ALEXANDER ALVAREZ ROSARIO

SISTEMA PARA MONITORAMENTO E ANÁLISE DE
PAISAGENS ACÚSTICAS SUBMARINAS

São Paulo

2015

ALEXANDER ALVAREZ ROSARIO

SISTEMA PARA MONITORAMENTO E ANÁLISE DE
PAISAGENS ACÚSTICAS SUBMARINAS

Tese apresentada à Escola Politécnica da
Universidade de São Paulo para obtenção
do título de Doutor em Engenharia em
Ciências.

Área de Concentração:

Engenharia Mecânica, Projeto e
Fabricação.

Orientador: Prof. Livre-Docente
Dr. Linilson R. Padovese

São Paulo

2015

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, _____ de _____ de _____

Assinatura do autor: _____

Assinatura do orientador: _____

Catálogo-na-publicação

Rosario, Alexander Alvarez

SISTEMA PARA MONITORAMENTO DE PAISAGENS ACÚSTICAS
SUBMARINAS / A. A. Rosario -- versão corr. -- São Paulo, 2015.
103 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo.
Departamento de Engenharia Mecânica.

1.Monitoramento acústico passivo submarino 2.Instrumentação de
acústica aubmarina 3.Detecção de eventos bioacústicos submarinos
4.Processamento de sinais acústicos submarinos I.Universidade de São
Paulo. Escola Politécnica. Departamento de Engenharia Mecânica II.t.

A meus pais Gonzalo (REP) e Jiorman

A meu querido filho, Nicolas, motivo de muitas alegrias e retos

A meus irmãos Darleds, Carlos, John e Jeimmy

A Jaqueline, o meu amor e inspiração

Agradecimentos

A DEUS, por me permitir viver e realizar este outro dos seus planos.

Ao Prof. Dr. Linilson R. Padovese, pela oportunidade oferecida, pela amizade e pelo apoio e pela confiança em mim depositada.

Ao CNPq, pela concessão da bolsa e pelo apoio financeiro, o que possibilitou a minha permanência no Brasil e dessa forma, conseguir realizar meus estudos de mestrado.

Aos colegas do Laboratório LADIN, especialmente ao Dr. Manuel Caldas pelas sugestões e a grande amizade.

A os meus amigos Ruben, Victor, Mona, Mariana, Alexandre, e todos aqueles com quem compartilhei e fizeram da minha permanência no Brasil uma experiência sumamente agradável.

Resumo

O Monitoramento acústico passivo (PAM) submarino refere-se ao uso de sistemas de escuta e gravação subaquática, com o intuito de detectar, monitorar e identificar fontes sonoras através das ondas de pressão que elas produzem. Se diz que é passivo já que tais sistemas unicamente ouvem, sem perturbam o meio ambiente acústico existente, diferentemente de ativos, como os sonares.

O PAM submarino tem diversas áreas de aplicação, como em sistemas de vigilância militar, seguridade portuária, monitoramento ambiental, desenvolvimento de índices de densidade populacional de espécies, identificação de espécies, etc.

Tecnologia nacional nesta área é praticamente inexistente apesar da sua importância. Neste contexto, o presente trabalho visa contribuir com o desenvolvimento de tecnologia nacional no tema através da concepção, construção e operação de equipamento autônomo de PAM e de métodos de processamento de sinais para detecção automatizada de eventos acústicos submarinos.

Foi desenvolvido um equipamento, nomeado OceanPod, que possui características como baixo custo de fabricação, flexibilidade e facilidade de configuração e uso, voltado para a pesquisa científica, industrial e para controle ambiental. Vários protótipos desse equipamento foram construídos e utilizados em missões no mar.

Essas jornadas de monitoramento permitiram iniciar a criação de um banco de dados acústico, o qual permitiu fornecer a matéria prima para o teste de detectores de eventos acústicos automatizados e em tempo real. Adicionalmente também é proposto um novo método de detecção-identificação de eventos acústicos, baseado em análise estatística da representação tempo-frequência dos sinais acústicos. Este novo método foi testado na detecção de cetáceos, presentes no banco de dados gerado pelas missões de monitoramento.

Palavras – chave: Monitoramento acústico passivo submarino. Detecção de eventos bioacústicos submarinos. Processamento de sinais acústicos submarinos.

Abstract

Passive acoustic monitoring (PAM) refers to the use of systems to listen and record underwater soundscape, in order to detect, track and identify sound sources through the pressure waves that they produce. It is said to be passive as these systems only hear, not put noise in the environment, such as sonars.

Underwater PAM has various application areas, such as military surveillance systems, port security, environmental monitoring, development of population density rates of species, species identification, etc.

National technology in the field is practically nonexistent despite its importance. In this context, this paper aims to contribute to the national technology development in the field by designing, building, and operating a self-contained PAM equipment, also developing signal-processing methods for automated detection of underwater acoustic events.

A device, named "OceanPod" which has characteristics such as low manufacturing cost, flexibility and ease of setup and use, intended for scientific, industrial research and environmental control was developed. Several prototypes of the equipment were built and used in several missions at seawaters.

These missions monitoring, enabled start creating an acoustic database, which provided the raw material for the automated acoustic events detectors and real-time test. Additionally, it is also proposed a new method of detecting, identifying sound events, based on statistical analysis of the time-frequency representation of the acoustic signals. This new method has been tested in the detection of cetaceans present in the database generated by missions monitoring.

Key-words: Underwater passive acoustic monitoring. Underwater bio-acoustic event detection. Underwater acoustics signal processing.

Lista de Figuras

2.1.	Esquema dos componentes do OceanPod. (a) baterias. (b) Conversor DC-DC. (c) RPI modelo A (RPI). (d) MicroSDx. (e) Relógio de tempo real (RTC). (f) Interface de áudio. (G) Condicionador. (h) Hidrofone. . .	20
2.2.	Representação esquemática do invólucro do OceanPod. (1) tampa inferior. (2) tampa superior. (3) Anéis de vedação – Oring. (4) Tubo de PVC. (5) parafusos para fixação de gaiola de proteção. (6) gaiola de proteção hidrofone.	21
2.3.	Esquema de arranjo das baterias na unidade de PAM. (a) pack de pilhas em série. (b) conjunto de pack de pilhas em paralelo e (c) conversor DC-DC.	22
2.4.	Unidade de processamento. (a) Placa RPI. (b) Relógio RTC. (c) Cartão Micro-SDx	23
2.5.	(a) Placa da interface de áudio, (b) Condicionador de sinal, (c) Hidrofone piezoelétrico.	23
2.6.	Fonte: imagem de UCA222 adaptada de (INSTRUCTABLES, 2015). Condicionador e hidrofone: o autor (2015)	23
2.7.	Curva de resposta em frequência experimental para a placa de condicionamento de sinal do hidrofone	24
2.8.	Algoritmo de inicialização do SO no Oceanpod	26
2.9.	Algoritmo de sincronização de hora e data do SO no Oceanpod	27
2.10.	Algoritmo de aquisição de dados no Oceanpod	28
2.11.	Locais de instalação dos OceanPod. (a) Parnaíba – PI. (b) Ilhéus - BA. (c) Ilha das Palmas – SP. (d) Laje de Santos – SP	29
2.12.	OceanPod com capacidade para 40 pilhas alcalinas tipo D	30
2.13.	Vaso de Pressão usado para teste de estanqueidade	31
2.14.	Consumo de energia no OceanPod em modo de gravação para diferentes frequências de amostragem	32
2.15.	Consumo de energia no OceanPod em modo de gravação para diferentes frequências de amostragem (incrementos percentuais associados com pós-processamento do sinal – “downsampling”)	33
2.16.	Curvas de consumo energético do OceanPod e de tensão e corrente fornecidas pelo conjunto de pilhas no Teste 3)	33
2.17.	Espectrogramas para diversos tipos de sinal. (a) Tráfego marinho no porto de Santos – Ilha das Palmas. (b) Vocalizações de golfinhos estuarinos no Litoral de Fortaleza – Pecém. (c) Vocalizações de Baleia Jubarte – Ilhéus.)	35

3.1. Configuração dos dispositivos no experimento. (a) RPI, (b) Roteador, (c) Computador.	42
3.2. Fonte: o autor (2015)	42
3.3. Diagrama de fluxo do algoritmo de avaliação dos métodos de calcular envelopes (funções) para todos os casos de Fs, W e Sobreposição.	44
3.4. Fonte: o autor (2015)	44
3.5. Segmento de áudio contendo assobios de boto cinza	47
3.6. Fonte: o autor (2015)	47
3.7. Envelopes obtidos com o método da média móvel para diferentes tamanhos de janela W	49
3.8. Fonte: o autor (2015)	49
3.9. Envelopes obtidos com o método rms para diferentes tamanhos de janela W e sobreposição de 75%	50
3.10. Fonte: o autor (2015)	50
4.1. Locais de instalação das unidades de Oceanpod no litoral de Ilhéus (BA)	57
4.3. Curvas de resposta em frequência de filtros	59
4.4. Boxplot para dados com distribuição normal	61
4.5. Ilustração do box-plot ajustado para dados com assimetria. $d1$ e $d2$ são os bigodes quando há suposição de normalidade nos dados. $s1$ e $s2$ são os bigodes de dados com assimetria. Pode-se observar que, de acordo com a metodologia selecionada de cálculo, o ponto $x2$ pode ser identificado ou não como sendo outlier	62
4.6. Pseudocódigo para o cálculo de espectrogramas e curvas de SPL para grandes volumes de dados	64
4.7. Tela da interface de usuário para cálculo do SPL e Espectrograma para grandes volumes de dados	65
4.8. Espectrograma de larga duração. (a) Vocalizações de baleias jubarte. (b) Ruído de embarcações com harmônicos. (c) Ruído de equipamento rotativo de alta energia não identificado. (d) Ruído de embarcações distantes poluindo a banda de frequência das vocalizações da baleia jubarte	65
4.9. Curva de SPL para dados de larga duração	66
4.10. Representação esquemática do algoritmo EVD (a)-(g), e denoising (h)-(k)	67
4.11. Cenários de teste do Algoritmo de detecção EVD	70
4.13. (a) Espectrograma de segmento de sinal com presença de vocalizações. (b) Função de detecção (linha vermelha) e threshold calculado por média móvel (linha azul). (c) Eventos detectados (linha azul)	71
4.14. Espectro de potência para o segmento de áudio analisado na Figura 4.13	73
4.15. Espectro de potência do sinal analisado antes (linha preta) e depois de filtragem (linha vermelha)	73
4.16. (a) Espectrograma do sinal após filtragem com equalização. (b) Função de detecção e limiar do sinal filtrado. (c) Eventos detectados (linha azul)	74
4.18. (a) Espectrograma de uma vocalização de baleia jubarte. (b) contorno extraído usando EVD e filtro de medianas 2D.	77

4.19. Boxplot (a) e densidade de probabilidade (b) do espectrograma para o evento acústico da Figura 4.18. Em amarelo os valores extremos calculados 78

Lista de Tabelas

2.1.	Desenho experimental para teste de consumo do Oceanpod	28
2.2.	Parâmetros usados em teste com baterias	29
2.3.	Locais de instalação dos Oceanpod. Unidades e configurações	30
2.4.	Resumo dados gravados nas diferentes instalações do Oceanpod	34
3.1.	Resultados da primeira rodada com arquivo de áudio de $Fs = 2kHz$. . .	44
3.2.	Resultados da primeira rodada com arquivo de áudio de $Fs = 2kHz$. . .	44
3.3.	Tempos de execução com modelo de Média móvel para cálculo do envelope	45
3.4.	Tempos de execução com modelo rms de janelas deslizantes e sobreposição 0.75	46
3.5.	Tempos de execução com modelo rms de janelas deslizantes e sobreposição 0.75	46
3.6.	Tempos de execução com modelo rms de janelas deslizantes e sobreposição 0.25	46
3.7.	Tempos de execução com modelo rms de janelas deslizantes e sem sobre- posição	47
4.1.	Bando de dados de PAM submarino usado nos testes	58
4.2.	Resultados comparativos do detector EVD e o EDet (Mellinger, 2002) . .	75
4.3.	Eventos detectados com sucesso em cada um dos métodos, comparados isoladamente e em conjunto	76

LISTA DE ABREVIATURAS E SIGLAS

ANN Rede Neural Artificial (Artificial Neural Networks)
CART Análise de Árvores de classificação e regressão (Classification and Regression analysis)
CLUSTER Análise de Agrupamentos
DEP Densidade Espectral de Potência
DFA Função Discriminante (Discriminant Function Analysis)
DN Diâmetro Nominal
DSP Processamento Digital de Sinais (Digital Signal Processing)
EVD Detecção de valores extremos (“Extreme Values Detection”)
FFT Transformada rápida de Fourier
Fs Frequência de Amostragem
GMMs Modelos de Mistura Gaussianas (Gaussian Mixture Models)
GPIO General Port Input Output
HMM Modelos ocultos de Markov (Hidden Markov Models)
IQR Distância interquartil
LDA Análise Discriminante Linear (Linear Discriminant Analysis)
LTSA Espectrogramas médios de longo prazo ou espectrogramas médios a longo tempo (Long Term Spectrogram Average)
LTSPL Nível de Pressão Sonora para Longos Períodos de tempo (Long Time Sound Pressure Level).
PAM Monitoramento Acústico Passivo (Passive Acoustic Monitoring)
PCA Análise de Componentes Principais (Principal Component Analysis)
QDA Análise Discriminante Quadrático (Quadratic Discriminant Analysis)
RPI Raspberry PI
SO Sistema Operacional
SPL Nível de Pressão Sonora (Sound Pressure Level)

