	$0,003563$	$0,62$	$0,000117$	$96,73$
0,0001	$0,000389$	$0,000356$	$3,73$	$0,000012$	$96,89$

Tabela 6.2: Desajuste do algoritmo LMMN em sua forma quantizada, com potência do ruído $n_k = 10^{-3}$, C=400, número de Bits (Nº de bits) implementado, passo de adaptação μ_Q, desajuste experimental quantizado M_{Q-exp} e desajuste proposto quantizado M_Q.

Nº de bits	μ_Q	M_{Q-exp}	M_Q
7	$0,015$	14410	35354
8	$0,008$	14944	17677
9	$0,012$	1558	2946
10	$0,012$	583	883
11	$0,01$	34	55
12	$0,01$	$0,8014$	3.37
14	$0,01$	$0,0249$	0.0158
32	$0,01$	$0,0249$	0.0158

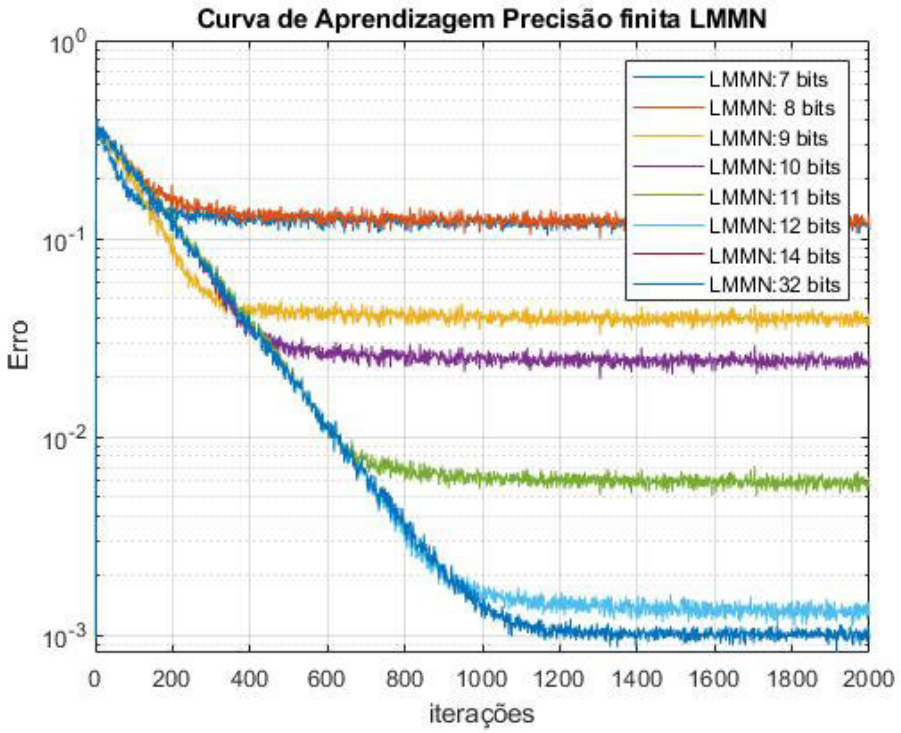


Figura 6.1: Curva de Aprendizagem dos Algoritmos LMMN Quantizado Sob Critério de Erro Geral, $\mu = 0,01$

6.3 O algoritmo WEM

Na tabela 6.3, apresentamos os resultados para o desajuste do algoritmo WEM [3], calculamos a média do erro relativo de todas as simulações realizadas onde encontramos o valor de $\Delta 1 = 38,79\%$ para o desajuste teórico proposto já para o desajuste teórico proposto por [10] a média do erro relativo encontramos o valor de $\Delta 2 = 99,52\%$. Na tabela 6.4 e Figura 6.2, apresentamos os resultados para o desajuste do algoritmo WEM [3] em sua forma quantizada.

Tabela 6.3: Desajuste do algoritmo WEM, com potência do ruído $n_k = 10^{-3}$, Passo de Adaptação μ, Desajuste Experimental M_{exp}, Desajuste proposto M_{p1} com Erro Relativo de $\Delta 1$ (%) e Desajuste proposto por [10] M_{p2} com Erro Relativo de $\Delta 2$ (%)

μ	M_{exp}	M_{p1}	$\Delta 1$ (%)	M_{p2}	$\Delta 2$ (%)
0,020	$0,088071$	$0,049961$	$43,27$	$0,000423$	$99,52$
0,010	$0,041758$	$0,024981$	$40,18$	$0,000194$	$99,53$
0,009	$0,038008$	$0,022490$	$40,83$	$0,000175$	$99,54$
0,006	$0,024924$	$0,014994$	$39,84$	$0,000114$	$97,54$
0,003	$0,012358$	$0,007497$	$39,33$	$0,000058$	$99,53$
0,002	$0,008464$	$0,004999$	$40,94$	$0,000038$	$99,55$
0,001	$0,003594$	$0,002501$	$30,41$	$0,000020$	$99,44$
0,0001	$0,000388$	$0,000250$	$35,55$	$0,000002$	$99,48$

Tabela 6.4: Desajuste do algoritmo WEM em sua forma quantizada com potência do ruído $n_k = 10^{-3}$, C=400, número de Bits (Nº de bits) implementado, passo de adaptação μ_Q, desajuste experimental quantizado M_{Q-exp} e desajuste proposto quantizado M_Q.

Nº de bits	μ_Q	M_{Q-exp}	M_Q
7	$0,009$	34379	30489
8	$0,008$	34723	18190
9	$0,011$	11833	7865
10	$0,01$	6410	859
11	$0,01$	577	245
12	$0,01$	30	$11,2$
14	$0,01$	$0,8544$	$0,3084$
32	$0,01$	$0,8544$	$0,3084$

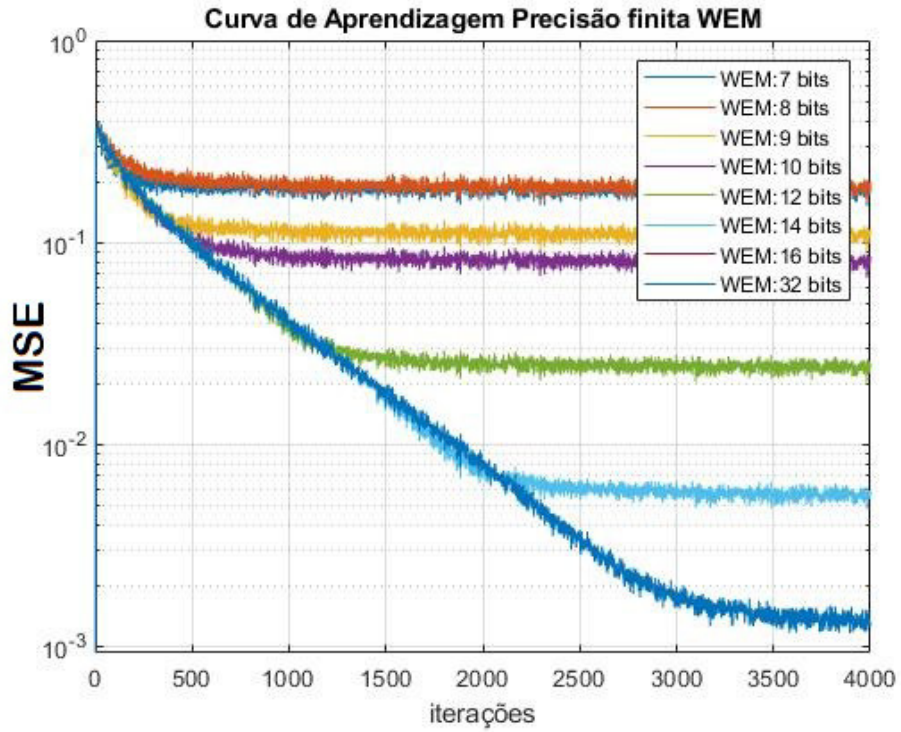


Figura 6.2: Curva de Aprendizagem dos Algoritmos WEM Quantizado Sob Critério de Erro Geral, $\mu = 0,01$

6.4 O algoritmo SIGMOIDAL

Na tabela 6.5, apresentamos os resultados para o desajuste do algoritmo SIGMOIDAL [35], calculamos a média do erro relativo de todas as simulações realizadas onde encontramos o valor de $\Delta 1 = 31.61\%$ para o desajuste teórico proposto já para o desajuste teórico proposto por [10] a média do erro relativo encontramos o valor de $\Delta 2 = 98.99\%$. Na tabela 6.6 e Figura 6.3, apresentamos os resultados para o desajuste do algoritmo SIGMOIDAL [35] em sua forma quantizada.

Tabela 6.5: Desajuste do algoritmo SIGMOIDAL, com potência do ruído $n_k = 10^{-3}$, Passo de Adaptação μ, Desajuste Experimental M_{exp}, Desajuste proposto M_{p1} com Erro Relativo de $\Delta 1$ (%) e Desajuste proposto por [10] M_{p2} com Erro Relativo de $\Delta 2$ (%)

μ	M_{exp}	M_{p1}	$\Delta 1$ (%)	M_{p2}	$\Delta 2$ (%)
0,020	$0,524737$	$0,271309$	$48,30$	$0,010167$	$98,06$
0,010	$0,188154$	$0,135654$	$27,90$	$0,001949$	$98,96$
0,009	$0,096054$	$0,122089$	$27,10$	$0,001226$	$98,72$
0,006	$0,059993$	$0,081393$	$35,67$	$0,000794$	$98,68$
0,003	$0,056902$	$0,040697$	$28,48$	$0,000356$	$99,37$
0,002	$0,037418$	$0,027131$	$27,49$	$0,000168$	$99,55$
0,001	$0,018604$	$0,013565$	$27,09$	$0,000134$	$99,28$
0,0001	$0,001961$	$0,001356$	$30,86$	$0,000013$	$99,32$

Tabela 6.6: Desajuste do algoritmo SIGMOIDAL em sua forma quantizada, com potência do ruído $n_k = 10^{-3}$, C=400, número de Bits (Nº de bits) implementado, passo de adaptação μ_Q, desajuste experimental quantizado M_{Q-exp} e desajuste proposto quantizado M_Q.