Sumário

1. INTRODUÇÃO	15
1.1. OBJETIVOS ESPECÍFICOS	16
1.2. ORGANIZAÇÃO DO TRABALHO	17
2. UNIDADE DE GRAVAÇÃO AUTÔNOMA OCEANPOD	18
2.1. MATERIAIS	20
2.1.1. Involucro ou case	21
2.1.2. Fornecimento de energia	21
2.1.3. Eletrônica de controle	22
2.1.4. Aquisição do sinal	22
2.1.5. Software de controle do OceanPod	25
2.1.6. Gestão de arquivos e aquisição de dados	26
2.1.7. Metodologia de teste de consumo	28
2.1.8. Teste de Gravador de dados no mar	29
2.2. RESULTADOS E DISCUSSÃO	30
2.2.1. Unidades fabricadas	30
2.2.2. Teste de estanqueidade	31
2.2.3. Teste de consumo	31
2.2.4. Teste de aquisição de dados no mar	34
2.3. CONCLUSÕES	35
3. DETECÇÃO EM TEMPO REAL	37
3.1. MÉTODOS DE DETECÇÃO	37
3.2. MÉTODOS DE CÁLCULO DO ENVELOPE DO SINAL	38
3.3. PLANEJAMENTO DO EXPERIMENTO COMPUTACIONAL	41
3.3.1. Hardware	41
3.3.2. Software	42
3.3.3. Planejamento experimental	43
3.4. RESULTADOS E DISCUSSÃO	43
3.4.1. Resultados média móvel	45
3.4.2. Resultados rms com janelas deslizantes	45
3.4.3. Desempenho de envelopes como funções de detecção	47
3.5. CONCLUSÕES	48
4. DESENVOLVIMENTO DE MÉTODOS DE ANÁLISE	51
4.1. DETETORES EM TEMPO REAL	51

4.2. DETECTORES BASEADOS EM ENERGIA E AMPLITUDE DO SINAL	52
4.3. DETETORES BASEADOS EM CORRELAÇÃO	53
4.4. DETECTORES BASEADOS EM EXTRAÇÃO DE CARACTERÍSTICAS	54
4.5. SISTEMAS DE DETECÇÃO DE EVENTOS BIO-ACÚSTICOS	56
4.6. BANCO DE DADOS	57
4.7. EMBASAMENTO TEÓRICO	58
4.7.1. Espectrograma – representação tempo-frequência de um sinal . . .	58
4.7.2. Filtragem do sinal	58
4.7.3. Outliers e Valores Extremos	60
4.8. PROCESSAMENTO E VISUALIZAÇÃO DE GRANDES VOLUMES DE DADOS	62
4.9. ALGORITMO DE DETECÇÃO BASEADO EM EVENTOS EXTREMOS [EVD]	66
4.10. Testes e resultados	69
4.10.1. Resultados e discussão detecção cenário 1	70
4.10.2. Resultados e discussão detecção cenário 2	72
4.10.3. Desempenho do Algoritmo de detecção EVD	75
4.11. REMOÇÃO DE RUÍDO EM ESPECTROGRAMA DE EVENTOS ACÚSTICOS SEGMENTADOS	76
4.12. CONCLUSÕES	78
5. CONCLUSÕES GERAIS	80
6. TRABALHOS FUTUROS	81
A. Publicações e trabalhos apresentados em congresso internacional	87
B. Envelopes Python	88
C. Funções Python para avaliar envelopes	90
D. Algoritmo EVD	94
E. SPL para grandes volumes de dados	99
F. Espectrograma para grandes volumes de dados	101

1. INTRODUÇÃO

A extensa área marítima brasileira é cenário de atividades industriais, tais como a exploração petrolífera, exploração portuária, navegação, pesca, mineração, dentre outras. É um fato que em decorrência dessas atividades, há incrementos nos níveis de poluição sonora no ambiente acústico submarino que afetam, direta ou indiretamente, a integridade fisiológica e/ou comportamental das espécies que frequentam as áreas de influência dessas atividades, como, p.ex., os cetáceos.

De um modo mais abrangente, o Brasil tem dado avanços importantes para a gestão de políticas direcionadas à preservação, recuperação e aproveitamento dos recursos naturais, ocupação de áreas e espaços preservados, uso dos recursos hídricos, fiscalização, financiamentos e pesquisas. Na consecução desses objetivos estão envolvidos todos os órgãos que compõem o Sistema Nacional do Meio Ambiente (Sisnama).

Além das instituições ligadas direta ou indiretamente com os órgãos públicos, existem iniciativas por parte de ONGs e outros grupos que realizam atividades de pesquisa que contribuem com a geração de conhecimento por meio de publicações e/ou criação de bancos de dados das espécies estudadas, informações estas de vital importância dadas às dificuldades próprias (logísticas, técnicas e tecnológicas) de realizar pesquisa ambientes marinhos.

Na revisão bibliográfica feita, encontraram-se pesquisas em linhas tais como estimativas de abundância espécies, estudos de densidade populacional, monitoramento de cetáceos, monitoramento de baleias por satélite, projeto golfinho flipper (*Tursiops truncatus*), para citar alguns exemplos. Para a localização e identificação de espécies marinhas o método de inspeção visual é ainda um dos mais usados, método que contém as limitações próprias da visão humana como instrumento de detecção, identificação de indivíduos e com a restrição a jornadas diurnas e com condições de visibilidade adequadas.

Como alternativa à inspeção visual, a técnica mais empregada para detecção e localização de espécies marinhas e fluviais é o monitoramento acústico submarino. Os indivíduos são caracterizados pelos “ruídos” ou sons por eles produzidos. No caso dos cetáceos por exemplo, alguns desses sons são chamados de “cantos”, “assobios” e “clicks”. A grande vantagem da análise através de sinais acústicos radica em aspectos próprios do som tais como as distâncias percorridas (dependentes da frequência e intensidade), que podem ir desde algumas centenas de metros a vários de quilômetros da fonte de emissão,

e da não dependência de condições de visibilidade ou climáticas que não perturbem o meio acústico.

No Brasil, o uso de tecnologias PAM tem sido pouco explorada, sobretudo em decorrência na inexistência de tecnologia nacional e dos altos custos dos equipamentos importados. Entende-se por tecnologia tanto equipamentos adequados para monitoramento (embarcado ou autônomos) com também métodos de detecção, identificação e geração de grandes bancos de dados implementados em softwares de uso minimamente amigável.

Embora seja uma área com grandes avanços no âmbito internacional, o monitoramento e estudo de paisagens acústicas no Brasil ainda é uma área pouco desenvolvida e somente graças às exigências e regulações do Ibama vem ganhando alguma importância.

Do ponto de vista da pesquisa, ainda que alguns trabalhos de bio-acústica tenham sido realizados no Brasil, o tema pode ser considerado como praticamente inexplorado. Eles estão mais concentrados no estudo da ecologia de aves, anuros, mamíferos terrestres e, em menor extensão, em mamíferos aquáticos.

No presente trabalho apresenta-se o desenvolvimento de um equipamento de monitoramento acústico passivo e de algoritmos de detecção e localização de eventos em sinais acústicos submarinos, principalmente embarcações e mamíferos marinhos que frequentam o litoral brasileiro.

1.1. OBJETIVOS ESPECÍFICOS

- Desenvolver equipamento de monitoramento acústico submarino
- Instalar e operar unidades de monitoramento acústico submarino em vários locais do litoral brasileiro
- Criar um banco de dados de sinais acústicos submarinos
- Desenvolver algoritmos de detecção de eventos acústicos automatizados, para grandes bancos de dados
- Implementar os métodos e algoritmos desenvolvidos num sistema integrado e baseado em software de uso livre.

1.2. ORGANIZAÇÃO DO TRABALHO

Este documento está organizado como descrito a seguir:

- No Capítulo 2 são apresentados a revisão bibliográfica de equipamentos autônomos de PAM, o projeto de gravador autônomo para monitoramento acústico passivo e alguns resultados de teste no mar.
- No capítulo 3 apresentam-se a revisão bibliográfica e propõe-se o conceito de detector de eventos acústicos (embarcados ou offline) para análise em tempo real. Também são apresentados testes de desempenho, tanto de computação como de detecção.
- No capítulo 4 é apresentado um novo algoritmo para detecção de eventos em sinais acústicos através de uma abordagem estatística utilizando o conceito de valores extremos. O mesmo conceito é analisado como método de remoção de ruído ou "*denoising*" em espectrogramas de eventos acústicos. Também são apresentados testes de desempenho.
- No capítulo 5 são apresentadas as conclusões gerais do trabalho

2. UNIDADE DE GRAVAÇÃO AUTÔNOMA OCEANPOD

Equipamentos de PAM são equipamentos eletrônicos desenvolvidos para digitalizar e armazenar sinais de tipo acústico em ambientes subaquáticos, compostos basicamente por um hidrofone, um conjunto de partes eletrônicas e algum tipo de interface homem-maquina (Zimmer, 2011) que permita sua configuração.

Estes equipamentos, dependendo da fonte de energia disponível, podem ser classificados como unidades autônomas, quando a energia é limitada e fornecida por baterias internas, ou unidades de uso embarcada, quando há fornecimento permanente de energia, seja da rede elétrica ou por baterias associadas a painéis solares ou outras fontes alternativas.

Analisando os equipamentos de PAM do ponto de vista da transmissão da informação, equipamentos permanentes podem oferecer dados em tempo real, já os de tipo autônoma não. Nessas últimas, cada unidade deve ser inicialmente recuperada, os dados extraídos e processados posteriormente. Com isso, a escolha da utilização de um ou outro equipamento de PAM vai depender das características de cada missão (localização geográfica, facilidade de acesso ao local, logística de instalação) e das facilidades em infraestrutura para instalação.

Além desta classificação prévia, unidades de PAM podem também ser classificadas do ponto de vista da tecnologia envolvida na sua concepção. Por uma lado estão os equipamentos baseados em micro-controladores que têm como principal vantagem a economia de energia mas possuem possibilidades de configuração mais limitadas e até protegidas pelo fabricante e, por outro lado há equipamentos baseados em micro-processadores que possibilitam um maior poder de processamento, facilidade de configuração e de uso de periféricos, embora tenham um consumo maior de energia.

Sousa-Lima (Sousa-Lima et al., 2013) identificaram mais de 30 equipamentos de PAM desenvolvidos por instituições de pesquisa, laboratórios de diferentes universidades e empresas especializadas na área. Com exceção dos equipamentos OceanPod e OceanBase, unidades de gravação autônoma e permanente, respectivamente, desenvolvidas no Laboratório de Dinâmica e Instrumentação – Ladin – Escola Politécnica – Universidade de São Paulo, todos os equipamentos citados no trabalho anterior são de fabricação internacional.

Existem várias iniciativas por equipes de pesquisa no esforço de desenvolver equipamentos autônomos de PAM, em (Wang et al., 2009) é usado o micro controlador MSP430 MCU (National Instruments) para digitalizar o sinal de áudio e armazenar os dados diretamente num (ou vários) cartões SD. Nestes casos é possível adquirir sinais com uma frequência de amostragem de até 200kHz, embora o nível de quantização (12 bits) não seja o ideal para monitoramento de eventos acústicos, onde o padrão é 16 bits mínimo.

Outros exemplos de equipamentos usando microcontroladores são o Acousonde, (Au et al., 2000; Burgess, 2000; Chen et al., 1998) que utiliza o micro-controlador Tattletale Model 7 (ONSET-HOBO Dataloggers), e o ARP (Acoustic Recording Package) (Wiggins, 2003), que usa o micro-processador Ocean Sensors OS500 Datalogger da empresa OceanSensors.

Ming-Hao (Chiu et al., 2007) usaram um computador industrial PC-104 com processador Intel-Celeron com uma placa AD de 12 bits para digitalização do áudio. O nível de quantização do sinal e a baixa autonomia (4,5 horas com 35,4 AH a 16V) fazem que essa proposta seja inviável, sobretudo em relação aos avanços das plataformas computacionais disponíveis atualmente. Há outros equipamentos de PAM que usa o mesmo computador PC-104 como a AUAR - Autonomous Underwater Acoustic Recorder e a versão T-AUAR que oferece transmissão de dados via rádio (S.V. Borisov, 2008), e a A-PANDA - Advances Pop-up Ambient Noise Data Acquisition (Koay et al., 2006).

Recentemente, a aparição de vários tipos de computador de placa única, baseados em tecnologia código aberto (open-source) trouxe algumas vantagens para seu uso tanto em educação quanto em pesquisa. Dentre das vantagens encontram-se o baixo custo (entre 50 e 100 USD no mercado internacional), uma crescente comunidade dedicada ao desenvolvimento de vários tipos de projetos embarcados e a contínua aparição de dispositivos de entrada e saída planejados para funcionar com essa plataforma.

No mercado internacional existem diferentes equipamentos comerciais, geralmente de alto custo e nem sempre estão disponíveis no mercado nacional. Já equipamentos baseados em micro-computadores (computadores industriais, computadores pessoais ou computadores de placa única), mesmo que com um consumo maior de energia, oferecem maior flexibilidade de programação de ciclos de gravação e de frequências de amostragem.