Nº de bits	μ_Q	M_{Q-exp}	M_Q
7	$0,008$	56784	67378
8	$0,008$	58811	33690
9	$0,012$	5426	5615
10	$0,011$	2045	1684
11	$0,01$	123	$105,2$
12	$0,01$	$5,04$	$6,42$
14	$0,01$	$0,0061$	$0,012$
32	$0,01$	$0,0061$	$0,0052$

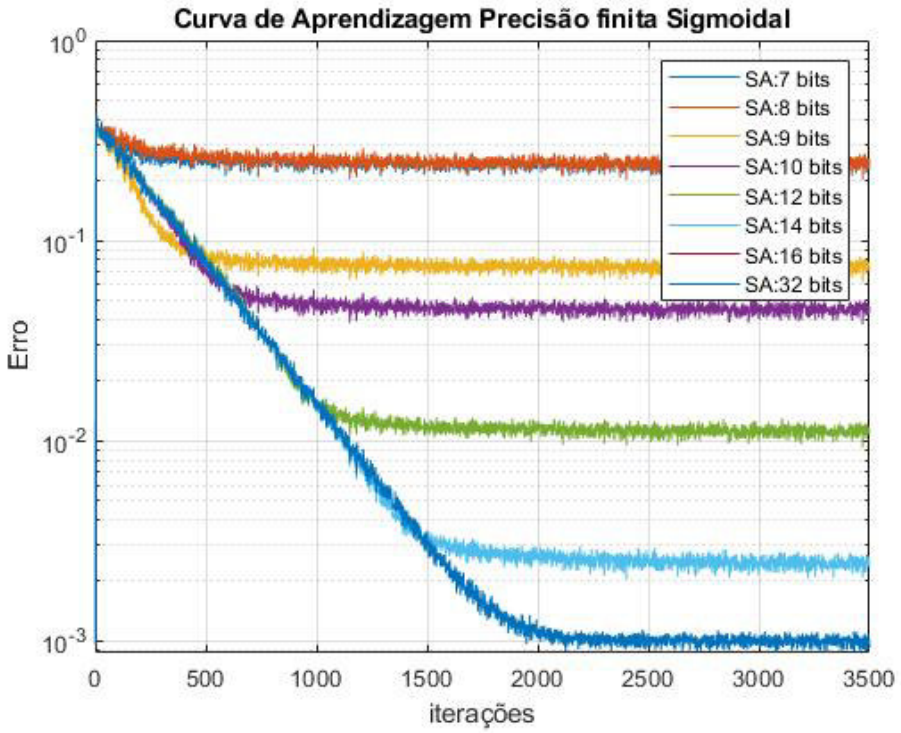


Figura 6.3: Curva de Aprendizagem dos Algoritmos SIGMOIDAL Quantizado Sob Critério de Erro Geral, $\mu = 0,01$

6.5 O algoritmo LMF

Na tabela 6.7, apresentamos os resultados para o desajuste do algoritmo LMF [10], calculamos a média do erro relativo de todas as simulações realizadas onde encontramos o valor de $\Delta 1 = 31,61\%$ para o desajuste teórico proposto já para o desajuste teórico proposto por [10] a média do erro relativo encontramos o valor de $\Delta 2 = 98,99\%$. Na tabela 6.8 e Figura 6.4, apresentamos os resultados para o desajuste do algoritmo LMF [10] em sua forma quantizada.

Tabela 6.7: Desajuste do algoritmo LMF, com potência do ruído $n_k = 10^{-1}$, Passo de Adaptação μ, Desajuste Experimental M_{exp}, Desajuste proposto M_{p1} com Erro Relativo de $\Delta 1$ (%) e Desajuste proposto por [10] M_{p2} com Erro Relativo de $\Delta 2$ (%)

μ	M_{exp}	M_{p1}	$\Delta 1$ (%)	M_{p2}	$\Delta 2$ (%)
0,007	$0,004354$	$0,003991$	$8,34$	$0,001075$	$75,31$
0,005	$0,003264$	$0,003326$	$1,90$	$0,000761$	$76,69$
0,0025	$0,001532$	$0,001663$	$8,55$	$0,00038$	$75,20$
0,00125	$0,000802$	$0,000890$	$10,97$	$0,00019$	$76,31$
0,0006	$0,000418$	$0,000399$	$4,66$	$0,00009$	$78,26$
0,0003	$0,000192$	$0,000199$	$3,65$	$0,00004$	$76,56$
0,00015	$0,000099$	$0,000097$	$2,02$	$0,00002$	$77,78$

Tabela 6.8: Desajuste do algoritmo LMF em sua forma quantizada, com potência do ruído $n_k = 10^{-1}$, C=8,número de Bits (Nº de bits) implementado, passo de adaptação μ_Q, desajuste experimental quantizado M_{Q-exp} e desajuste proposto quantizado M_Q.

Nº de bits	μ_Q	M_{Q-exp}	M_Q
7	0	14	inf
8	$0,008$	$0,2647$	2
9	$0,006$	$0,3623$	$0,957$
10	$0,005$	$0,0175$	$0,0186$
11	$0,005$	$0,0031$	$0,00704$
12	$0,005$	$0,0056$	$0,00352$
32	$0,005$	$0,0055$	$0,0024$
64	$0,005$	$0,0055$	$0,0024$

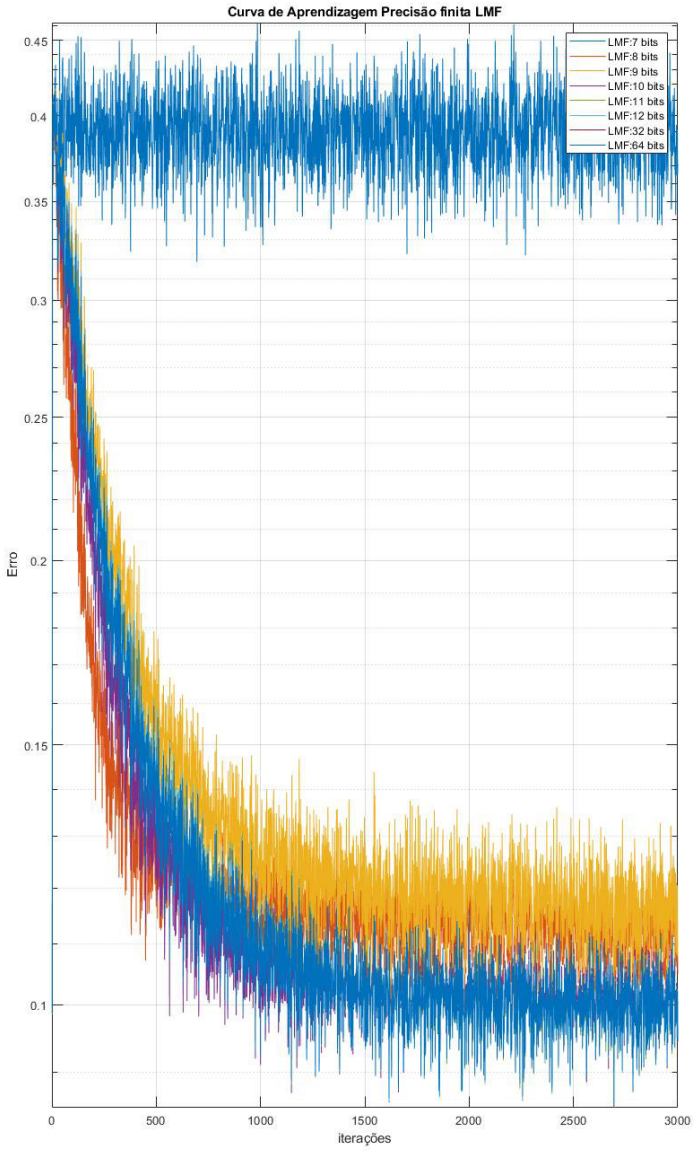


Figura 6.4: Curva de Aprendizagem dos Algoritmos LMF Quantizado Sob Critério de Erro Geral, $\mu = 0,005$

7 Circuito Eletrônico baseado em algoritmo Sigmoidal de ponto fixo aplicado em Nanossatélite

7.1 Fundamentos da implementação

Nesta seção, apresentamos os conceitos da Tecnologia FPGA, o padrão Cubesat e o Algoritmo Sigmoidal de Ponto Fixo.

7.1.1 Tecnologia FPGA

Uma FPGA, ou Field Programmable Gate Array, é na usa essência um dispositivo programável e a maior aproximação possível de um produto comercial e barato que possibilite desenhar de raiz o seu próprio chip. Esta tecnologia permite desenhar e implementar virtualmente qualquer função digital que possa ser possível imaginar e conceber dentro de um único chip universal, sendo a diferença entre um FPGA e qualquer outro chip semelhante é que este não faz efetivamente nada por si só, não tem nenhum propósito nenhuma função embutida nele [41] e [29].

A FPGA apresenta-se como um chip que pode desempenhar as mais variadas funções sem restrições de complexidade, aliado à sua enorme vantagem de velocidade em função do paralelismo, torna-se um ambiente de hardware ideal para implementações de filtros adaptativos para processamento de sinais. Devido aos requisitos de alto desempenho e a crescente complexidade do processamento de sinais digitais e diversidade de aplicações, são necessários filtros com um comprimento elevado para aumentar o desempenho, de forma a produzir uma alta taxa de amostragem. Por isso os FPGAs tornaram-se um meio extremamente econômico de realizar e processar algoritmos de processamento de sinais digitais que sejam computacionalmente intensivos além de melhorar o desempenho geral do sistema [29] e [29].

7.1.2 Nanossatélite na perspectiva do padrão CubeSat

Nesta subsessão apresentamos os conceitos relacionados à nanossatélites sobre a perspectiva do padrão CubeSat, os subsistemas que o compõe e as condições aos quais estes podem ser expostos em ambiente espacial.

O padrão CubeSat designa um satélite miniaturizado e normalmente utiliza componentes eletrônicos e materiais acessíveis aos pesquisadores e a toda comunidade acadêmica um meio prático para o desenvolvimento de Satélites miniaturizados. Esse padrão surgiu do esforço colaborativo dos pesquisadores Jordi Puig-Suari e Bob Twiggs da Universidade Politécnica do Estado da Califórnia (Cal Poly) e da Universidade de Stanford, respectivamente [34].

Devido ao seu custo reduzido no desenvolvimento desses satélites, o padrão CubeSat, possui grande aceitação, quando comparado com os custos envolvidos em projetos com satélites de grande porte. Esse tipo de satélite possui dimensões de 10x10x10 cm e massa de ate 1,33 Kg. Um CubeSat com essas especificações é denominado de 1U ou uma unidade. A simplificação da infraestrutura de satélites torna possível projetar e produzir satélites funcionais a um baixo custo. Esse sistema é escalável ao longo do eixo, através de incrementos de "1U", permitindo a criação de CubeSats "2U"(20x10x10 cm) e "3U"(30x10x10 cm).

O CubeSat também especifica como padrão os protocolos de comunicação entre os subsistemas do satélites, todos esses requisitos podem ser atendidos usando os chamados componentes COTS (do inglês Comercial off-the-shelf), que são componentes eletrônicos de custo reduzidos e encontrados com facilidade no mercado, o que torna possível reduzir o tempo de desenvolvimento e o custo do projeto. Em todo caso, essa padrão tem como premissa o desenvolvimento de tecnologias espaciais com custo reduzidos, no entanto deve-se manter os requisitos de confiabilidade do sistema completo.