Apresenta-se neste capítulo o desenvolvimento e resultados realizados no projeto, fabricação e implementação de unidade de gravação autônoma, nomeada ‘OceanPod’. Trata-se de um desenvolvimento único no âmbito nacional, já que não se tem notícias de algo similar, desenvolvido e empregado no país. Informações adicionais podem ser obtidas em (Caldas-Morgan et al., 2015).

Este equipamento foi concebido para suprir a necessidade de aquisição de sinais acústicos submarinos de maneira autônoma por longos períodos de tempo. As características gerais que procura atender são:

- Autonomia mínima de 1 mês, 24hrs por dia,
- Banda de frequência de áudio de 10Hz a 16kHz;
- Baixo custo
- 16 bits de quantização do áudio

Essas características possibilitam um tempo de autonomia que minimiza custos de logística de instalação e, ao mesmo tempo, garante uma banda de frequência de sinal que abrange a maior parte dos eventos antrópicos e biológicos encontrados na costa brasileira.

A questão de um projeto de baixo custo visa dispor de um equipamento que esteja dentro da realizada orçamentária da maior parte das equipes de pesquisa do sistema de ensino e pesquisado do país.

2.1. MATERIAIS

Na Figura 2.1 é apresentado de forma esquemática os componentes de um OceanPod projetado e implementado no presente trabalho. Como definido no início do capítulo, essas unidades são compostas por um hidrofone (Figura 2.1-h), um conjunto de partes eletrônicas (Figura 2.1-b,c,d,e,f,g), e algum tipo de interface homem-maquina, que no OceanPod é fornecida pelo computador de placa única (Figura 2.1-c) através de conexão de rede, ou um teclado e tela.

Figura 2.1.: Esquema dos componentes do OceanPod. (a) baterias. (b) Conversor DC-DC. (c) RPI modelo A (RPI). (d) MicroSDx. (e) Relógio de tempo real (RTC). (f) Interface de áudio. (G) Condicionador. (h) Hidrofone.

Fonte: o autor (2015)

Apresenta-se a seguir a metodologia de desenvolvimento, a descrição dos componentes e resultados de operação.

2.1.1. Involucro ou case

Todos esses componentes, com exceção do hidrofone, encontram-se alojados dentro de um recipiente estanque cilíndrico. O equipamento foi instalado e recuperado através de mergulho autônomo, cuja profundidade máxima de segurança recomendada é de 30m.

Portanto projetou-se um recipiente de PVC de 4" de diâmetro e 9mm de espessura e 52.5mm de comprimento (Figura 2.2), rosqueado nos extremos e com tampas feitas no mesmo tipo de material. Todas as peças rosqueadas possuem um anel de vedação *o-ring* que garante a estanqueidade. Pode-se apreciar na mesma figura que as únicas peças expostas à água são o hidrofone e sua gaiola de proteção. A estanqueidade, conforme descrito no item 2.2.2, foi verificada até 7 bar.

Figura 2.2.: Representação esquemática do invólucro do OceanPod. (1) tampa inferior. (2) tampa superior. (3) Anéis de vedação – Oring. (4) Tubo de PVC. (5) parafusos para fixação de gaiola de proteção. (6) gaiola de proteção hidrofone.

Fonte: o autor (2015)

2.1.2. Fornecimento de energia

Para as dimensões do equipamento citado no item anterior, é possível utilizar até 5 pacotes de pilhas ligadas em paralelo, sendo que cada pacote é composto por 4 pilhas ligadas em série, como apresentado na Figura 2.3.

Figura 2.3.: Esquema de arranjo das baterias na unidade de PAM. (a) pack de pilhas em série. (b) conjunto de pack de pilhas em paralelo e (c) conversor DC-DC.

Fonte: o autor (2015)

A tensão elétrica nominal fornecida pelo arranjo de pilhas é de 6V. A eletrônica de gravação trabalha com tensão de 5V. Portanto, utiliza-se um conversor DC-DC que transforma a tensão de 6V para 5V. Na configuração atual não há interruptor de energia, sendo que assim que o equipamento é energizado, entra em modo de gravação de dados.

2.1.3. Eletrônica de controle

O OceanPod é baseado em um computador de placa única que liga e controla diferentes periféricos necessários para a aquisição dos dados acústicos. O modelo usado foi a Raspberry PI (RPI) modelo A, desenvolvida pela fundação Raspberry PI (Figura 2.4-a). Essa unidade de controle centraliza as funções de controle da interface de aquisição, gravação de dados, sincronização temporal e interface de usuário. Utilizou-se como sistema operacional (SO) o Linux. Duas distribuições foram testadas, a Raspbian (www.raspbian.org) e o Archlinux (www.archlinux.org), sendo que esta última se mostrou mais adequada, robusta e confiável na configuração e funcionamento dos diversos periféricos necessários foi a Archlinux. O SO é instalado num cartão MicroSDx classe 10 (Figura 2.4-c).

Para sincronizar a data e hora do sistema foi instalado um relógio externo RTC (Real Time Clock) DS-3231 de alta precisão da Maxim Integrated (Systems, 2015), cuja deriva temporal é inferior a um minuto por ano.

A RPI além de ter memória embutida, um microprocessador, portas USB e ser capaz de executar um SO baseado em Linux, possui um conjunto de portas de entrada / saída de uso geral (GPIO), que possibilitam a conexão e controle de diversos tipos de periféricos. No OceanPod esses pinos são usados para energizá-la e para conectar o RTC.

2.1.4. Aquisição do sinal

Para a captura, condicionamento e digitalização dos sinais de áudio, foi usado um hidrofone piezoelétrico ligado a uma placa de condicionamento de sinal, a qual por sua vez está ligada a uma interface de áudio comercial, esta última encarregada de digitalizar e

Figura 2.4.: Unidade de processamento. (a) Placa RPI. (b) Relógio RTC. (c) Cartão Micro-SDx

Fonte: (WALLACE; RICHARDSON,2012) E (SARPI,2015)

enviar o sinal na RPI para armazenamento (Figura 2.5).

Figura 2.5.: (a) Placa da interface de áudio, (b) Condicionador de sinal, (c) Hidrofone piezoelétrico.

Figura 2.6.: Fonte: imagem de UCA222 adaptada de (INSTRUCTABLES, 2015). Condicionador e hidrofone: o autor (2015)

Como apresentado na figura, o sinal vindo do hidrofone (c) passa para a placa de condicionamento (b) que amplifica o sinal. Esse sinal é enviado para a interface de áudio (a) que o digitaliza e o envia para a RPI, que finalmente armazena os sinais adquiridos.

O hidrofone usado (Figura 2.5-c) é um cilindro piezoelétrico PZT-5 (e), cuja sensibilidade é , e com resposta em frequência plana, desde aproximadamente $0Hz$ (DC) até $30kHz$. O hidrofone está conectado a uma placa de condicionamento de sinal projetada no Ladin (Figura 2.5-b).

Essa placa possui um filtro passa banda com limites entre $5Hz$ e $24.000 Hz$, além de dois estágios de ganho que podem amplificar o sinal vindo do hidrofone em até . A curva de resposta de frequência experimental é mostrada na Figura 2.7. Pode-se observar que a resposta da placa de condicionamento do sinal é plana na banda de $20Hz$ e até $20Khz$, aproximadamente.

Figura 2.7.: Curva de resposta em frequência experimental para a placa de condicionamento de sinal do hidrofone

Fonte:(CALDAS-MORGAN, M.; ALVAREZ-ROSARIO, A.; PADOVESE, L.R. 2015)

A interface de áudio USB usada é a UCA222, do fabricante Behringer, a qual está conectada à RPI via USB. Esta interface possui um conversor ADC de 16 bits que permite usar frequências de amostragem de 8.000 , 11.025 , 16.000 , 22.050 , 32.000 , 44.100 e $48.000Hz$.

A UCA222 digitaliza o sinal vindo do hidrofone através da placa de condicionamento (amplitude do sinal em milivolts) em 2^{16} valores ou “Full scale”. Os arquivos são gravados

em formato PCM-WAV, que normaliza o sinal entre -1 e 1. Para expressar o sinal em unidades de pressão acústica [Pa] é necessário uma calibração cujo procedimento é descrito em (Caldas-Morgan et al., 2015).

Os sinais são gravados no cartão de memória microSDx da RPI. Desta forma, a autonomia de armazenamento de dados irá depender do tamanho do cartão utilizado. Convém observar que essa autonomia pode ser aumentada utilizando-se um hub USB e pen-drives ou HD USB. Mas verificou-se que isto incrementa o consumo de energia.

Portanto, a autonomia depende de um balanço entre o consumo de toda a eletrônica e a capacidade de armazenamento.

2.1.5. Software de controle do OceanPod

Como foi mencionado anteriormente, para a placa RPI funcionar é necessário instalar um SO. O sistema instalado foi a distribuição de Linux Archlinux. Junto com o SO também são instalados os programas requeridos para controlar cada um dos periféricos conectados, como o RTC e a interface de áudio. A seguir são descritos os programas necessários para gerenciar o sistema e a aquisição de dados e que devem ser instalados manualmente no Linux:

- **Alsa** (Advanced Linux Sound Architecture) é um software que oferece os drivers necessários para reconhecimento da interface de áudio.
- **Sox** (Sound eXchange) é um software utilitário que oferece um conjunto de instruções para processamento de arquivos de áudio.
- **I2C** (Inter-Integrated Circuit) é um protocolo de comunicação utilizado para conectar periféricos a uma placa mãe. O I2C é usado para configurar o relógio RTC, responsável pela data e hora do sistema no OceanPod.
- **WiringPi** é uma biblioteca escrita em linguagem C que permite o acesso e controle dos pinos de entrada e saída (input/output GPIO) presentes na RPI, onde são conectados os diversos periféricos, como o RTC por exemplo.
- **Shellscript** é uma ferramenta de automação de instruções e que vem junto com o Linux. Através de uma série de comandos inseridos num arquivo de texto são controladas todas as ações e iterações entre periféricos e o SO.

2.1.6. Gestão de arquivos e aquisição de dados

A gestão lógica do sistema é feita através de 3 processos principais. O primeiro deles contido num script de inicialização do sistema que, além de realizar algumas configurações básicas para a RPI, controla o lançamento dos outros processos, sincronização de data e hora e aquisição de áudio. A seguir são tratados com detalhe cada um desses subprocessos.

Inicialização do Sistema

Assim que o OceanPod é ligado, o SO é carregado e de forma automática um script de inicialização é executado. A sequência de operações é apresentada no diagrama de blocos da Figura 2.8. A primeira tarefa a realizar é desativar alguns processos na RPI (bloco 2), que são carregados no início do SO de forma padrão e não são necessários para o funcionamento do OceanPod, como por exemplo o suporte para saída de vídeo e com o intuito de economizar energia. Depois disso, são executados em forma sequencial os subprocessos de ajuste de data e hora do sistema (bloco 3), e o subprocesso de gravação (bloco 4), explicados mais a frente neste capítulo.

Figura 2.8.: Algoritmo de inicialização do SO no Oceanpod

Fonte: o autor (2015)

Configurando data e hora do SO

O subprocesso de sincronização da data e hora do sistema é lançado a partir do script de inicialização do sistema, apresentado anteriormente (Figura 2.8 bloco 3). As etapas deste subprocesso são apresentadas no diagrama de blocos da Figura 2.9. Uma vez iniciado o sistema, os módulos de software necessários para estabelecer comunicação com o RTC

são carregados (módulos I2c), com isso é possível acessar a hora e data real e atualizar o SO (bloco 3).

Uma vez que data e hora do SO são sincronizados, os módulos I2c são descarregados da memória (bloco 4) e o script é finalizado (bloco 5), retornando o controle ao script de inicialização do sistema.

Figura 2.9.: Algoritmo de sincronização de hora e data do SO no Oceanpod

Fonte: o autor (2015)

Aquisição do áudio

O subprocesso de aquisição do áudio é feito em 3 etapas, ilustradas nos blocos B1, B2, e B3 da Figura 2.10. Cada etapa consta de uma sequência linear de duas operações básicas, aquisição e pós-processamento do segmento de áudio. No entanto, para garantir a continuidade na sequência de dados de áudio total, após a aquisição de cada segmento (bloco 1) é iniciada a aquisição em paralelo do próximo segmento (bloco 4) com o pós-processamento do segmento inicial (bloco 2). A mesma lógica é aplicada para cada bloco B1, B2, e B3.

O script de aquisição de áudio, como implementado, é um loop infinito onde a gravação vai parar assim que a energia do OceanPod acabar ou quando atingida a capacidade máxima de armazenamento. Já que não há como sair desse loop, um bloco adicional (B4) faz uma comparação da hora do SO com a hora do RTC, e caso haja alguma diferença significativa, a hora do SO é atualizada mediante a invocação do script de sincronização de data e hora explicado no tópico anterior.

Figura 2.10.: Algoritmo de aquisição de dados no Oceanpod

Fonte: o autor (2015)

Tabela 2.1.: Desenho experimental para teste de consumo do Oceanpod

Fonte de energia: Fonte de alimentação de bancada						
Canais de áudio	Voltagem de entrada	$F_s(kHz)$				
1 (mono)	7,5V	11.025	16	22	32	44 48

2.1.7. Metodologia de teste de consumo

Para estimar o consumo de energia do OceanPod, três experimentos foram feitos. No primeiro experimento (Tabela 2.1), o OceanPod foi energizado com uma fonte de alimentação de bancada e várias frequências de amostragem.