O Cubesat compreende vários objetivos de alto nível. A simplificação da arquitetura desses satélites torna possível projetar e produzir satélites funcionais a um baixo custo. O formato em módulos ou subsistemas independentes que executam diferentes tarefas formam um sistema completo de complexo. No entanto alguns desses subsistemas são relevantes para o funcionamento do satélite e outros variam com o tipo de missão que o mesmo será destinado. São exemplos de subsistemas o computador de bordo (OBC – On-board Computer), o módulo de energia (EPS – Electrical Power Supply), o sistema de

comunicação por rádio frequência (RF), o controle de atitude (ADCS – Attitude Determination and Control System), a carga útil (Payload), etc. Na Figura abaixo é possível verificar uma representação dos subsistemas de um CubeSat.

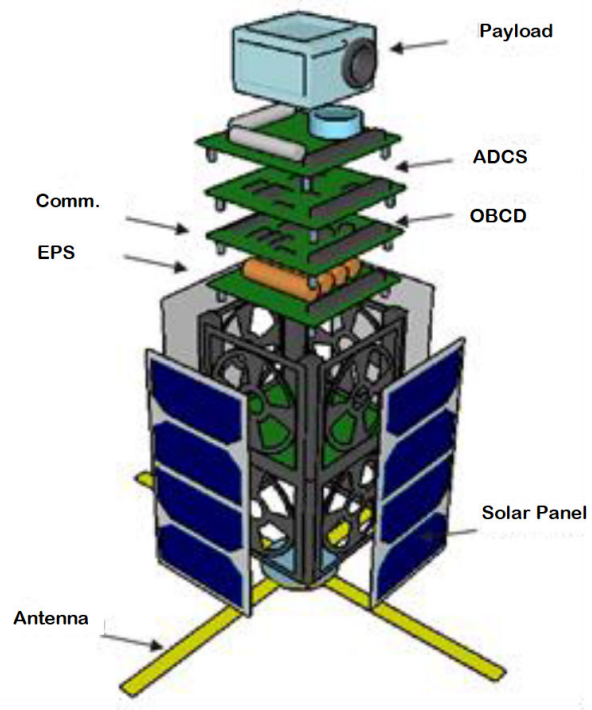


Figura 7.1: Subsistemas de um CubeSat

O satélite precisa de um subsistema capaz de gerenciar todos os outros subsistemas. Um computador de bordo (OBC - On-board Computer) é o subsistema responsável por garantir o funcionamento de todos os subsistemas em um CubeSat [32]. Os principais componentes são: microcontrolador central, memória RAM, memória Flash, memória de armazenamento para os dados obtidos, barramento de dados, barramento de controle, entradas de periféricos e clock. As principais funções de um OBC são [39]:

1. Monitoramento e armazenamento da telemetria gerada pelo satélite para transmissão futura;

2. Codificação e decodificação dos pacotes oriundos da comunicação com a estação base;

3. Processamento de telecomandos da estação base;

4. Monitoramento dos subsistemas, analisando o seu correto funcionamento e reinicializando o sistema quando necessário.

7.1.3 O algoritmo Sigmoidal de Ponto Fixo

O satélite precisa de um subsistema capaz de filtrar um sinal de rádio, devido a diversas interferências internas e externas. Este sistema é responsável por garantir um sinal sem ruído na transmissão dos dados entre o satélite e operador ou subsistemas acoplados. Uma forma de garantir essa transmissão sem ruídos é através de algoritmos de filtragem adaptativa. Devido as características dos dispositivos de FPGAs como custo e eficiência, os algoritmos de alta ordem (HOS) são indicados para resolução desse problema [35].

Como mostrado o algoritmo Sigmoidal é formado pelo gradiente da função de custo logarítmica do cosseno hiperbólico aplicado sobre o erro ($Ln[cosh(e_k)]$) [35].

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha\mu tanh(\alpha e_k)\mathbf{x}_k, \tag{7.1}$$

O algoritmo Sigmoidal de Ponto Fixo, consiste na aproximação dessa função $tanh(\alpha e_k)$ em série de Taylor, α é uma constante de inclinação da curva. Ficamos com o seguinte algoritmo em expansão de Taylor até a terceira ordem:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha\mu(\frac{2^2}{15} - \frac{2e^3}{3} + 2e^3)\mathbf{x}_k, \tag{7.2}$$

Este é o algoritmo Sigmoidal de Ponto Fixo implementado no circuito eletrônico de FPGA. O presente algoritmo foi desenhado em formato PC/104 para ser implementado em nanossatélites do tipo cubesat.

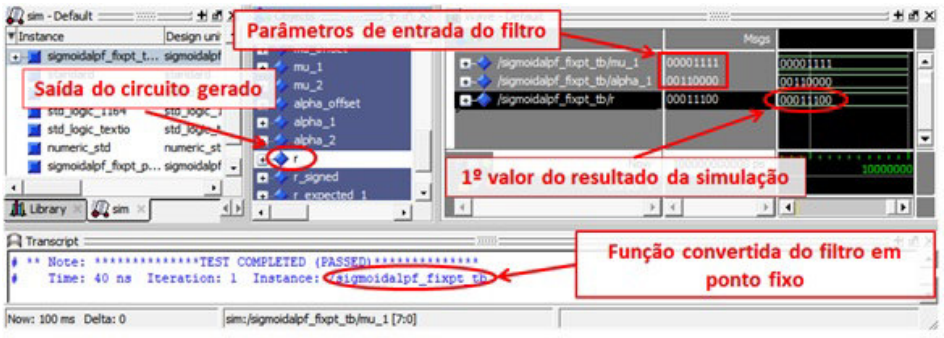
7.2 Realização do algoritmo Sigmoidal de Ponto Fixo em FPGA

O circuito que será apresentado baseia-se na teoria de filtragem adaptativa. Trata-se de uma placa de circuito integrado contendo um microcontrolador e um circuito de lógica reconfigurável baseado em FPGA. O objetivo do circuito é realizar a filtragem de sinais utilizando um algoritmo adaptativo de ponto fixo baseado em função sigmoidal. O presente circuito foi desenhado em formato PC/104 para ser implementado em nanossatélites do tipo cubesat.

7.2.1 Geração do circuito em VHDL

Uma vez que as simulações foram realizadas, foi escrita uma função no MATLAB, contendo o algoritmo para realizar o filtro usando a função sigmoidal. Tal função contém a implementação mais básica do filtro, com operações matemáticas mais elementares, uma vez que se torna muito dispendioso escrever operações matemáticas complexas em VHDL. Uma dessas implementações envolve a função tangente, representada a partir da série de Taylor utilizando apenas três termos. Após, foi usada a ferramenta HDL Coder do MATLAB, para efetuar a conversão da função que efetua a filtragem de sinais passados como parâmetro de entrada, juntamente com o ruído, e os parâmetros do filtro. Esta ferramenta apresenta duas opções de linguagem mais usadas para elaboração do circuito,sendo elas Verilog e VHDL. Neste trabalho foi usado a linguagem VHDL. A ferramenta HDL Coder do MATLAB necessita de duas entradas para efetuar a geração do código em linguagem de definição de hardware (Hardware Definition Language, HDL) do circuito. A primeira entrada é a função a ser convertida em HDL, enquanto a segunda é um arquivo que faz o papel de teste de bancada, comumente chamado de Testbench. O teste de bancada é um script capaz de chamar a função que será convertida em HDL, informando os parâmetros que serão usados nas simulações realizadas pela ferramenta. Foi criado um script no MATLAB contendo a chamada à função do filtro sigmoidal contendo a passagem dos parâmetros do filtro. Após, a ferramenta HDL Coder foi usada, e nela foram efetuadas algumas configurações, tais como a seleção da entrada de dados em ponto fixo e a quantidade de bits a serem usados nas representações dos valores em ponto flutuante. Na figura 7.2 é possível observar as configurações dos tipos de dados em ponto fixo de cada uma das variáveis envolvidas na implementação do filtro no MATLAB. Alguns ajustes foram feitos para proporcionar uma melhor precisão nos cálculos, e foram destacados em vermelho. Tal ajuste foi necessário para manter a precisão nos cálculos em ponto flutuante e os resultados serem aproximados se comparados com as simulações realizadas com algoritmos em ponto flutuante. Percebeu-se durante os experimentos que, dependendo dos valores armazenados em cada variável, era necessário aumentar ou diminuir a quantidade de bits de informação, ou mesmo a quantidade de bits usados para representar as casas decimais. Durante a geração do código em HDL usando a ferramenta HDL Coder, são efetuados testes automatizados usando o Testbench criado e informado como entrada. Vale ressaltar que a ferramenta HDL Coder gera um conjunto de scripts para realizar a validação do circuito utilizando a ferramenta de simulação ModelSim, am-

plamente utilizada em simulação de circuitos gerados em VHDL. Durante a geração de código, a ferramenta ModelSim incorporada ao MATLAB é chamada e é possível realizar a simulação do Testbench criado e, dessa forma, avaliar o resultado obtido a partir do circuito VHDL gerado. Na figura X é possível observar a tela do ModelSim durante o processo de simulação do circuito em VHDL gerado e avaliar a saída do circuito. É possível ver os parâmetros, alpha e mu, usados no filtro, convertidos para ponto fixo, e representados em binário. Também é possível se ver o primeiro valor do conjunto de dados resultante da simulação, e dessa forma, comparar com o resultado da simulação efetuada pelo MATLAB. Após gerar a conversão da função do filtro e do Testbench, o MATLAB salva em disco os arquivos resultantes da conversão em VHDL. Na Figura 7.2, é possível ver o nome do arquivo da função convertida em VHDL usando aritmética de ponto fixo.