O segundo experimento segue o mesmo padrão do experimento 1 mas com uma operação de reamostragem digital do sinal (filtragem passa baixa e reamostragem), ou seja, com um esforço computacional um pouco maior (Tabela 2.1).

No terceiro experimento, o OceanPod foi alimentado por um conjunto de pilhas e testado no caso mais crítico, ou seja operando à máxima F_s e aplicando uma filtragem do sinal adquirido. O projeto do teste é apresentado na Tabela 2.2.

Tabela 2.2.: Parâmetros usados em teste com baterias

Fonte de energia: Baterias		
Canais de áudio	Voltagem de entrada	Frequências de amostragem
1 (mono)	7,5V (5x1.5V)	48.000

2.1.8. Teste de Gravador de dados no mar

O teste final dos protótipos consistiu na instalação deles em alguns locais do litoral brasileiro (Figura 2.11). As configurações dos OceanPod foram diferentes para algumas das missões. A seguir são descritos os locais de instalação e configuração das unidades instaladas (Tabela 2.3):

Figura 2.11.: Locais de instalação dos OceanPod. (a) Parnaíba – PI. (b) Ilhéus - BA. (c) Ilha das Palmas – SP. (d) Laje de Santos – SP

Fonte: o autor (2015)

Tabela 2.3.: Locais de instalação dos Oceanpod. Unidades e configurações

Local	Objetivo	Unidades	Fs(Hz)	Tipo gravação
Ilhéus(BA)	Baleia Jubarte	4	11.025	Agendada
Parnaíba (PI)	Peixe boi	1	11.025	Manual
Laje de Santos (SP)	Golfinhos	1	44.100	Continua
Ilha das Palmas (SP)	Embarcações	1	16.000	Continua
Pecém (CE)	Golfinhos	1	48.000	Continua

2.2. RESULTADOS E DISCUSSÃO

2.2.1. Unidades fabricadas

Foram fabricadas várias unidades OceanPod, em duas configurações dimensionais distintas. O primeiro modelo tinha capacidade para alocar no máximo 5 packs de pilhas (20 pilhas), enquanto que no segundo (Figura 2.12) a capacidade foi dobrada, sendo necessário aumentar o comprimento do tubo de PVC de 52 cm para 100 cm.

Figura 2.12.: OceanPod com capacidade para 40 pilhas alcalinas tipo D

Fonte: o autor (2015)

2.2.2. Teste de estanqueidade

Para analisar e testar as funcionalidades do OceanPod, bem como calibrar o hidrofone, foi necessário desenvolver uma pequena infraestrutura de apoio. Um dos equipamentos de apoio desenvolvidos foi o tanque de pressão para teste hidrostático dos OceanPod. Ele se compõe de um vaso de pressão cilíndrico, com uma tampa parafusada, e de uma bomba hidráulica manual (Figura 2.13).

Cada protótipo do OceanPod construído foi testado mantendo a pressão constante em 7 bar por períodos de três dias consecutivos. Em cada caso foram avaliadas as tampas e o conector do hidrofone e foi comprovado que não houve vazamento no interior do invólucro.

O projeto tanto dos conectores quanto das tampas com anel de vedação “o-ring” mostrou-se robusto o suficiente para garantir estanqueidade a profundidades de até 80m. Os protótipos foram instalados em locais com profundidade menor que 30m. Não foi detectado vazamento em nenhuma das instalações realizadas no mar.

Figura 2.13.: Vaso de Pressão usado para teste de estanqueidade

Fonte: o autor (2015)

2.2.3. Teste de consumo

No item 2.1.7 **OJO AQUI** descreveu-se a metodologia de teste de consumo de energia pela eletrônica do OceanPod.

Teste 1: O resultado do consumo de energia ao longo do tempo é mostrado na Figura 2.14, onde é observado que o consumo de energia aumenta ligeiramente, com a taxa de

amostragem, numa proporção linear. O consumo médio de energia em modo de espera (todos os periféricos eletrônicos alimentados, mas sem aquisição e gravação de áudio) e em estado de repouso “Standby” (Conversor DC-DC principal com alimentação mas RPI e restantes periféricos sem alimentação) foi $828.73mWH$ e $89.5mWH$, respectivamente.

Figura 2.14.: Consumo de energia no OceanPod em modo de gravação para diferentes frequências de amostragem

Fonte: o autor (2015)

Teste 2: Os resultados são coerentes com os apresentados no Teste 1. Pode ser observado na Figura 2.15, que houve um aumento no consumo de energia para todas as taxas de amostragem. Os incrementos no consumo são apresentados como percentagem em relação aos dados obtidos no Teste 1.

Teste 3: Na Figura 2.16 apresentam-se as curvas de consumo do OceanPod alimentado por um conjunto de pilhas, conforme descrito no item 2.1.7. No início do teste, o consumo do sistema se aproxima de $880mW$. Passados 1.92 dias aproximadamente, a corrente consumida permanece constante perto de $144mA$.

O sistema continua a funcionar bem até passadas 74.4 horas (aproximadamente 3,1 dias). Nesse instante, as pilhas já não podem continuar energizando o sistema RPI e este para de funcionar, fato evidenciado pela queda dramática na corrente (linha vermelha na Figura 2.16). A partir desse patamar, o OceanPod não pode ser mais ligado. Por causa disso, a leitura da tensão das baterias aumenta ligeiramente, mas num nível menor ao requerido pela RPI para operar normalmente ($5.5V$).

Figura 2.15.: Consumo de energia no OceanPod em modo de gravação para diferentes frequências de amostragem (incrementos percentuais associados com pós-processamento do sinal – “downsampling”)

Fonte: o autor (2015)

Figura 2.16.: Curvas de consumo energético do OceanPod e de tensão e corrente fornecidas pelo conjunto de pilhas no Teste 3)

Fonte: : CALDAS-MORGAN; ALVAREZ-ROSARIO; PADOVESE,2015

Tabela 2.4.: Resumo dados gravados nas diferentes instalações do Oceanpod

Local	Fs(kHz)	Data instalação	Tempo	Tamanho (GB)
Ilhéus (BA)	11	06/07/14-30/09/14	3.281,3	248,4
Parnaíba (PI)	16	25/03/14-11/06/14	169,08	12,8
Laje de Santos (SP)	11	28/01/15-03/03/15	1.667,6	120,7
Ilha das Palmas (SP)	16	24/11/13-11/12/15	348,73	26,4
Pecém (CE)	44	06/05/14-07/05/14	118,9	9

Teste de aquisição de dados no mar

Com estes testes foi possível avaliar o desempenho e confiabilidade dos protótipos em condições de gravação e operação real no mar.

Já com os testes de campo foi possível a criação de um banco de dados de sinais acústicos submarinos que contêm eventos acústicos de diversos tipos, tanto biológicos como antropogênicos.

Este banco de dados foi essencial para o desenvolvimento dos capítulos subsequentes neste projeto de pesquisa, que tratam do desenvolvimento de métodos de processamento de sinais para detecção de eventos acústicos.

Na Tabela 2.4 são resumidos os tempos de gravação e tamanho dos arquivos obtidos nas diferentes campanhas de medição assim como locais de instalação. Na Figura 2.17 são apresentados alguns espectrogramas dos diversos tipos de sinais registrados pelos OceanPod.

2.2.4. Teste de aquisição de dados no mar

Com estes testes foi possível avaliar o desempenho e confiabilidade dos protótipos em condições de gravação e operação real no mar.

Já com os testes de campo foi possível a criação de um banco de dados de sinais acústicos submarinos que contêm eventos acústicos de diversos tipos, tanto biológicos como antropogênicos.

Este banco de dados foi essencial para o desenvolvimento dos capítulos subsequentes neste projeto de pesquisa, que tratam do desenvolvimento de métodos de processamento de sinais para detecção de eventos acústicos.

Na Tabela 2.4 são resumidos os tempos de gravação e tamanho dos arquivos obtidos nos diferentes locais de instalação e, na Figura 2.17 são apresentados alguns espectrogramas

Figura 2.17.: Espectrogramas para diversos tipos de sinal. (a) Tráfego marinho no porto de Santos – Ilha das Palmas. (b) Vocalizações de golfinhos estuarinos no Litoral de Fortaleza – Pecém. (c) Vocalizações de Baleia Jubarte – Ilhéus.)

(a)

(b)

(c)

Fonte: : CALDAS-MORGAN; ALVAREZ-ROSARIO; PADOVESE,2015

dos diversos tipos de sinais registrados pelos OceanPod.

2.3. CONCLUSÕES

O equipamento de autônomo de gravação acústica submarina OceanPod desenvolvido mostrou-se robusto e confiável em operação no mar. Atende às especificações necessárias para monitoramento de paisagens acústicas submarinas.

Os testes de consumo em laboratório permitiram estabelecer qual é autonomia do OceanPod em alguns cenários. Posteriormente os testes de campo permitiram obter essa informação em condições reais de operação.

Existe um compromisso (para um dado conjunto de baterias) entre o consumo de energia e a capacidade de armazenamento disponível.

Assim, com as restrições atuais do projeto (geometria, número de baterias, capacidade do cartão SD), e em regime de gravação contínua, o protótipo é capaz de operar com sucesso durante 31 dias, usando uma F_s de $11.025Hz$, e um cartão MicroSDx de 64GB.

O consumo do OceanPod é maior se comparado com equipamentos comerciais disponíveis no mercado. No entanto, essa não é uma desvantagem significativa se considerada a flexibilidade do OceanPod para digitalizar o áudio em quase qualquer F_s via pós-processamento, permitindo realizar missões de alguns dias até vários meses de duração.

Também é importante observar que o custo de replicação do OceanPod é uma fração do custo de equipamentos adquiridos no mercado internacional. Estima-se em R\$2.000,00, que compreende componentes e mão de obra de usinagem e de montagem de circuitos eletrônicos.

Adicionalmente, o OceanPod é um equipamento micro-processado onde realizar mudanças nos scripts de aquisição e processamento é relativamente simples já que não requer conhecimentos avançados de programação. Isso o torna relativamente fácil de adaptá-lo a diferentes tipos de missões.

Os dados coletados nas missões de gravação indicaram que o protótipo é útil para gravar sinais de fontes naturais, biológicas e antropogênicas. Este banco de dados coletado foi especialmente útil no desenvolvimento de métodos de detecção automática de eventos acústicos, apresentados nos capítulos seguintes.

3. ALGORITMOS DE TEMPO-REAL PARA DETECÇÃO DE EVENTOS ACÚSTICOS

Em campanhas de monitoramento acústico é comum que o volume de dados gerado seja muito grande. Campanhas de monitoramento diário e contínuo, durante meses, são algo comum. Esse grande volume de dados torna inviável a análise visual ou auditiva, para detecção de eventos acústicos em geral ou de algum padrão em particular. Portanto, alguma estratégia de detecção automatizada precisa ser adotada.

O procedimento de detecção pode ser realizado off-line, quando os dados são analisados após recuperação das unidades PAM, e de maneira on-line ou em tempo real, quando ela é realizada em paralelo com o monitoramento. Neste último caso, a detecção só é possível se a tecnologia embarcada nos gravadores autônomos tiver recursos para processar posteriormente os sinais em tempos compatíveis com a taxa de aquisição.

Como a tecnologia utilizada nos equipamentos de monitoramento desenvolvidos no Ladin é baseada num computador de placa única, existe a possibilidade de se implementar algoritmos de detecção de forma paralela ao processo de aquisição e gravação. Resta analisar se o esforço computacional exigido pelos algoritmos é compatível com o poder computacional desse computador.

Adicionalmente, a questão da computação em tempo-real (ou tempo menor que a velocidade de aquisição) é importante também para análise off-line na medida em que diminui o tempo de cálculo em grandes bancos de dados.

Neste contexto, o presente capítulo apresenta alguns métodos de detecção utilizados no PAM, a implementação de algoritmos de detecção e testes realizados em diferentes cenários e configurações de processamento, visando a computação em tempo real.

Informações complementares podem também ser em (Caldas-Morgan et al., 2015), trabalho apresentado no RioAcoustics 2015.

3.1. MÉTODOS DE DETECÇÃO

Os métodos de detecção de eventos acústicos podem ser classificados em dois grandes grupos dependendo do tipo de processamento feito no sinal: os métodos supervisionados

e os não supervisionados.

Os métodos supervisionados são utilizados quando se quer detectar um tipo determinado de evento acústico (uma embarcação de tipo determinado, um cetáceo, etc.). Em geral são usados métodos baseados na correlação (Abbot et al., 2010; Mellinger, 2004; Mellinger et al., 2011; Nanaware et al., 2014), redes neurais (Dugan et al., 2010; Mellinger and Clark, 2000), métodos estatísticos multivariados (Binder and Hines, 2012; Diaz Lopez, 2011), dentre outros. Esses métodos supõem que amostras do padrão do evento acústico de interesse sejam conhecidas e estejam disponíveis. De forma geral esses métodos “comparam” esses padrões com os sinais do banco de dados, selecionando os trechos em que há uma semelhança.

Há ocasiões em que não se deseja procurar um evento acústico determinado, mas sim detectar todas as ocorrências de eventos acústicos no monitoramento da paisagem acústica local. Uma vez localizados esses eventos, eles podem ser analisados individualmente com maior detalhe. Para esse tipo de análise, métodos não supervisionados são os mais adequados e possuem um custo computacional menor. Os métodos não supervisionados por excelência são aqueles que utilizam a envoltoria ou envelope do sinal como função de detecção. Podem ser baseados em médias móveis de 1ª e 2ª ordem, e na transformada de Hilbert.

No contexto do presente trabalho o interesse está em desenvolver métodos de detecção globais computacionalmente leves. Portanto, serão utilizados métodos de cálculo de envelope. Na sequência apresentam-se diferentes formas de cálculo do envelope.

3.2. MÉTODOS DE CÁLCULO DO ENVELOPE DO SINAL

Baseiam-se nas mudanças dos níveis de energia do sinal e são chamados de detectores de energia. São métodos de detecção úteis para encontrar qualquer evento que seja de interesse, particularmente quando os sons procurados são desconhecidos ou altamente variáveis. Podem ser usados como o primeiro passo para um sistema de classificação mais sofisticado (Mellinger, 2002). A eq.3.1 expressa o teorema de Parseval (Oppenheim et al., 1999), bem como a forma de calcular a energia normalizada de um sinal, seja no domínio temporal (2º membro da equação), seja no domínio espectral (3º membro da equação).

$$E_x = \frac{1}{N} \sum_{n=0}^{N-1} |x[n]|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2 \quad (3.1)$$

onde $x[n]$ é o sinal temporal discreto, N é o número de amostras, $|X[k]|$ é o módulo Transformada de Fourier Discreta (DFT do termo em inglês).

Estes detectores de energia são utilizados pela comunidade de PAM nos seguintes softwares livres atualmente disponíveis: Ishmael (Mellinger, 2002), PAMGUARD (Gillespie and Chappell, 2002), Xbat (Charif and Pitzrick, 2008) e no software comercial Raven (of Ornithology, 2010).

O algoritmo de implementação é relativamente simples: o sinal é segmentado em janelas temporais de tamanho previamente fixado, dependendo da duração do evento a ser detectado (1/3 da duração do evento procurado). Para cada janela, a energia é calculada usando eq. 3.1. O resultado obtido é um novo sinal que expressa o comportamento da energia do sinal original ao longo do tempo. Esse novo sinal é chamado de **função de detecção**, já que com ajuda de um valor limiar pré-determinado, é possível detectar onde há mudanças abruptas da energia do sinal, o que indicaria a ocorrência de um evento acústico.

Uma forma derivada da expressão de energia normalizada, descrita pela eq.3.1, é a sua raiz quadrada, conhecida como rms do sinal. A função de detecção (envelope) é então calculada através do cálculo do rms em uma janela móvel sobre o sinal.

Em termos computacionais, é possível implementar a função de cálculo do rms usando várias estratégias, as quais podem apresentar um melhor ou pior desempenho em tempos de execução, dependendo das capacidades de cálculo do sistema avaliado. Neunaber (Neunaber, 2015) discute algumas estratégias de implementação do rms, otimizadas para vários tipos de hardware embarcado, mostrando que, nem sempre um método é melhor que os outros de forma absoluta, mas que isso depende da forma em que os cálculos são feitos no processador matemático.

Apresenta-se a seguir vários métodos de cálculo do envelope através do valor rms do sinal, assim como o tradicional método da transformada de Hilbert e a média móvel.

Envelope usando a Transformada de Hilbert

A transformada de Hilbert é comumente utilizada para o cálculo de envelopes e implica no cálculo do sinal analítico de um determinado sinal $x(t)$. O sinal analítico é um sinal complexo, no qual, a parte real é o sinal original e a parte imaginária é a transformada de Hilbert do sinal original.

Matematicamente o envelope $e(t)$ de um sinal $x(t)$ é definido como a magnitude do sinal analítico, como mostrado pela seguinte equação:

$$e(t) = \sqrt{x(t)^2 + \hat{x}(t)^2} \quad (3.2)$$

onde $\hat{x}(t)$ é a transformada de Hilbert do sinal $x(t)$ (Oppenheim et al., 1999).

Média móvel

Outra forma de calcular o envelope do sinal é usando o filtro de média móvel. O filtro de média móvel é obtido calculando-se a média de um número de pontos do sinal original, para produzir cada ponto numa função de saída. A expressão matemática é escrita como (Smith, 1997):

$$e[i] = \frac{1}{M} \sum_{j=0}^{M-1} |x[i+j]| \quad (3.3)$$

onde $x[i]$ é o sinal de entrada, $e[i]$ é o sinal de saída, e M o número de pontos usados para o cálculo da média móvel.

rms para janelas deslizantes

Rms em janelas deslizantes pode ser visto como uma medida de energia média no sinal (The MathWorks, 2015). Também pode ser visto como um filtro de média móvel, onde a média é de 2ª ordem. Essa abordagem é usada atualmente em softwares de PAM como o Ishmael (Mellinger, 2002), Pamguard (Gillespie and Chappell, 2002), Raven (of Ornithology, 2010) e X-bat (Figuerola, 2007). A equação que descreve o cálculo do rms em janelas deslizantes é apresentado a seguir:

$$e[i] = \sqrt{\frac{1}{N} \sum_{j=1}^w x^2[i+j]} \quad (3.4)$$

"Full" rms

Nesta implementação, o valor rms é calculado centrado no ponto $x[i]$. Isto é, em cada ponto do vetor x , são avaliados os $\frac{W}{2}$ pontos prévios e os $\frac{W}{2}$ pontos posteriores. Pode ser entendido como um rms calculado por janelas deslizantes mas, com uma maior porcentagem de sobreposição entre janelas.

$$e[i] = \frac{\sqrt{(\sum_{j=1}^W x[i + j + W/2]^2)}}{W} \quad (3.5)$$

Onde:

x é o vetor de dados W é a janela ou número de pontos para calcular o valor instantâneo do rms

"Fast" rms

Produz o mesmo resultado que a implementação *full rms*, mas usando a técnica computacional de memorização para diminuir o número de cálculos necessários. Neste algoritmo, para cada posição rms calculada, o primeiro elemento da janela W é retirado e o valor do novo ponto $[i + W/2]$ é acrescentado.

$$e[i] = \frac{\sqrt{e[i-1]^2 - D[i - W/2 - 1]^2 + D[i + W/2]^2}}{W} \quad (3.6)$$

rms recursivo

Aqui um novo valor *rms* instantâneo é calculado, usando metade do valor *rms* previamente calculado (3.6). O valor " a " é definido de forma empírica como 0,7.

$$e[i] = a \times RMS[i] - 2(1 - a) \frac{\sqrt{\sum_{j=1}^{W/2} RMSrec[i - j]}}{W} \quad (3.7)$$

3.3. PLANEJAMENTO DO EXPERIMENTO COMPUTACIONAL

3.3.1. Hardware

Para a execução dos testes, os materiais usados foram uma RPI modelo *B+*, um computador e um roteador para estabelecer a ponte de comunicação com a RPI. O arranjo dos dispositivos mencionados é apresentado na Figura 3.1.

Figura 3.1.: Configuração dos dispositivos no experimento. (a) RPI, (b) Roteador, (c) Computador.

Figura 3.2.: Fonte: o autor (2015)

Trata-se de um experimento de desempenho usando um banco de sinais de áudio já existente.

3.3.2. Software

Os softwares usados na RPI foram o SO Archlinux (www.archlinux.org), e Python (www.python.org) como linguagem de programação junto com as bibliotecas Numpy e Scipy. A escolha desta linguagem de programação obedece à simplicidade de uso e o suporte que oferece para manipulação de dados e processamento de sinais respectivamente.

Banco de sinais acústicos submarinos

Para realizar o experimento, foi utilizado um grupo de arquivos de áudio de PAM contendo vocalizações de baleia jubarte e boto cinza. Os sinais contendo vocalizações de baleia Jubarte foram obtidos em campanha de monitoramento na região de Ilhéus (BA), durante o período de migração de baleias jubarte (agosto a novembro de 2014). Esse monitoramento foi realizado em colaboração com a bióloga marinha MSc Isabel Gonçalves da UESC. Os arquivos de boto cinza foram obtidos em monitoramento na região do Porto de Pecém, CE.

Com objetivo de se avaliar o impacto da F_s no cálculo do envelope, esses arquivos foram re-amostrados para criar um banco de sinais com diferentes valores. de F_s como apresentado a seguir:

- baleia jubarte, $Fs = [2\ 4\ 8\ 11]kHz$
- boto cinza, $Fs = [16\ 22\ 24\ 32\ 44]kHz$

Para cada Fs , há um conjunto de 10 arquivos de áudio. Assim o banco de sinais utilizado no experimento computacional tem 100 arquivos - 50 arquivos com vocalizações de baleia jubarte e 50 arquivos com vocalizações de boto cinza.

Cada arquivo de áudio possui uma duração de 60 segundos, que corresponde ao valor mínimo adotado na aquisição usando o OceanPod.

3.3.3. Planejamento experimental

O experimento consiste em avaliar cada um dos métodos de cálculo de envelopes para cada um dos arquivos do banco de dados (diferentes taxas de amostragem), para diferentes tamanhos de janela W e porcentagens de sobreposição entre janelas. Os valores destes parâmetros são definidos como segue:

- $W = [128\ 256\ 512\ 1024\ 2048\ 4096\ 8192]$ (pontos)
- Sobreposição = $[0,0\ 0,25\ 0,50\ 0,75]$ (porcentagem)
- $Fs = [2\ 4\ 8\ 11\ 16\ 22\ 24\ 32\ 44]kHz$

Para execução dos testes, um algoritmo e uma biblioteca de funções foi implementada na linguagem Python. O algoritmo pode ser observado na Figura 3.3. Nele, todos os métodos, valores de W , sobreposição e arquivos a serem processados são armazenados em listas (bloco 1).

A partir bloco 2 e até o bloco 5, cada item das listas é avaliado, realizando os cálculos (bloco 6) até que todas as listas sejam esvaziadas. O experimento foi repetido 100 vezes para calcular o tempo médio de processamento. Dada a quantidade de combinações O código Python correspondente ao algoritmo apresentado, assim como a implementação dos métodos aparecem no Anexo B e C respectivamente.

3.4. RESULTADOS E DISCUSSÃO

Foi feito um teste inicial, visando determinar a viabilidade de cada método. No teste, um arquivo de áudio com $Fs = 2kHz$ foi processado em janelas de 128 pontos, para cada um dos métodos.

Figura 3.3.: Diagrama de fluxo do algoritmo de avaliação dos métodos de calcular envelopes (funções) para todos os casos de Fs, W e Sobreposição.

Figura 3.4.: Fonte: o autor (2015)

Tabela 3.1.: Resultados da primeira rodada com arquivo de áudio de $Fs = 2kHz$

	Full rms	Fast rms	rms recursivo	Transformada de Hilbert
Tempo (s)	160,30	127,64	245,86	142,80

Nessa configuração de arquivo, esperava-se que os algoritmos tivessem o melhor desempenho, por ser arquivos com menor número de pontos.

Os tempos de processamento dos métodos Full rms, Fast rms, rms Recursivo e Transformada de Hilbert foram maiores que 60s (Tabela 3.1) , que é o tempo de duração do áudio analisado, ou sinal de entrada no algoritmo.

Pelo fato dos tempos de processamento serem maiores que o tempo de duração do sinal analisado, esses métodos são inviáveis para processamento em tempo real, pois é necessário que o tempo de cálculo seja inferior ao tempo de duração do sinal processado.

No mesmo teste, os tempos de execução para os métodos da média móvel e o rms em janelas deslizantes, este último com diferentes porcentagens de sobreposição, apresentaram tempos inferiores a 60s como apresentado na Tabela 3.2.

Uma vez descartados os métodos não viáveis (no caso mais favorável) para processa-

Tabela 3.2.: Resultados da primeira rodada com arquivo de áudio de $Fs = 2kHz$

Tempo (s)	rms janelas deslizantes (% de sobreposição)				
	Média móvel	75	50	25	0
	0,26	2,69	1,39	0,97	0,72

Tabela 3.3.: Tempos de execução com modelo de Média móvel para cálculo do envelope

Fs(Hz)	Tamanho de janela (Número de amostras)						
	128	256	512	1024	2048	4096	8192
2000	3,84	2,05	1,20	0,73	0,49	0,36	0,29
4000	7,64	4,10	2,40	1,45	0,97	0,72	0,58
8000	15,26	8,19	4,80	2,90	1,95	1,43	1,17
11025	21,05	11,28	6,58	3,99	2,68	1,96	1,59
16000	30,55	16,39	9,58	5,81	3,90	2,79	2,28
22050	42,21	22,61	13,23	7,98	5,36	3,93	3,20
24000	45,91	24,55	14,38	8,70	5,85	4,32	3,45
32000	61,30	32,81	19,12	11,57	7,78	5,70	4,69
44100	84,33	45,24	26,45	16,02	10,73	7,84	6,41

mento em tempo real, os métodos de rms em janelas deslizantes e Média Móvel foram analisados na totalidade do conjunto de dados de áudio, e para todos os tamanhos de janela e sobreposição. Os resultados são apresentados nos itens seguintes.

3.4.1. Resultados média móvel

A Média Móvel apresentou resultados satisfatórios para 61 das 63 combinações possíveis de Fs e W e sobreposição de 75%, falhando apenas quando o tamanho de janela $W = 128$ e $Fs \geq 32kHz$. O Caso apresentado é o mais custoso computacionalmente devido à percentagem de sobreposição escolhida. Para valores de sobreposição menores, a média móvel mostrou-se satisfatória para o cálculo do envelope. Os resultados são apresentados na Tabela 3.3.