Variables | Function Replacements | Output

Variable	Type	Sim Min	Sim Max	Whole ...	Proposed Type
Input					
mu	double	0.12	0.12	No	numerictype(1, 8, 7)
alpha	double	3	3	Yes	numerictype(1, 8, 4)
Output					
r	1 x 8 double	-0.03	0.05	No	numerictype(1, 8, 8)
Local					
x	14 x 1 double	0.01	0.1	No	numerictype(1, 8, 7)
noise	14 x 1 double	0.01	0.1	No	numerictype(1, 8, 7)
aux1	double	0	0.02	No	numerictype(1, 8, 7)
aux2	double	-0.04	0.09	No	numerictype(1, 8, 7)
aux3	double	0	0.01	No	numerictype(1, 8, 7)
pow1	double	0	1	No	numerictype(1, 16, 8)
pow2	double	0	1	No	numerictype(1, 16, 8)
temp	double	1.08	1.08	No	numerictype(1, 8, 7)
w	7 x 1 double	0	0.19	No	numerictype(1, 8, 7)
buffer	7 x 1 double	0	0.1	No	numerictype(1, 8, 7)
desired	1 x 8 double	-0.03	0.09	No	numerictype(1, 8, 7)
k	double	1	10	Yes	numerictype(1, 8, 7)
i	double	1	8	Yes	numerictype(1, 8, 7)
s	double	0	0.05	No	numerictype(1, 8, 7)
jj	double	1	8	Yes	numerictype(1, 16, 8)
j	double	1	14	Yes	numerictype(1, 8, 7)
jjj	double	1	8	Yes	numerictype(1, 16, 8)

Figura 7.2: Função convertida em VHDL usando aritmética de ponto fixo

7.2.2 Implementação do algoritmo em FPGA

Após o processo de geração dos arquivos em VHDL para o filtro sigmoidal de ponto fixo, foi feita a montagem de um experimento físico, utilizando um sistema contendo dispositivos eletrônicos capazes de realizar a filtragem de dados e apresentação dos resultados. Na figura 7.3 (a), é apresentado o funcionamento básico do método proposto, contendo os blocos básicos constituintes do sistema. Os blocos principais são formados por um microcontrolador (MCU) e um FPGA. O MCU irá assumir o papel do OBC do CubeSat, enquanto que o FPGA será responsável pelo armazenamento dos dados em uma memória SRAM, bem como por comportar o circuito para filtragem dos dados oriundos do MCU, assim como transmitir os dados filtrados de volta para o MCU. Em razão de limitações dos componentes utilizados no experimento, não foi possível realizar a comunicação bilateral entre o FPGA e o MCU. Desta forma optou-se por deixar os dados do experimento gravados diretamente dentro do FPGA e efetuar apenas a comunicação de saída dos dados filtrados, ocupando praticamente todos os pinos de entrada e saída do MCU. Na figura 7.3 (b) é possível observar a quantidade de pinos empregada na comunicação e controle entre o FPGA e o MCU. O barramento de endereços possui 6 pinos, permitindo o endereçamento de até 2 elevado a 6 posições de memória, totalizando 64 endereços disponíveis para dados dentro da SRAM gerada no FPGA. O barramento de leitura de dados possui 8 pinos, possibilitando o envio de dados de 8 bits para a memória. O barramento de controle de operação de leitura foi usado para indicar para o MCU quando um novo dado estivesse disponível para transmissão. Dessa forma, foram usados 15 pinos para este experimento.

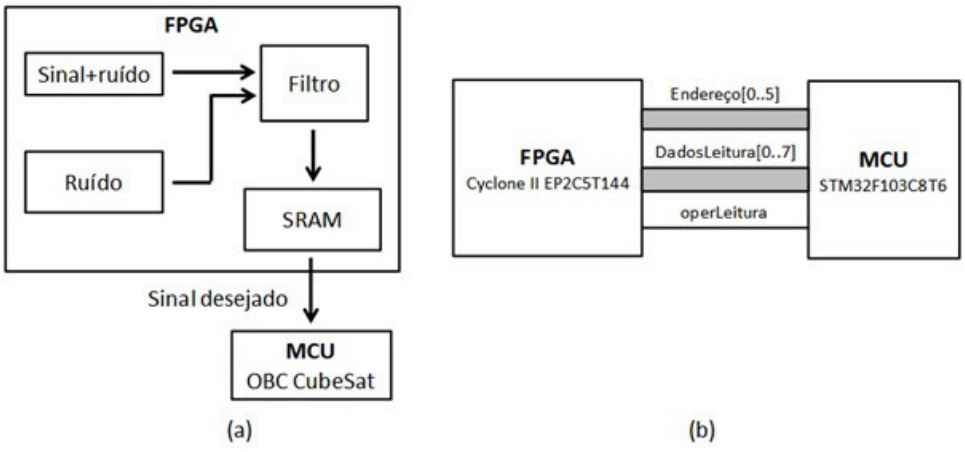


Figura 7.3: (a) Funcionamento básico do método proposto; (b) Quantidade de pinos empregada na comunicação e controle entre FPGA e o MCU

Na figura 7.4 é possível observar o diagrama esquemático do circuito proposto, contendo os detalhamentos das ligações entre o FPGA e o MCU, elaborado no software KiCad. À esquerda, o chip FPGA utilizado e, à direita, o MCU. Pode-se perceber que praticamente todos os pinos disponíveis no MCU foram utilizados. Os pinos restantes não puderam ser utilizados por se tratarem de pinos especiais, reservados para o MCU. Por exemplo, os pinos de comunicação serial, PA9 e PA10. Estes foram usados para conectar com um módulo de comunicação serial e conectado ao computador, permitindo visualizar a saída de dados resultantes do experimento. Os pinos de gravação de programas no MCU, PA13 e PA14, foram utilizados para conectar o módulo de gravação de firmware ST Link V2. O pino PC13 foi usado para conectar um LED, permitindo a comunicação visual com o usuário, indicando estados do sistema, tais como, efetuando cálculos, transmitindo dados, dentre outros. Os demais são pinos reservados e não puderam ser usados para comunicação entre o MCU e o FPGA, o que impossibilitou mais funcionalidades no protótipo, tais como a criação de um barramento de entrada de dados.

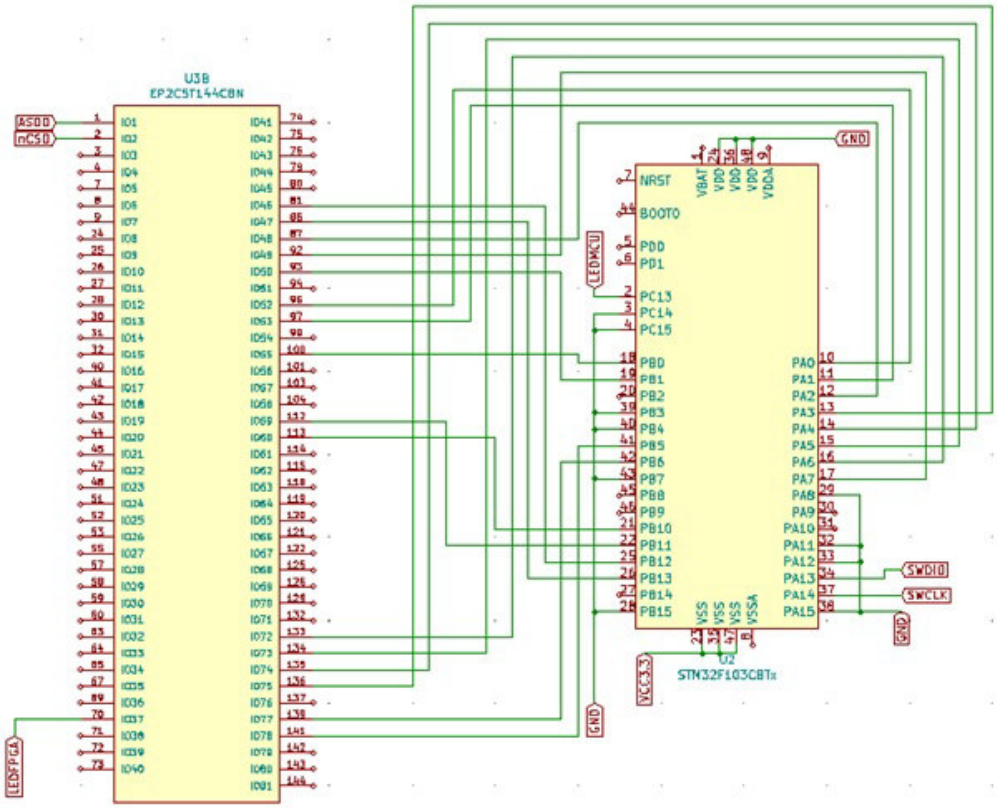


Figura 7.4: Diagrama esquemático do circuito proposto

Usando os softwares Altera Quartus II Web Edition e o MATLAB R2021b, o circuito para realizar o experimento foi desenvolvido e simulado. A função referente ao filtro baseado no algoritmo sigmoidal foi convertido para VHDL usando a ToolBox HDL

Coder, do MATLAB. Foram usadas as entradas de dados em ponto fixo e os arquivos VHDL exportados foram incorporados ao projeto do experimento no Altera Quartus II. Usando o visualizador RTL do Altera Quartus II, é possível produzir a visão RTL do circuito gerados a partir do projeto criado, conforme pode-se observar na figura 7.5. Observa-se que o circuito é formado por dois blocos principais, destacados na cor verde, responsáveis pela filtragem usando o algoritmo sigmoidal de ponto fixo e a memória de trabalho do tipo SRAM. Ao redor, tem-se as portas de entrada e saída, bem como os blocos adicionais, responsáveis por auxiliar no processo de interligação dos componentes. Percebe-se que ferramentas de geração de código alcançaram um nível de maturidade a ponto de acelerar o processo de desenvolvimento de aplicações em FPGA aplicados a sistemas embarcados espaciais.