3.4.2. Resultados rms com janelas deslizantes

Os tempos de processamento obtidos com o método de *rms* com janelas deslizantes foi menor que o tempo de duração do áudio para 100% dos casos, e para qualquer combinação de Fs , W e porcentagem de sobreposição (Tabelas 3.4 a 3.7).

Os tempos de processamento caem, com a diminuição da Fs , com a porcentagem de sobreposição e com o aumento da janela temporal. Isto era esperado já que, quanto menor número de janelas e o número de pontos de cada uma, menor serão os cálculos necessários para completar a tarefa.

Tabela 3.4.: Tempos de execução com modelo rms de janelas deslizantes e sobreposição 0.75

Fs(Hz)	Tamanho de janela (Número de amostras)						
	128	256	512	1024	2048	4096	8192
2000	2,69	1,43	0,82	0,49	0,33	0,24	0,19
4000	5,39	2,86	1,64	0,99	0,65	0,47	0,38
8000	10,83	5,74	3,29	1,97	1,31	0,95	0,76
11025	14,81	7,89	4,52	2,71	1,79	1,30	1,05
16000	21,52	11,47	6,59	3,94	2,60	1,89	1,52
22050	29,77	15,81	9,06	5,43	3,58	2,61	2,10
24000	32,37	17,21	9,88	5,91	3,89	2,84	2,30
32000	43,04	22,90	13,23	7,91	5,20	3,78	3,05
44100	59,26	31,64	18,10	10,86	7,15	5,22	4,20

Tabela 3.5.: Tempos de execução com modelo rms de janelas deslizantes e sobreposição 0.75

Fs(Hz)	Tamanho de janela (Número de amostras)						
	128	256	512	1024	2048	4096	8192
2000	1,38	0,75	0,44	0,27	0,18	0,14	0,11
4000	2,75	1,49	0,88	0,55	0,36	0,27	0,22
8000	5,51	3,00	1,76	1,09	0,73	0,54	0,44
11025	7,61	4,10	2,42	1,50	0,99	0,74	0,61
16000	11,01	6,00	3,52	2,18	1,44	1,07	0,89
22050	15,24	8,25	4,84	3,01	1,99	1,49	1,23
24000	16,56	8,97	5,28	3,27	2,16	1,61	1,33
32000	22,03	11,94	7,10	4,37	2,89	2,14	1,78
44100	30,35	16,51	9,69	6,00	3,98	2,95	2,47

Tabela 3.6.: Tempos de execução com modelo rms de janelas deslizantes e sobreposição 0.25

Fs(Hz)	Tamanho de janela (Número de amostras)						
	128	256	512	1024	2048	4096	8192
2000	0,94	0,52	0,31	0,20	0,13	0,10	0,08
4000	1,87	1,03	0,62	0,39	0,26	0,20	0,17
8000	3,74	2,07	1,25	0,78	0,53	0,40	0,33
11025	5,17	2,85	1,71	1,07	0,72	0,55	0,46
16000	7,52	4,14	2,48	1,55	1,06	0,79	0,66
22050	10,38	5,70	3,43	2,13	1,46	1,10	0,93
24000	11,28	6,22	3,73	2,33	1,58	1,19	1,00
32000	15,01	8,25	4,98	3,11	2,11	1,60	1,33
44100	20,60	11,37	6,84	4,27	2,90	2,20	1,85

Tabela 3.7.: Tempos de execução com modelo rms de janelas deslizantes e sem sobreposição

Fs(Hz)	Tamanho de janela (Número de amostras)						
	128	256	512	1024	2048	4096	8192
2000	0,72	0,40	0,25	0,16	0,11	0,08	0,07
4000	1,44	0,81	0,49	0,31	0,22	0,17	0,14
8000	2,87	1,63	0,99	0,62	0,43	0,33	0,28
11025	3,97	2,21	1,36	0,85	0,60	0,46	0,39
16000	5,77	3,21	1,97	1,23	0,86	0,67	0,56
22050	7,97	4,43	2,71	1,70	1,19	0,93	0,79
24000	8,65	4,83	2,96	1,86	1,28	1,00	0,84
32000	11,48	6,42	3,95	2,49	1,71	1,34	1,15
44100	15,80	8,85	5,44	3,43	2,38	1,86	1,58

Figura 3.5.: Segmento de áudio contendo assobios de boto cinza

Figura 3.6.: Fonte: o autor (2015)

3.4.3. Desempenho de envelopes como funções de detecção

Um segmento de áudio de 35s e $Fs = 22.050Hz$, contendo assobios de boto cinza foi usado para avaliar o desempenho dos envelopes obtidos com as funções de Média móvel e rms, as quais se mostraram adequadas à detecção tempo quase-real de eventos de interesse.

Na Figura 3.5 é apresentado o espectrograma do sinal avaliado. Pode-se observar a presença de 6 eventos bem definidos (retângulos amarelos), na banda de $4,5kHz$ até $9,0kHz$. No mesmo espectrograma é possível apreciar alguns eventos de tipo impulsivo.

Para avaliar o desempenho dos métodos da Média móvel e o rms, foram construídos os envelopes do sinal em referência para todos os tamanhos de janela W considerados nos testes (128, 256, 512, 1024, 2048, 4096 e 8192). Os resultados para cada caso são apresentados na Figura 3.7 – (Envelopes obtidos com o método da média móvel para diferentes tamanhos de janela W) e Figura 3.9 para o rms.

Os envelopes obtidos tanto na Média móvel quanto no rms apresentam o mesmo formato, sendo possível, em cada um deles, identificar claramente as ocorrências dos eventos contidos no sinal analisado.

Pode-se observar também que, à medida que o tamanho da janela W aumenta, a presença de eventos impulsivos tende a diminuir, obtendo-se um sinal mais suave e com melhor percepção da presença dos eventos de interesse.

A pesar dos desempenhos dos dois métodos avaliados serem similares, quando analisado o formato do envelope obtido, os tempos de processamento da Média móvel aumentam com o acréscimo do tamanho da janela utilizada (Tabela 3.3). Já o rms apresenta um comportamento inverso, isto é, um aumento na janela W implica em menores tempos de processamento computacional (Tabelas 3.4 a 3.7).

3.5. CONCLUSÕES

Foi demonstrado que a RPI possui a capacidade de realizar tarefas de aquisição de dados e processamento de sinais em paralelo, visando detecção de eventos acústicos, em arquivos de áudio de diferentes F_s , entre 2kHz y 44.1kHz.

Os testes mostraram que, o método da média móvel e o rms deslizante apresentam melhor desempenho no processando de segmentos de áudio de 60s, viabilizando assim seu uso como geradores de funções de detecção.

De acordo com o comportamento observado em ambos os métodos de cálculo do envelope, e visando processamento em tempo real, a melhor estratégia seria o uso preferencial deles, usando a Média móvel nos casos de análise que requeiram janelas menores que 256 pontos, e o Rms nos casos de janelas maiores que 256 pontos.

A criação de funções de detecção é o primeiro passo para o desenvolvimento de um detector automático; por conseguinte, esta experiência pode ser estendida usando-se um valor limiar para extrair os eventos e posteriormente segmentando-se o áudio, concomitantemente com as tarefas de aquisição de dados.

Nesta experiência computacional, o tamanho do segmento de áudio processado foi de 60s, a fim de se ajustar algoritmos para seu uso nas unidades OceanPod desenvolvidas no LADIN. Tamanhos de arquivo de áudio menores poderiam ser usados, com o intuito de calcular qual o tamanho mínimo possível de processar, visando aquisição e detecção em tempo quase-real.

Figura 3.7.: Envelopes obtidos com o método da média móvel para diferentes tamanhos de janela W

Figura 3.8.: Fonte: o autor (2015)

Figura 3.9.: Envelopes obtidos com o método rms para diferentes tamanhos de janela W e sobreposição de 75%

Figura 3.10.: Fonte: o autor (2015)

4. DESENVOLVIMENTO DE MÉTODOS DE ANÁLISE

A detecção e identificação de eventos acústicos em PAM submarino, focada no estudo de mamíferos marinhos, é uma área de pesquisa bastante ativa, motivada em grande parte pelo efeito negativo de atividades antropogênicas, as quais são cada vez mais frequentes nos oceanos.

Na revisão bibliográfica feita, os métodos e algoritmos de detecção e ou classificação foram agrupados, dependendo dos contextos de utilização e dos princípios matemáticos subjacentes para cada caso, na hierarquia apresentada a seguir:

- Detectores em tempo real
- Detectores baseados em energia e amplitude do sinal
- Detectores baseados em correlação
- Detectores baseados em extração de características
 - Detecção e ou classificação com métodos estatísticos
 - * Modelos de mistura de Gaussianas
 - * Modelos ocultos de Markov
 - Redes neurais
 - Técnicas de análise e processamento de voz humana

Vale a pena salientar que apesar da anterior classificação, muitos dos métodos citados podem ser usados de forma conjunta, visando melhorias no desempenho das detecções para casos particulares, fato que será evidenciado no desenvolvimento do capítulo.

4.1. DETETORES EM TEMPO REAL

Detecção em tempo real é possível dependendo do projeto e das capacidades dos equipamentos de PAM, assim como da natureza do evento a ser detectado (p. ex. uma espécie

em particular); feita de forma analógica ou digital.

Quando a detecção é feita de forma analógica, a técnica comum consiste em processar o sinal que vem do hidrofone, com um filtro passa-banda, e monitorar os níveis da amplitude do sinal filtrado, comparando-o com um determinado valor ou limiar definido a priori. As detecções positivas são armazenadas ou transmitidas para outras análises ou processamentos posteriores.

Na detecção ou classificação digital, a energia do sinal é monitorada para uma banda de frequência específica (de forma similar ao caso analógico), e naqueles casos onde a energia supera o limiar, o evento é determinado como positivo, armazenado e/ou transmitido para posteriores análises.

Há alguns equipamentos que fazem detecção em tempo real tais como:

- T-POD (Timed POrpoised Detector) (Philpott et al., 2007), planejado para detectar toninhas (*Phocoena phocoena*) através dos cliques emitidos por elas.
- C-POD (Cetacean POrpoise Detector) (Limited, 2011), o sucessor do T-POD, capaz de detectar cliques de uma grande variedade de cetáceos, com exceção do cachalote (sperm whale) devido a limitações na banda de frequência do equipamento.
- PAM-Buoy (Gillespie et al., 2013), que possui o mesmo sistema de identificação e classificação usado no software PAMGUARD (Gillespie et al., 2008).

Detecção em tempo real nem sempre é possível devido à disponibilidade de recursos como energia permanente, facilidades de acesso ao local de instalação e infraestrutura de comunicações para transmissão de dados; como consequência, nas campanhas de monitoramento são gerados grandes volumes de dados para serem analisados após a sua recuperação.

4.2. DETECTORES BASEADOS EM ENERGIA E AMPLITUDE DO SINAL

Em detectores baseados em energia ou amplitude do sinal, podem ser impostas restrições no tempo de duração do evento, no tempo entre eventos a serem detectados, e bandas de frequência a serem analisadas, além de diversas formas de cálculo da energia do sinal.

Como exemplo de detectores baseados em energia pode-se citar o sistema de detecção de toninhas (*Phocoena phocoena*), no qual os sinais procedentes dos hidrofones são segmentados em três sinais com bandas de frequência diferentes. As informações das

amplitudes relativas dos sinais obtidos, assim como do formato dos pulsos detectados em cada um dos três sinais calculados, permitem distinguir quais pulsos são produzidos pelas toninhas e quais não (Gillespie and Chappell, 2002).

De forma similar, em (Harland and Armstrong, 2004) o sinal original é decomposto em 5 bandas diferentes (filtro passa-banda), cada uma delas associada a uma espécie em particular, entre odontocetos e mysticetos. Para a detecção, é usado um limiar na amplitude dos sinais filtrados e, em combinação com parâmetros tais como frequências mínimas e máximas dos eventos, frequência de início e finalização, duração do evento, largura de banda do evento, e outros parâmetros extraídos do espectrograma do evento, é feita uma classificação dos eventos detectados.

Em (Ward et al., 2000) foi implementado um detector de energia focado na detecção de cachalotes (*Physeter macrocephalus*), que emite cliques entre 8kHz e 26 kHz (Madsen et al., 2002).

Em (Ma et al., 2010) o envelope calculado baseia-se na energia do sinal usando o operador de energia Teager-Kaiser, comparado posteriormente com um threshold e, junto com a duração das vocalizações e os tempos entre cliques, determinadas as ocorrências dos eventos procurados.

Em (Zaugg et al., 2012) foi definida uma métrica para os picos espectrais em espectrogramas do sinal analisado, feito por bandas específicas do sinal e a partir do qual é construído um envelope, que ao ser comparado com um limiar pre-definido, permite identificar os possíveis eventos.

Pode-se observar, através dos exemplos de detectores de energia apresentados, que estes podem ser usados junto com outras técnicas, (que são discutidas posteriormente neste capítulo), para melhorar a detecção ou permitir fazer classificação dos eventos acústicos.

Esses tipos de detectores estão implementados nos softwares de PAM mais populares como o Ishmael (Mellinger, 2002), Panguard (Gillespie et al., 2008), Raven Pro (of Ornithology, 2010), e o Xbat (Figuerola, 2007), Avisoft (Bioacoustics, 2015) entre outros.

4.3. DETETORES BASEADOS EM CORRELAÇÃO

Nos detectores baseados em correlação, um sinal conhecido ou padrão, também chamado de template ou kernel, é correlacionado com o sinal analisado (cross-correlation), gerando um novo sinal que evidencia a presença de eventos semelhantes ao padrão (função de detecção (Mellinger and Clark, 1997)).

Quando os detectores baseados em correlação usam um segmento de áudio, e a função de detecção é calculada no sinal no domínio amplitude-tempo, é chamado de filtro adaptado ou *Matched Filter*.

A correlação também é usada no domínio frequência – tempo; neste caso, o sinal analisado é transformado em um espectrograma, e é correlacionado com um outro espectrograma menor que representa o evento a ser detectado. É uma técnica vinda da análise de sinais e comumente conhecida como *Template Matching*.

Detectores baseados em correlação têm sido usados com sucesso na detecção de cetáceos, como p. ex. baleia jubarte (Abbot et al., 2010), o cachalote (Kandia and Stylianou, 2006), baleia-da-groenlândia (Mellinger and Clark, 1997), baleia-azul (Stafford et al., 1998).

Estes detectores também estão implementados nos softwares de PAM mencionados anteriormente.

4.4. DETECTORES BASEADOS EM EXTRAÇÃO DE CARACTERÍSTICAS

Nesta categoria temos os métodos que, além da detecção, são capazes de fazer uma classificação dos eventos detectados, em grupos relacionados com uma espécie em particular.

O quesito neste tipo de métodos é um vetor de características extraídas do evento detectado, o qual é analisado e, de acordo com um cálculo probabilístico é julgado como pertencente ou não a uma determinada espécie.

Dentre as técnicas usadas para analisar os vetores de características dos eventos têm-se: os puramente estatísticos como, por exemplo, a análise de componentes principais (PCA), análise e classificação com árvores de regressão (CART), análise discriminante (DFA), análise discriminante linear (LDA) ou quadrático (QDA), análise de agrupamentos (Clusters), modelos de mistura Gaussianas (GMMs) e modelos ocultos de Markov (HMM).

Podem-se citar como exemplos de detecção com métodos estatísticos os trabalhos desenvolvidos por:

- (Oswald et al., 2003), que usa DFA para classificação de assobios em nove espécies diferentes
- (Oswald et al., 2007a), CART e DFA são usados para identificar espécies através das vocalizações detectadas previamente com o software Ishmael (MELLINGER,

2002). Resultado desse trabalho é o algoritmo ROCCA (“Real-time Odontocete Call Classification Algorithm” (Oswald et al., 2007b) implementado no software Pamguard (Gillespie et al., 2008).

- (Parnum et al., 2011), LDA redução dimensional dos vetores de características em cliques de baleia-bicuda e posterior classificação.
- (Baumgartner and Mussoline, 2011), QDA para classificação de chamados (“calls”) da baleia-franca-do-atlântico-norte (*Eubalaena glacialis*).
- (Rickwood and Taylor, 2008), detecção automática e caracterização automática de vocalizações de baleia jubarte (*megaptera novaenvliae*) usando HMM.
- (Brown and Smaragdis, 2009), classificação automática de vocalizações de orca (*Orcinus orca*) usando HMM e GMMs.

Redes neurais artificiais (ANNs) também têm sido usadas com sucesso para detecção de vocalizações de baleias em ambientes com ruído antrópico, p. ex.:

- Em (Erbe, 2000) sons da baleia beluga (*Delphinapterus leucas*) foram misturados com vários tipos de ruído e depois detectados usando ANNs e comparados com detecção feita por humanos.
- Em (Mercado and Kuh, 1998), é usada uma ANNs auto organizativa para classificação de vocalizações de baleia jubarte (*megaptera novaengliae*).
- Em (Zaugg et al., 2010), uma ANN tipo “Feedforward” é usada para detectar automaticamente cliques de cachalote (*Physeter macrocephalus*), em eventos detectados previamente com banco de filtros e limiar dinâmico

Metodologias usadas em análise e processamento de voz humana também têm sido implementadas para detecção em PAM submarino. Em (Clemins et al., 2006) a função de coeficientes cepstrais (GFCC – Greenwood Funtion Cepstral Coefficients) e percepção linear preditiva generalizada (GPLP – Generalized Perceptual Linear Prediction) são usados para extração de modelos para análise de vocalizações de várias espécies.

Discussões mais abrangentes no âmbito das técnicas, algoritmos e espécies, podem ser encontrados em (Au and Hastings, 2008; Bittle and Duncan, 2013; Zimmer, 2011).

4.5. SISTEMAS DE DETECÇÃO DE EVENTOS BIO-ACÚSTICOS

Dentre os sistemas de detecção de eventos acústicos de origem biológica, estão aqueles orientados para processar sinais vindos de uma espécie animal em particular, e aqueles que oferecem ferramentas genéricas para a análise e tratamento dos sinais.

A duas abordagens possuem vantagens e desvantagens de acordo com o tipo de usuário final. Assim, usuários com um bom conhecimento em análise digital de sinais (DSP) preferem o uso de software multipropósito, como p. ex. Matlab (www.mathworks.com), R (www.r-project.org) ou Python (www.python.org), e até mesmo o Xbat (Charif and Pitzrick, 2008)

Já usuários com menor destreza em *DSP* dão preferência a programas especializados, que apresentam facilidades de uso e configuração de detectores. A seguir são citados alguns dos softwares normalmente usados em pesquisas deste tipo:

Software especializado em bio-acústica:

- Avisoft
- Pamguard
- Raven
- Xbat

Software para visualização e manipulação de áudio

- Audacity
- Adobe Audition
- Goldwave

Um dos maiores problemas no processamento e análise de dados coletados em PAM submarino é o grande volume de dados gerados. A título de exemplo, um monitoramento realizado com uma taxa de amostragem de $32kHz$ (suficiente para monitorar uma banda de frequência que permita a detecção de odontocetos (p.ex. golfinhos) durante $24hrs$, por 3 meses gera, por unidade de gravação, $0,5TB$ de dados.

Processar e analisar esses sinais requer atividades como visualização, através de análise tempo-frequência, cálculo do nível de pressão sonora (SPL do termo em inglês), filtragem, detecção e identificação de eventos.

Apresentam-se na sequência três contribuições que procuram mitigar o esforço de análise de grandes bancos de sinais: A primeira é o desenvolvimento de algoritmo otimizado e software para visualização exploratória (tempo-frequência e SPL) em grandes janelas temporais (horas, dias ou meses).

A segunda contribuição é a proposta de uma nova metodologia para detecção de eventos acústicos usando o conceito estatístico de “outliers” (valores extremos). Por fim, a terceira é um método de remoção de ruído na representação tempo-frequência do sinal acústico (espectrogramas) contendo eventos acústicos segmentados.

4.6. BANCO DE DADOS

Com o objetivo de testar o desempenho das ferramentas de processamento de sinais propostas, é utilizado um banco de dados gerados ao longo dos últimos 3 anos com equipamentos desenvolvidos no Ladin, OceanPod (Caldas-Morgan et al., 2015). Ele é composto por gravações das campanhas feitas em Ilhéus 4.1. No primeiro caso, os sons de tipo biológico dominantes nas gravações são produzidos por baleias do tipo Jubarte (*Balaenoptera Engliae*).

Figura 4.1.: Locais de instalação das unidades de Oceanpod no litoral de Ilhéus (BA)

(a) Fonte: o autor (2015)

O formato dos arquivos de áudio é o padrão PCM-WAV . Na Tabela 4.1 são apresentadas as características do conjuntos de dados usados nos testes de detecção. A Fs dos arquivos é 11.025 kHz, a qual foi julgada adequada à faixa de frequência característica das vocalizações da baleia jubarte, situada abaixo dos 4kHz (Tasker, 2000).

Tabela 4.1.: Bando de dados de PAM submarino usado nos testes

	Fs(Hz)	Bits	Nro. de Arquivos	Tempo de gravação (min)
Ilhéus (BA)	11.025	16	9	135

4.7. EMBASAMENTO TEÓRICO

Apresentam-se a seguir os principais conceitos de processamento de sinais e estatística que serão utilizados no desenvolvimento das contribuições mencionadas no item anterior.

4.7.1. Espectrograma – representação tempo-frequência de um sinal

O espectrograma é uma representação tempo-frequência de um sinal utilizando o conceito da Transformada de Fourier de Curto Termo, STFT (Short Time Fourier Transform do termo em inglês)(Mallawaarachchi et al., 2008). O sinal é dividido em um certo número de janelas temporais Nj de tamanho especificado J . Para uma certa janela temporal, a Densidade Espectral de Potência (DEP) é calculada. Desta forma tem-se a matriz $X[n, k]$ de dimensões $J/2 \times Nj$ que representa o espectrograma do sinal. Quando essa matriz é representada num gráfico tridimensional, tem-se a visualização chamado de espectrograma.

$$X[n, k] = \sum_{n=1}^{J-1} x[n]N[n - m]exp(-iwn) \quad (4.1)$$

4.7.2. Filtragem do sinal

Sinais analógicos reais estão sujeitos a ruídos e perturbações aleatórias que podem dificultar os procedimentos de análise e identificação de eventos de interesse. Isso também acontece em sinais acústicos submarinos. Para contornar estes problemas, uma prática comum é a utilização de filtros, cujo objetivo é conservar somente os sinais de interesse presentes no sinal original. Filtrar um sinal é deixar passar a informação de interesse enquanto a informação não desejada é bloqueada. Os filtros também podem ser usados para atenuar ou diminuir a influência do ruído.

Figura 4.3.: Curvas de resposta em frequência de filtros

Filtros de resposta em frequência

Há basicamente quatro tipos de filtros de acordo com a forma de sua curva de resposta em frequência deles. Filtros passa-baixa, passa-alta, passa-banda e rejeita-banda (Figura 4.3). Basicamente, os nomes indicam a faixa de frequência (banda passante), onde se espera que a magnitude da resposta de frequência do filtro seja igual à unidade (numa escala linear) ou 0 dB (em escala logarítmica) (Giannakopoulos and Pikrakis, 2014).

Existem diversas bibliotecas de implementações digitais desses filtros, tanto em software de uso geral para programação (c, c++, java, etc.) quanto em softwares de computação científica como o R, Matlab, Python, etc. Também estão presentes em softwares de manipulação de áudio de uso geral como o Audacity, Goldwave, Adobe Audition, Sox, etc., ou especializados como o Raven, Avisoft, Ishmael, e Pampguard entre outros. Neste trabalho são usadas as implementações existentes no Matlab e no Sox.

Filtro móvel estatístico – de Média e de Medianas

O conceito de filtro estatístico móvel usa uma janela móvel de J amostras para calcular a média dessa sequência de dados.

A representação discreta de um filtro do tipo média-móvel é

$$y(k) = y(k-1) + \frac{u(k) - u(k-J)}{J}, \quad (4.2)$$

Onde J é o tamanho da janela móvel, $u(k)$ é o valor medido ou entrada do filtro e $y(k)$ é a saída filtrada.

O filtro de mediana é constituído por uma janela deslizante abrangendo um número N ímpar de amostras. A mediana de uma sequência discreta $a_1, a_2, \dots, a_n$ para N ímpar é o membro a_i da sequência para a qual $(N-1)/2$ elementos são menores ou iguais em valor a_i e $(N-1)/2$ são elementos cujo valor é maior ou igual a a_i (Pratt, 2007).

Filtro de Medianas 2D

Em processamento de imagens, quando um filtro de medianas é aplicado, uma janela desloca-se ao longo da imagem, e a mediana da intensidade dos pixels (elementos constitutivos da imagem) dentro da janela substitui a intensidade dos pixels que estão sendo processados (Smith, 1997). O efeito é a suavização do ruído de tipo impulsivo no sinal original.

4.7.3. Outliers e Valores Extremos

O boxplot (gráfico de caixa ou bigodes) é uma ferramenta de análise estatística exploratória que permite avaliar a distribuição de um conjunto de dados (Figura 4.4). O boxplot é formado pelo primeiro e terceiro quartil e pela mediana (Q_1 , Q_3 e $\bar{x}$), quando os dados possuem distribuição normal. Neste caso, a mediana e a média são as mesmas. As hastes inferior e superior se estendem, respectivamente, do quartil inferior (Q_1) até o menor valor não inferior ao limite inferior (bigode inferior), e do quartil superior (Q_3) até o maior valor não superior ao limite superior (bigode superior). Os limites do boxplot são calculados da forma abaixo:

$$\text{Limite inferior } L_{inf} = Q_1 - 1,5(Q_3 - Q_1) \quad (4.3)$$

$$\text{Limite superior } L_{sup} = Q_3 + 1,5(Q_3 - Q_1) \quad (4.4)$$

As amostras com valores maiores que L_{sup} e menores que L_{inf} são chamadas de valores atípicos ou “outliers”. Outliers são definidos como valores ou observações que se encontram a uma distância anormal de outros valores em uma amostra aleatória de

Figura 4.4.: Boxplot para dados com distribuição normal

Fonte: O autor usando o software R. Valores normalmente distribuídos com media = 0 e desvio padrão = 1.

uma população (Natrella, 2010).

Os limites definidos anteriormente são válidos para dados que apresentam distribuição normal. No caso de dados onde isso não é válido ou não se tem certeza quanto à normalidade da distribuição, o ideal é usar um método robusto o suficiente na detecção de outliers e valores extremos, que não seja afetado quando a distribuição dos dados não é normal.