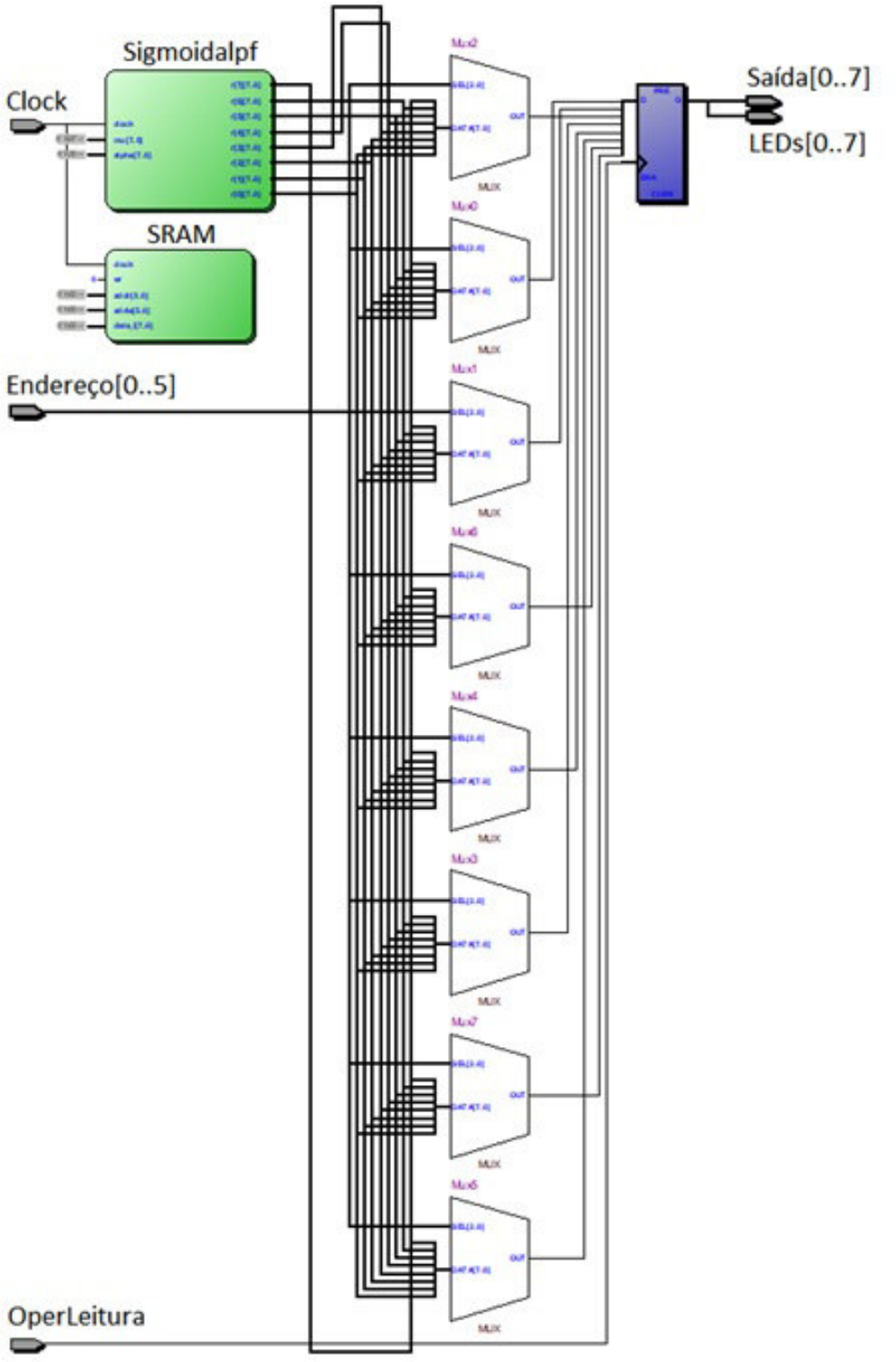


Figura 7.5: Visão RTL do circuito gerado a partir do projeto criado

Para os experimentos implementados fisicamente, foram usados os componentes FPGA Altera Cyclone II EP2C5T144C8, disponível em uma placa Cyclone II EP2C5 Mini Dev Board, e o microcontrolador (Microcontroller Unit, MCU) STMF103C8T6 da STMicroeletronics. Tanto o FPGA quanto o MCU foram embarcados em uma placa de circuito universal, de dimensões 10 cm x 10 cm, onde também foram acoplados LEDs

para visualização de dados do sistema e os estados do circuito armazenados no FPGA. Na figura X é possível observar a placa desenvolvida para os experimentos. Optou-se neste protótipo experimental por confeccionar em uma placa universal pelo baixo custo e rápido desenvolvimento.

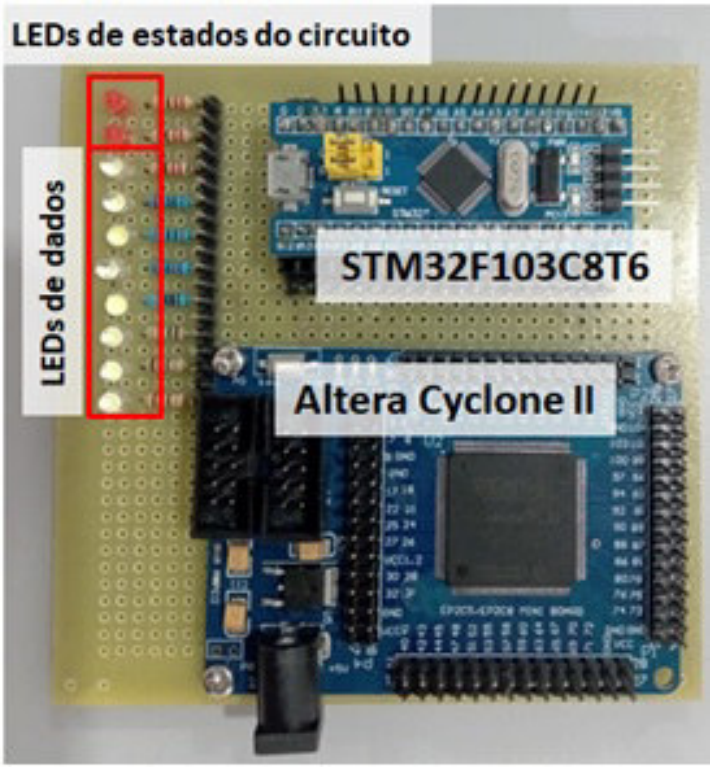


Figura 7.6: Placa de desenvolvimento para o experimento

Após a confecção da placa, foram definidos os pinos do FPGA que seriam interligados ao MCU, bem como os pinos que seriam ligados aos LEDs de estado e aos LEDs de dados. Os pinos conectados ao MCU foram utilizados para endereçamento da memória SRAM no FPGA, realizar leitura de dados a partir da memória e envio do sinal de controle para efetuar a leitura. Os pinos do MCU conectados à saída de dados do FPGA tiveram resistores de PULL UP internos habilitados, impedindo que valores flutuantes nos pinos ocasionassem erros de leitura no MCU. O MCU foi programado usando o programador ST Link V2 e a codificação do firmware foi feita utilizando a IDE Arduino. A depuração do sistema foi realizada através de um módulo conversor USB-Serial, conectado à porta USB de um computador pessoal.

7.2.3 Resultados da implementação

Objetivando avaliar o desempenho do circuito desenvolvido, foi desenvolvido um experimento que envolveu a síntese do circuito gerado a partir do HDL Coder em VHDL usando o Altera Quartus II e o chip FPGA Altera Cyclone II EP2C5T144C8. Na Figura 7.7 pode-se observar o relatório da síntese fornecido pelo Altera Quartus II, onde é possível se ter uma ideia da quantidade de blocos lógicos ocupados pelo circuito gerado, total de pinos utilizados, e outros recursos do chip EP2C5T144C8, alocados para esta tarefa.

Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Full Version
Revision Name	MCU_RAM
Top-level Entity Name	MCU_RAM
Family	Cyclone II
Device	EP2C5T144C8
Timing Models	Final
Total logic elements	7 / 4,608 (< 1 %)
Total combinational functions	7 / 4,608 (< 1 %)
Dedicated logic registers	7 / 4,608 (< 1 %)
Total registers	7
Total pins	25 / 89 (28 %)
Total virtual pins	0
Total memory bits	0 / 119,808 (0 %)
Embedded Multiplier 9-bit elements	0 / 26 (0 %)
Total PLLs	0 / 2 (0 %)

Figura 7.7: Relatório da síntese fornecido pelo Altera Quartus II

Na Figura 7.8 é possível observar a placa de prototipagem confeccionada em funcionamento, apresentando nos LEDs um dos resultados da filtragem do sinal armazenado na memória SRAM e transportada para o MCU através do barramento de dados. Pode-se ver nos detalhe à esquerda o módulo conversor serial/USB usado na comunicação entre o MCU e o computador, à direita o módulo de gravação STLink V2 e abaixo, nos LEDs, um dos resultados da filtragem do sinal armazenado na SRAM do circuito armazenado no chip FPGA.

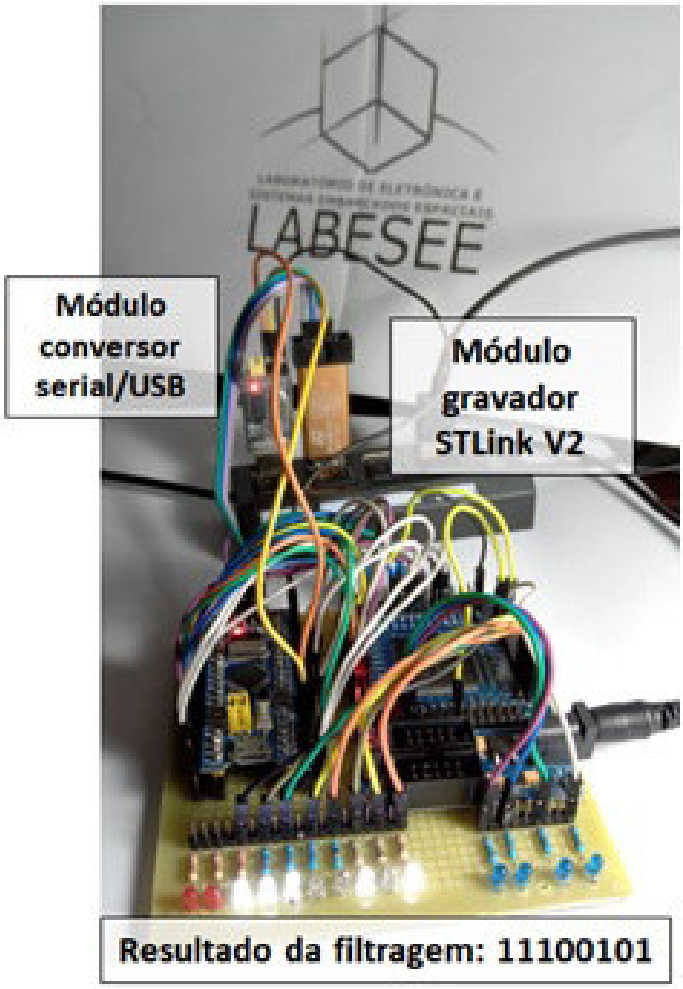


Figura 7.8: Placa de prototipagem confeccionada em funcionamento

Foi usado o programa de computador AccessPort para se ter acesso à saída dos resultados do experimento, através do módulo conversor serial/USB. Na figura 7.8 é possível observar a saída de um dos vários testes efetuados, e a comparação com as simulações realizadas pelo HDL Coder. Na Figura 7.9 é apresentada a resposta do filtro aplicado aos dados armazenados na SRAM, exibidos a partir dos LEDs de dados. Desta forma, o processamento realizado no FPGA pode ser conferido a partir dos LEDs da placa e também no PC, a partir do programa AccessPort. No canto inferior esquerdo, pode-se ver a captura da tela do programa AccessPort apresentando os dados retornados pelo MCU e que foram, por sua vez, retornados pelo FPGA. À direita, o resultado da simulação efetuada durante a geração do código VHDL a partir da ferramenta HDL Coder. Percebe-se, dessa forma, que os dados simulados no computador são iguais aos dados calculados pelo filtro gravado no FPGA.

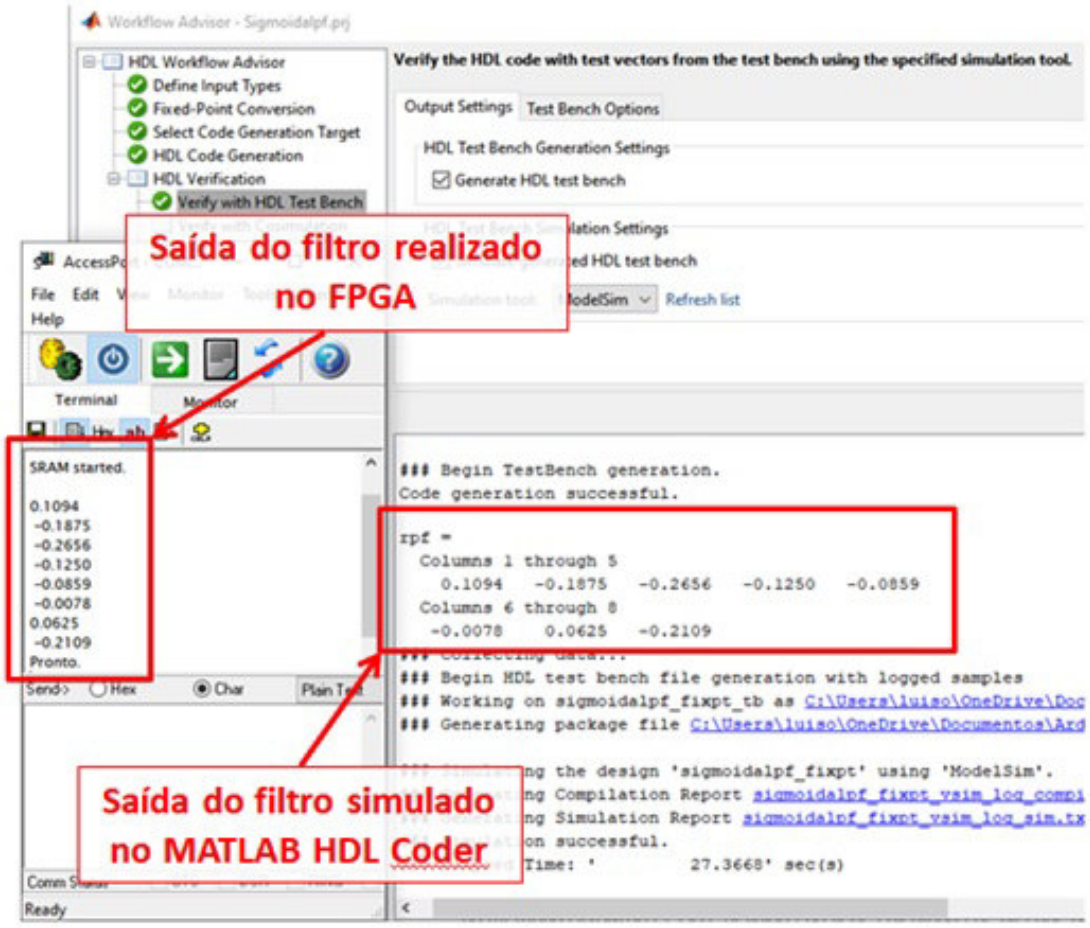


Figura 7.9: Resposta do filtro aplicado ao armazenamento

Assim que o experimento foi concluído, foi iniciada a fase de desenho do circuito que será, futuramente, produzido e embarcado como carga útil de um possível nanossatélite em uma missão espacial. O diagrama esquemático esquemático completo, elaborado no software KiCad, encontra-se no Apêndice 1. Neste diagrama, além do FPGA e do MCU, e suas conexões entre pinos, está descrito outros componentes, tais como uma memória flash utilizada para armazenar de forma permanente o circuito sintetizado a partir do software Altera Quartus II, um cristal de frequencia igual a 50 MHz, para servir de clock para o FPGA, um circuito de conversão de tensão de 3,3 V para 1,2 V, para alimentação do FPGA, e conectores para gravação de firmware no MCU, e para atualização do circuito no FPGA. Após definição dos componentes, layout da placa no padrão PC/104, com furações no padrão CubeSat, a placa de circuito impresso foi desenhada e é possível observar o resultado final na figura 7.10.

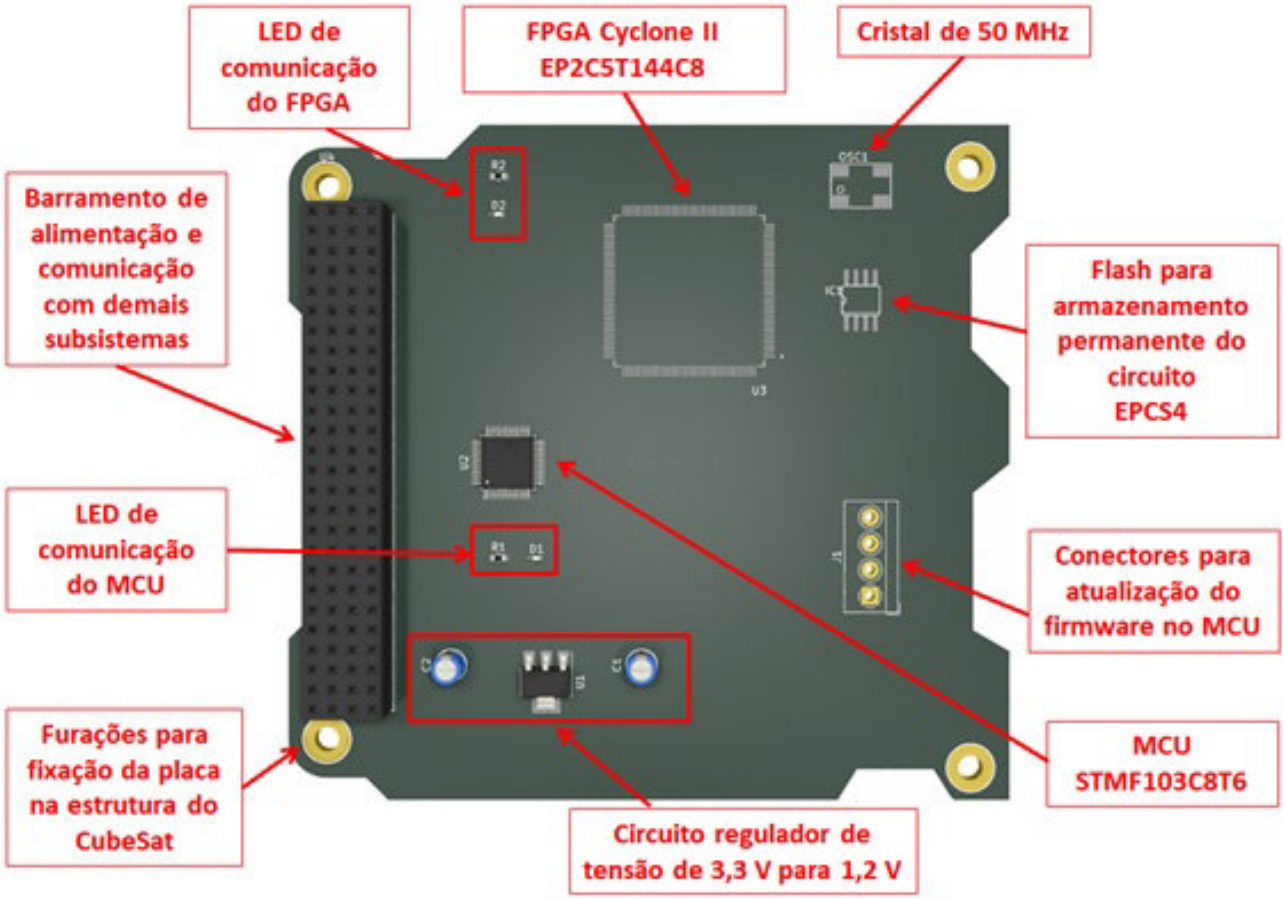


Figura 7.10: Placa de prototipagem confeccionada em funcionamento

8 Discussão

Dada a importância dos filtros adaptativos no processamento de sinais e aprendizado de máquinas, especificamente analisar e desenvolver uma modelagem matemática para o desajuste é fundamental para o entendimento do comportamento desses filtros. O método empregado no Capítulo 3, consiste numa análise de performance, onde analisamos a covariância de uma família de algoritmos adaptativos, baseada em funções não lineares, pares, contínuas, as quais admitem expansão em série de Taylor, como critério aplicado sobre o erro, aplicamos uma série de pressupostos e teoremas para derivar a covariância da função, chegando a uma expressão analítica para o desajuste dada pela Equação 3.24.

No Capítulo 5, analisamos em precisão finita a aplicação do mesmo conjunto de funções utilizadas no Capítulo 3. Realizamos uma análise matemática para descrever as condições de convergência dos algoritmos adaptativos e deduzimos as equações para mensurar a constante de tempo e o desajuste dessa família de algoritmos em sua forma quantizada conforme as Equações 5.28 e 5.37.

As expressões analíticas encontradas na análise de precisão finita para a constante de tempo, mostram que a velocidade de convergência do algoritmo é influenciada pelo sinal de entrada, pela variância do ruído adicionado ao sinal de entrada, pela variância do ruído de quantização do erro e pelo passo de aprendizagem. Quando consideramos uma Relação Sinal Ruído (SNR) alta, tanto para o ruído do sinal de entrada quanto para o ruído de quantização do erro, obteremos uma expressão que é controlada pelo sinal de entrada e passo de adaptação, tal como encontrada no [36], e se considerarmos que o sinal de entrada sofrerá quantização existe a possibilidade de uma modificação na matriz de autocorrelação do sinal de entrada do filtro adaptativo o que pode resultar em uma modificação da velocidade de convergência conforme a mudança do espalhamento dos autovalores dessa matriz.

Na análise da convergência no estado estacionário em precisão finita, observamos que o nível para o desajuste pode ser influído pela quantização do erro e dos vetores de pesos se supormos que as variâncias destes são nulas, o ruído de quantização desaparece, direcionando ao desajuste proposto na Equação 3.24 deste trabalho.

As simulações foram realizadas, dado um experimento estocástico, com os seguintes algoritmos: LMMN, WEM, SIGMOIDAL e LMF, nota-se nas Tabelas 6.1, 6.3, 6.5 e 6.7 o erro relativo $\Delta 1$ e $\Delta 2$, podemos inferir um erro relativo médio respectivamente dos algoritmos LMMN, WEM, SIGMOIDAL e LMF de $\Delta 1 = 4,11\%$, $\Delta 1 = 38,79\%$, $\Delta 1 = 31,61\%$ e $\Delta 1 = 5,73\%$ para o desajuste teórico proposto e $\Delta 2 = 96,89\%$, $\Delta 2 = 99,52\%$, $\Delta 2 = 98,99\%$ e $\Delta 2 = 76,59\%$ para o desajuste apresentado por [10]. Estes resultados indicam em cada algoritmo um erro relativo menor para o desajuste teórico proposto em relação ao apresentado por [10], comparando-os com o desajuste experimental simulado.

O desajuste teórico proposto, dadas as condições estabelecidas, mostra-se como uma alternativa superior ao apresentado por [10]. Uma diferença fundamental entre estes trabalhos, é o ponto de expansão da série de Taylor, onde [10] utilizou o ruído para expansão e calculou tanto a constante de tempo quanto o desajuste teórico. Em outro trabalho, proposto por [36], este calcula a constante tempo utilizando como ponto de expansão a saída do vetor de desvio, chegando a resultados práticos superiores ao encontrado na literatura [36], seguindo este mesmo raciocínio, prosseguimos com análise para o desajuste.

Os resultados preliminares encontrados nas tabelas 6.2, 6.4, 6.6 e 6.8 para essa família de algoritmos analisada em precisão finita indicam uma previsão considerável no entendimento do comportamento quando esses algoritmos forem implementados. Observamos nesses resultados os efeitos da quantização no passo de aprendizagem μ modificando a taxa de convergência de acordo com as Figuras 6.1, 6.2, 6.3 e 6.4. Observamos também a elevação no nível do desajuste relacionado ao número de bits implementado, mostrando uma certa estabilidade dessa família de algoritmo nessas implementações.

Estes resultados analíticos, podem representar uma generalização de muitos algoritmos adaptativos encontrados na literatura, que tem como função de custo uma não linearidade, baseadas no erro não quadrático médio. Os resultados apresentados são significativos pois permitem entender o funcionamentos desses algoritmos quando estes estiverem sendo implementados em precisão finita. Estes tipos de algoritmos tem sido sugeridos por diversos autores [45], [10] para potenciais melhorias de desempenho em diversas situações em que a distribuição de ruído pode ser não gaussiana, que são regularmente encontrados em uma série de problemas práticos de processamento de sinais.

Na Figura 7.10, pode-se observar todos os componentes envolvidos para o

desenvolvimento de uma carga útil para ser usada em um sistema de filtragem de sinal de rádio oriundo de uma estação base, por exemplo. Esta é apenas uma das aplicações disponíveis que envolvem o uso de FPGAs em nanossatélites. Outras aplicações poderiam envolver compressão de grandes volumes de dados, cálculos de parâmetros para controles de atitude, processamento digital de imagens, dentre outros.

Destaca-se, ainda, que o método proposto neste trabalho permite o desenvolvimento de sistemas embarcados para nanossatélites, na forma de cargas úteis capazes de auxiliar em processamento de dados com mais rapidez e com maior volume de dados, garantindo assim que o computador de bordo do satélite não fique sobrecarregado. A partir do diagrama esquemático apresentado, é possível fazer adequações para outros modelos de microcontrolador e adaptar a carga útil para as mais variadas aplicações e missões espaciais. Através dos testes efetuados, acredita-se que a técnica de filtragem proposta permite estimar o sinal desejado e atender às necessidades de aplicações em missões espaciais que envolvam comunicação de dados, e outros tipos de funcionalidades.

9 Conclusões

O trabalho apresenta uma nova metodologia para o desajuste de uma família de algoritmos adaptativos de gradiente estocástico de erro não médio quadrático e analisa os efeitos da precisão finita nesses tipos de algoritmos. Apresentamos uma expressão analítica para o desajuste de uma família de algoritmos adaptativos caracterizados por uma função de custo não linear agindo sobre o erro não quadrático médio e validamos através de simulações e resultados. Desenvolvemos também um circuito eletrônico contento um microcontrolador e um circuito de lógica reconfigurável baseado em FPGA, utilizando um algoritmo proposto denominado Sigmoidal de Ponto Fixo. O circuito eletrônico foi desenhado em formato PC/104 para ser implementado em nanossatélites do tipo Cubesat.

Os resultados encontrados sugerem a utilização dessa expressão analítica para o desajuste em algoritmos desse tipo. Entender o comportamento dos filtros adaptativos é relevante para a implementação e difusão desses algoritmos no mercado tecnológico e científico. Isso implica em novas alternativa para resolução de problemas de processamento de sinais e aprendizado de máquinas, encontrados em diversas áreas, como por exemplo, na medicina, militar e outras. A necessidade de implementações de novos algoritmos adaptativos em aplicações aeroespaciais, vem crescendo a medida que cresce a utilização de sistemas configuráveis em nanossatélites do tipo cubesat.

Como propostas de trabalhos futuros, pretende-se continuar avaliando o desajustes desses algoritmos para diferentes sinais de entrada e ruídos, assim como também modificar o circuito para permitir a comunicação bilateral entre o FPGA e o microcontrolador. Além disso, pretende-se realizar a confecção da placa de circuito impresso no padrão PC/104 para cubesats, a fim de testar o sistema integrado aos demais subsistemas, incluindo energia, telecomunicações e possíveis cargas úteis, tanto em bancada quanto integrados em uma estrutura de tamanho 1U ou maior.

A Apêndice

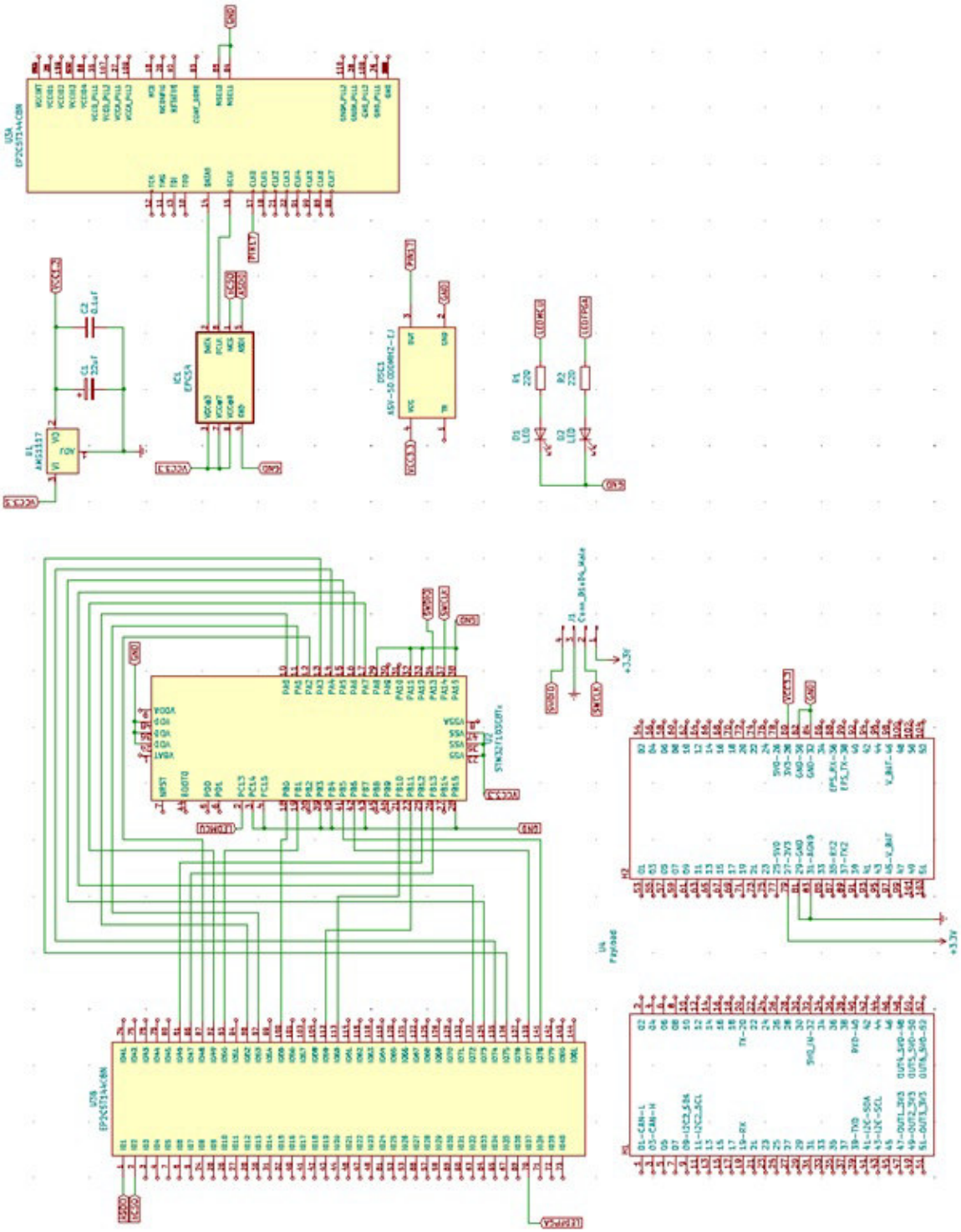
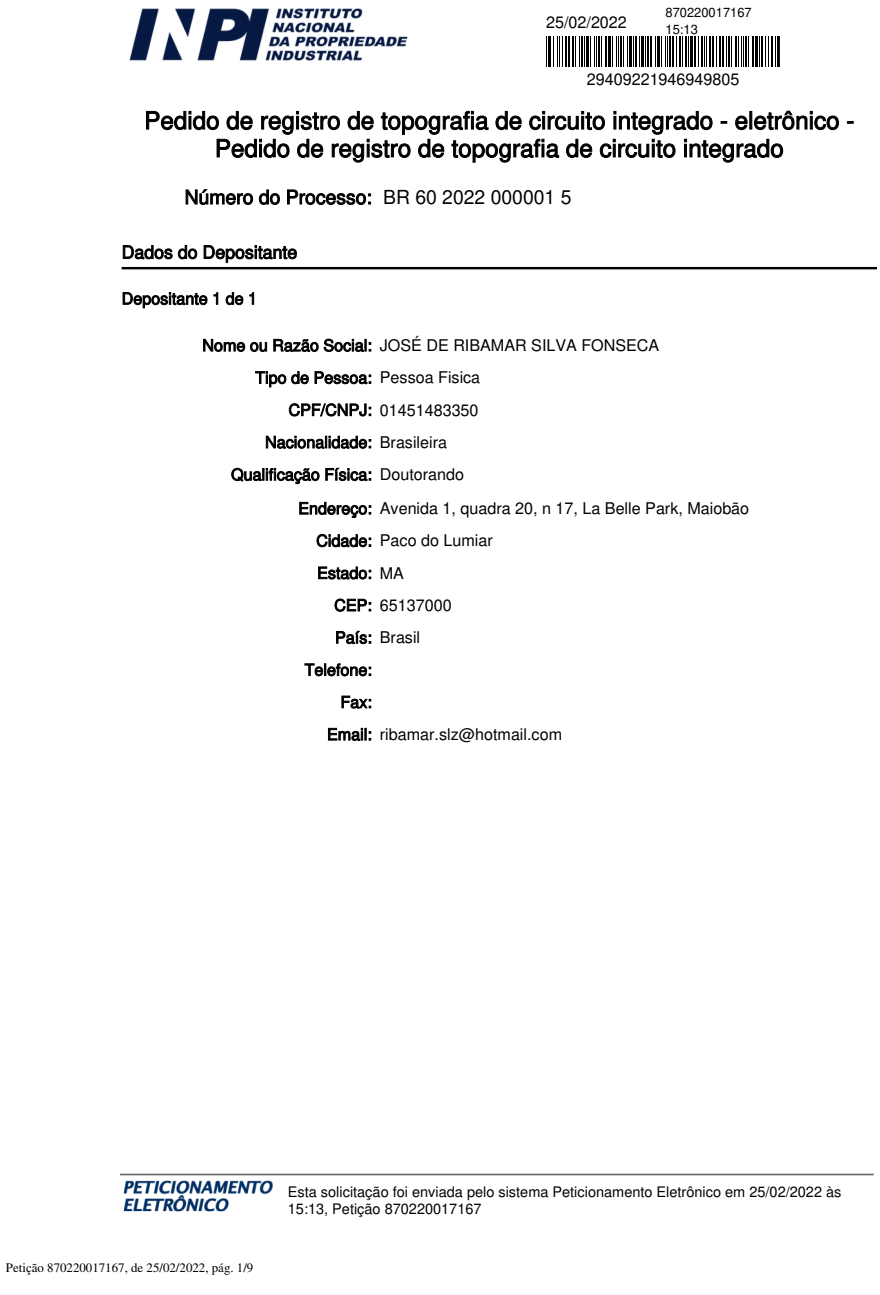


Figura A.1: Diagrama esquemático da placa de circuito impresso

B Apêndice



INSTITUTO NACIONAL DA PROPRIEDADE INDUSTRIAL

25/02/2022 870220017167 15:13

29409221946949805

Pedido de registro de topografia de circuito integrado - eletrônico - Pedido de registro de topografia de circuito integrado

Número do Processo: BR 60 2022 000001 5

Dados do Depositante

Depositante 1 de 1

Nome ou Razão Social: JOSÉ DE RIBAMAR SILVA FONSECA
Tipo de Pessoa: Pessoa Fisica
CPF/CNPJ: 01451483350
Nacionalidade: Brasileira
Qualificação Física: Doutorando
Endereço: Avenida 1, quadra 20, n 17, La Belle Park, Maiobão
Cidade: Paco do Lumiar
Estado: MA
CEP: 65137000
País: Brasil
Telefone:
Fax:
Email: ribamar.slz@hotmail.com

PETICIONAMENTO ELETRÔNICO Esta solicitação foi enviada pelo sistema Peticionamento Eletrônico em 25/02/2022 às 15:13, Petição 870220017167

Petição 870220017167, de 25/02/2022, pág. 1/9

Figura B.1: Pedido de Registro de topografia de circuito eletrônico

Referências Bibliográficas

[1] T. Y. Al-Naffouri, A. Zerguine, and M. Bettayeb. Convergence properties of mixed-norm algorithms under general error criteria. In *ISCAS'99. Proceedings of the 1999 IEEE International Symposium on Circuits and Systems VLSI (Cat. No. 99CH36349)*, volume 3, pages 211–214. IEEE, 1999.

[2] J. A. Apolinário, M. G. Siqueira, and P. S. Diniz. Fast qr algorithms based on backward prediction errors: a new implementation and its finite precision performance. *Circuits, Systems and Signal Processing*, 22(4):335–349, 2003.

[3] A. K. Barros, J. Principe, Y. Takeuchi, C. H. Sales, and N. Ohnishi. An algorithm based on the even moments of the error. pages 879–885, 2003.

[4] N. J. Bershad and J. C. M. Bermudez. A nonlinear analytical model for the quantized lms algorithm-the power-of-two step size case. *IEEE Transactions on Signal Processing*, 44(11):2895–2900, 1996.

[5] G. Bishop, G. Welch, et al. An introduction to the kalman filter. *Proc of SIGGRAPH, Course*, 8(27599-23175):41, 2001.

[6] C. Caraiscos and B. Liu. A roundoff error analysis of the lms adaptive algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(1):34–41, 1984.

[7] J. Chambers, O. Tanrikulu, and A. Constantinides. Least mean mixed-norm adaptive filtering. *Electronics letters*, 30(19):1574–1575, 1994.

[8] P. S. Diniz. *Adaptive filtering*. Springer, 2013.

[9] S. Douglas and T.-Y. Meng. A nonlinear error adaptive notch filter for separating two sinusoidal signals. pages 673–677, 1991.

[10] S. C. Douglas and T.-Y. Meng. Stochastic gradient adaptation under general error criteria. *IEEE Transactions on Signal Processing*, 42(6):1335–1351, 1994.

[11] E. Eweda. A stable normalized least mean fourth algorithm with improved transient and tracking behaviors. *IEEE Transactions on Signal Processing*, 64(18):4805–4816, 2016.

[12] B. Farhang-Boroujeny. *Adaptive filters: theory and applications*. John Wiley & Sons, 2013.

[13] M. A. Fernandez, G. Guillois, Y. Richard, J.-L. Issler, P. Lafabrie, A. Gaboriaud, D. Evans, R. Walker, O. Koudelka, P. Romano, et al. Game-changing radio communication architecture for cube/nano satellites. 2015.

[14] J. R. S. Fonseca et al. Finite precisio effect of adaptative algorithm sigmoidal. *AMC2016*.

[15] J. R. S. Fonseca et al. Analise do efeito da precisão finita no algoritmo adaptativo sigmoidal. In *Dissertação - PPGEE,BRASIL*, page 65. UFMA, 2017.

[16] J. R. S. Fonseca et al. Analise do efeito da precisão finita no algoritmo adaptativo sigmoidal. In *X-EAMC- Petropólis-RJ,BRASIL*, page 8. EAMC, 2017.

[17] J. Gibson and S. Gray. Mvse adaptive filtering subject to a constraint on mse. *IEEE transactions on circuits and systems*, 35(5):603–608, 1988.

[18] G. Gui and F. Adachi. Adaptive sparse system identification using normalized least mean fourth algorithm. *International Journal of Communication Systems*, 28(1):38–48, 2015.

[19] R. Gupta and A. O. Hero. Transient behavior of fixed point lms adaptation. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100)*, volume 1, pages 376–379. IEEE, 2000.

[20] S. S. Haykin. *Adaptive filter theory*. Pearson Education India, 2014.

[21] K. Hosseini, A. Montazeri, H. Alikhanian, and M. H. Kahaei. New classes of lms and lmf adaptive algorithms. In *2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications*, pages 1–5. IEEE, 2008.

[22] D. H. Johnson and P. Rao. On the existence of gaussian noise (signal modelling). In *[Proceedings] ICASSP 91: 1991 International Conference on Acoustics, Speech, and Signal Processing*, pages 1673–1676. IEEE, 1991.

[23] R. A. Khan, Mohd Tasleem e Shaik. Arquitetura vlsi de alto desempenho do filtro adaptável dlms para convergência rápida e baixo mse. *Transações IEEE em Circuitos e Sistemas II: Resumos Expressos*.

[24] A. Lovascio, A. D'Orazio, and V. Centonze. Design of cots-based radio-frequency receiver for cubesat applications. In *2019 IEEE 5th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, pages 399–404. IEEE, 2019.

[25] Y. R. M. Maluenda et al. Propriedades do algoritmo lms operando em precisao finita. 2005.

[26] D. G. Manolakis, V. K. Ingle, S. M. Kogon, et al. *Statistical and adaptive signal processing: spectral estimation, signal modeling, adaptive filtering, and array processing*. McGraw-Hill Boston, 2000.

[27] P. K. Meher and S. Y. Park. Area-delay-power efficient fixed-point lms adaptive filter with low adaptation-delay. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 22(2):362–371, 2013.

[28] P. Mitra, A. Gupta, G. P. Reddy, S. Srivastava, M. Tyagi, M. Sharma, S. Gadkari, P. Chaudhury, and A. V. Kumar. An environmental gamma spectrometry system with csi (tl) scintillator and fpga based mca for open field deployment. *Applied Radiation and Isotopes*, 172:109677, 2021.

[29] R. M. Narsale, D. Gawali, and A. Kulkarni. Fpga based design & implementation of low power fir filter for ecg signal processing. *International Journal of Science, Engineering and Technology Research (IJSETR)*, 3(6), 2014.

[30] W. T. Padgett and D. V. Anderson. Fixed-point signal processing. *Synthesis Lectures on Signal Processing*, 4(1):1–133, 2009.

[31] S. Peng, W. Ser, B. Chen, L. Sun, and Z. Lin. Robust constrained adaptive filtering under minimum error entropy criterion. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 65(8):1119–1123, 2018.

[32] E. Razzaghi. Design and qualification of on-board computer for aalto-1 cubesat, 2012.

[33] P. A. Regalia. Review of fundamentals of adaptative filtering. pages 77–79, 2005.

[34] A. Salman, E. Shufan, I. Lapidot, L. Tsror, R. Moreh, S. Mordechai, and M. Huleihel. Assignment of colletotrichum coccodes isolates into vegetative compatibility groups using infrared spectroscopy: a step towards practical application. *Analyst*, 140(9):3098–3106, 2015.

[35] E. Santana, A. K. Barros, R. Freire, et al. An adaptive algorithm based on the sigmoidal function. In *2006 Ninth Brazilian Symposium on Neural Networks (SBRN'06)*, pages 1–5. IEEE, 2006.

[36] E. Santana, A. K. Barros, and R. C. Freire. On the time constant under general error criterion. *IEEE Signal Processing Letters*, 14(8):533–536, 2007.

[37] A. H. Sayed. *Fundamentals of adaptive filtering*. John Wiley & Sons, 2013.

[38] R. Sharma, W. A. Sethares, and J. A. Bucklew. Asymptotic analysis of stochastic gradient-based adaptive filtering algorithms with general cost functions. *IEEE Transactions on Signal Processing*, 44(9):2186–2194, 1996.

[39] L. Stras, D. D. Kekez, G. J. Wells, T. Jeans, R. E. Zee, F. Pranajaya, and D. Foisy. The design and operation of the canadian advanced nanospace experiment (canx-1). In *Proc. AMSAT-NA 21st Space Symposium, Toronto, Canada*, pages 150–160. Citeseer, 2003.

[40] S. U. Talha, H. Hussain, and M. Moinuddin. An efficient regularized nlmf algorithm. In *2016 6th International Conference on Intelligent and Advanced Systems (ICIAS)*, pages 1–6. IEEE, 2016.

[41] R. J. Tocci, N. S. Widmer, and G. L. MOSS. Sistemas digitais: Princípios e aplicações–11ª. *Edição. Rio de Janeiro: MZEditora*, 2007.

[42] M. Verhaegen. Round-off error propagation in four generally-applicable, recursive, least-squares estimation schemes. *Automatica*, 25(3):437–444, 1989.

[43] T. Villela, C. A. Costa, A. M. Brandão, F. T. Bueno, and R. Leonardi. Towards the thousandth cubesat: A statistical overview. *International Journal of Aerospace Engineering*, 2019, 2019.

[44] T. S. Wada and B.-H. Juang. Enhancement of residual echo for robust acoustic echo cancellation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):175–189, 2012.

[45] E. Walach and B. Widrow. The least mean fourth (lmf) adaptive algorithm and its family. *IEEE transactions on Information Theory*, 30(2):275–283, 1984.

[46] B. Widrow and D. Stearns, S. *Adaptive signal processing*, volume 1. Prentice Hall, Englewood Cliffs, 1985.

[47] S. Wu, W. Chen, C. Cao, C. Zhang, and Z. Mu. A multiple-cubesat constellation for integrated earth observation and marine/air traffic monitoring. *Advances in Space Research*, 67(11):3712–3724, 2021.

[48] T. Yuan and G. Wang. Simplified minimum fourth-order moment haze channel equalization algorithm. In *2019 IEEE 11th International Conference on Communication Software and Networks (ICCSN)*, pages 275–279. IEEE, 2019.

[49] S. Zhang, W. X. Zheng, J. Zhang, and H. Han. A family of robust m-shaped error weighted least mean square algorithms: performance analysis and echo cancellation application. *IEEE Access*, 5:14716–14727, 2017.