Hubert e Vandervieren (Hubert and Vandervieren, 2008) propõem um ajuste ao boxplot que permite identificação de outliers, robusto o suficiente inclusive para distribuições com caudas alongadas (assimetria ou skewness). No desenvolvimento do boxplot ajustado é usada uma medida robusta de assimetria chamada “medcouple” (MC) introduzida por Aucremanne et al (Aucremanne et al., 2004). No caso de dados univariados $x_1, x_2, \dots, x_n$, o parâmetro MC é definido como:

$$MC = x_i \leq Q_2 \leq x_j^{h(x_i, x_j)} \quad (4.5)$$

Onde Q_2 é a mediana, para todos os valores de $x_i \neq x_j$ e a função de h é definida por:

$$h(x_i, x_j) = \frac{(x_j - Q_2) - (Q_2 - x_i)}{x_i - x_j} \quad (4.6)$$

Assim, quando $MC \geq 0$, todas as amostras com valores fora do intervalo são identificadas como possíveis outliers:

$$[Q_1 - 1, 5e^{-4MC} IQR; Q_3 + 1, 5e^{-3MC} IQR] \quad (4.7)$$

Onde IQR é a distância interquartil $Q_3 - Q_1$, como indicado na figura 4.4.

Da mesma forma, quando $MC \leq 0$, todas as amostras com valores fora do intervalo são identificadas como possíveis outliers:

$$[Q_1 - 1, 5e^{-3C} IQR; Q_3 + 1, 5e^{-4MC} IQR] \quad (4.8)$$

Figura 4.5.: Ilustração do box-plot ajustado para dados com assimetria. d_1 e d_2 são os bigodes quando há suposição de normalidade nos dados. s_1 e s_2 são os bigodes de dados com assimetria. Pode-se observar que, de acordo com a metodologia selecionada de cálculo, o ponto x_2 pode ser identificado ou não como sendo outlier.

Fonte: (HUBERT; VEEKEN, 2010)

4.8. PROCESSAMENTO E VISUALIZAÇÃO DE GRANDES VOLUMES DE DADOS

Durante a análise de sinais acústicos resultantes de campanhas de monitoramento, uma das primeiras tarefas a ser realizada é a análise exploratória dos dados, visando avaliar sua integridade ou identificar o comportamento geral da paisagem acústica e que possam orientar a análise posterior.

Neste contexto, as ferramentas básicas utilizadas são o espectrograma e o SPL em função do tempo. No entanto, em função do grande volume de dados, o custo computacional pode ser alto, comprometendo dessa forma uma análise exploratória rápida e ágil do banco de dados.

Poucas soluções para essa questão são encontradas na literatura. Um exemplo é o software Triton (Scripps Whale Acoustic Lab., 2007), desenvolvido por Scripps Whale Acoustic Lab. na Universidade de San Diego, California. Esse software permite a criação e visualização de um espectrograma para grandes volumes de dados ou LTSA (do inglês Long Term Spectrogram Average). No entanto ele supõe uma estratégia que visa solucionar esse único problema de visualização do espectrograma, deixando do lado outras ferramentas como o SPL para grandes volumes de dados (LTSPL).

Outros softwares como o Xbat (Figueroa, 2007), ou Ishmael (Mellinger, 2002), permitem processar grandes volumes de dados. No entanto o objetivo é a detecção de eventos de curta duração, não permitindo uma visualização do panorama por longo tempo em gravações contínuas e de grande duração.

No sentido de contribuir com uma solução adicional para esse problema, propõe-se um algoritmo que adote a estratégia de segmentação do áudio, onde pequenos segmentos de áudio são carregados em memória e posteriormente processados. Esse processo de segmentação é iterativo e somente finaliza após o processamento da totalidade dos dados do(s) arquivo(s) de áudio escolhidos.

Os parâmetros calculados para cada segmento de áudio são espectro de potência e o nível de pressão sonora ou SPL, os quais são armazenados numa matriz de resultados para posterior visualização mediante representação gráfica (LTSA e LTSPL).

O algoritmo desenvolvido é apresentado no pseudocódigo da Figura 4.6, e é detalhado na sequência:

1. Início
2. Leitura dos arquivos a processar. São selecionados um conjunto específico de arquivos ou uma pasta inteira para ser processada. Os arquivos devem ser formatados no padrão PCM-wav, de um único canal.
3. Segmentação do áudio. O tamanho de segmento define o número de pontos do sinal original a serem usados para calcular o valor instantâneo do SPL e a densidade espectral de potência - DEP. Com o tamanho do segmento e o conhecimento da duração da totalidade dos arquivos a serem processados, são calculados os tamanhos da matriz e vetor para armazenar os dados do espectrograma e SPL respectivamente.
4. É iniciado o laço “PARA” encarregado de processar cada arquivo contido na lista criada na linha 2 do pseudocódigo.
5. É calculado o número de segmentos necessários para processar o arquivo “i”
6. É iniciado o laço “PARA” encarregado de controlar os segmento de áudio a serem

- processados do arquivo “i”.
7. É carregado em memória o segmento de áudio “j” do arquivo “i” (Dataset).
 8. É calculado SPL do “Dataset”
 9. É calculado o espectro de potência do “Dataset”
 10. O valor calculado do SPL é armazenado no vetor de resultados criado na linha 3 do pseudocódigo.
 11. O valor calculado do espectro de potência é armazenado na matriz criada na linha 3 do pseudocódigo.
 12. Finaliza o laço uma vez processados todos os segmentos “j” do arquivo “i”. É processado o próximo arquivo se houver.
 13. Finaliza o laço uma vez processados todos os arquivos.
 14. Imprime ou visualiza na tela o gráfico do SPL
 15. Imprime ou visualiza na tela o gráfico do espectrograma

Figura 4.6.: Pseudocódigo para o cálculo de espectrogramas e curvas de SPL para grandes volumes de dados

```

1      INICIO
2      |      Ler arquivos a processar
3      |      Ler tamanho de segmento
4      |      PARA i DE 1 ATÉ <Último arquivo> FAÇA
5      |          |      NumSegmentos <- Calcular número de segmentos por arquivo
6      |          |      PARA j DE 1 ATÉ <Último Segmento> FAÇA
7      |          |          |      Dataset <- Ler Segmento j do Arquivo i
8      |          |          |      SPL <- Calcular SPL
9      |          |          |      Espectro <- Calcular espectro de potência
10     |          |          |      Armazenar SPL(i,j)
11     |          |          |      Armazenar Espectro (i,j)
12     |          |      FIM PARA
13     |      FIM PARA
14     |      Imprimir gráfico SPL
15     |      Imprimir gráfico Espectrograma
16     FIM

```

Fonte: o autor (2015))

Esse algoritmo foi implementado em um software básico para criação de espectrogramas e curvas SPL, para grandes volumes de dados de PAM acústico submarino. O software é inteiramente desenvolvido no Matlab e pode rodar em Windows, Linux e MAC e possui uma interface bastante simples e intuitiva (Figura 4.7). É possível processar um arquivo individual, ou um conjunto de arquivos contidos numa determinada pasta.

Figura 4.7.: Tela da interface de usuário para cálculo do SPL e Espectrograma para grandes volumes de dados

Fonte: o autor (2015))

Nos ANEXOS D, E e F, encontram-se o código do algoritmos de detecção (EVD) assim como SPL e espectrograma para grandes volumes de dados respectivamente.

Na Figura 4.8 e Figura 4.9 são apresentados um exemplo do espectrograma e o SPL obtidos com o software desenvolvido. Nos exemplos, os gráficos correspondem a um período de 10 horas de gravação contínua, durante um dia de medições feitas com OceanPod no litoral da região de Ilhéus.

Figura 4.8.: Espectrograma de larga duração. (a) Vocalizações de baleias jubarte. (b) Ruído de embarcações com harmônicos. (c) Ruído de equipamento rotativo de alta energia não identificado. (d) Ruído de embarcações distantes poluindo a banda de frequência das vocalizações da baleia jubarte

Fonte: o autor (2015))

No espectrograma é possível observar a presença de vários eventos de origem antrópica (b,c,d) e eventos de origem biológica (a), enquanto o gráfico do SPL (Figura 4.9) permite visualizar o incremento significativo no nível de ruído, associado com o evento antrópico, relacionado ao evento na Figura 4.8-c.

Figura 4.9.: Curva de SPL para dados de larga duração

Fonte: o autor (2015))

A totalidade dos arquivos de medições de PAM submarino da campanha em Ilhéus (13.125 arquivos), equivalente a 3.281,25 horas de áudio, foi processado com o software de análise exploratório proposto.

Neste contexto, o processamento dos dados foi feito em um computador com 16GB de memória RAM e processador Intel I5. Os dados foram processados em grupos de 40 arquivos por dia, equivalentes a 10 horas de áudio. O tempo médio de processamento e obtenção do gráfico de espectrograma para cada 10 horas de áudio foi de 40 segundos, enquanto o gráfico do SPL foi obtido em tempo médio de 30 segundos.

O tempo de processamento para criação do espectrograma é comparável com o tempo obtido com o software Triton (Scripps Whale Acoustic Lab., 2007), que também é desenvolvido em Matlab e pode ser usado de forma livre. No entanto, no Triton não é possível calcular as curvas de SPL, incluídas na proposta deste projeto.

É importante notar que, embora o método proposto tenha sido desenvolvido e testado usando o Matlab, é simples o bastante para implementado em uma linguagem de uso livre como, p.ex. o Python. Dessa forma, poderá ser disponibilizado um aplicativo, de uso livre, multiplataforma e sem dependência de software que requer licença, como é o caso do Matlab.

4.9. ALGORITMO DE DETECÇÃO BASEADO EM EVENTOS EXTREMOS [EVD]

Apresenta-se neste tópico um algoritmo de remoção de ruído de espectrograma e um de detecção de eventos acústicos baseado em análise estatística. Ambos podem ser utilizados independentemente um do outro ou em sinergia.

O algoritmo apresentado esquematicamente na Figura 4.10 apresenta duas etapas distintas. Inicialmente é feita a detecção dos eventos propriamente ditos - passos (a) até (g)-, e posteriormente, é feita a remoção de ruído no espectrograma de cada evento detectado – passos (h) até (k).

A principal característica do algoritmo de detecção é que nele é usado o conceito de outliers estatísticos (valores extremos) eq. 4.1 como critério de pré-seleção em cada coluna da matriz do espectrograma $X[\omega, n]$ eq. 4.2 Este algoritmo é nomeado EVD (Detecção baseada em Eventos Extremos).

Figura 4.10.: Representação esquemática do algoritmo EVD (a)-(g), e denoising (h)-(k)

Fonte: o autor (2015))

Descrição do algoritmo:

-
- (a) Leitura dos dados a serem processados. Nesta etapa os dados são carregados em memória, parâmetros como F_s e número de amostras do sinal são também calculados. A entrada é um arquivo de áudio, formatado como PCM–WAV de um canal.
 - (b) Filtragem do sinal. Se há conhecimento prévio da banda de frequência dos eventos a serem detectados, o sinal pode ser filtrado (passa alta, passa baixa ou passa banda), melhorando assim a relação sinal/ruído.
 - (c) Filtragem do sinal. Se há conhecimento prévio da presença de ruído distribuído uniformemente numa banda específica do sinal, este é atenuado usando-se um filtro rejeita banda.
O uso deste tipo de filtro faz com que a banda de frequência que se encontra poluída adquira uma intensidade próxima do ruído de fundo. Em consequência disso, eventos de maior intensidade que, se encontram mascarados nessa banda, aparecem de forma mais evidente com relação ao ruído de fundo, permitindo uma melhora na detecção dos eventos procurados.
 - (d) Construção do espectrograma do sinal. O espectrograma é calculado e uma matriz contendo os valores da DSP no plano tempo-frequência é criada para cálculos posteriores.
 - (e) Extração de valores extremos. É criada uma matriz das mesmas dimensões que o espectrograma calculado anteriormente. Nela são armazenados aqueles pontos identificados como outliers ou valores extremos no espectrograma original. O cálculo é feito coluna por coluna usando a eq. 4.7.
 - (f) Cálculo da função de detecção. Usando a matriz de valores extremos, a função de detecção é calculada somando cada coluna da matriz.
 - (g) Detecção de eventos. A função de detecção é comparada com um valor limiar. Aqueles segmentos onde o sinal é maior que o limiar definido, são identificados como possíveis eventos de interesse.
 - (h) Segmentação do áudio. O áudio é segmentado em trechos que contêm o evento detectado previamente.
 - (i) Extração de contorno do evento. Para cada evento é calculado seu espectrograma e os valores extremos são extraídos usando-se a eq. 4.7; mas desta vez usando-se todos os pontos da matriz do espectrograma e não coluna por coluna como feito no passo (e) do algoritmo.
 - (j) Ao considerar todos os valores da matriz do espectrograma, aumenta-se o tamanho da amostra, e por conseguinte, aumenta-se o nível de confiança no cálculo dos valores extremos. Como resultado, obtém-se um espectrograma contendo o contorno

do evento definido pelos valores extremos.

- (k) Aprimoramento do contorno. O contorno obtido no passo anterior, é processado usando-se um filtro de medianas. Como resultado o contorno é definido de forma mais precisa com menor presença de pontos isolados na imagem.
- (l) Aprimoramento do contorno. O contorno obtido no passo anterior, é processado usando um filtro de medianas. Como resultado, o contorno é definido de forma mais limpa com menor presença de pontos isolados na imagem.
- (m) Armazenamento da matriz do evento. O resultado do passo j é armazenado para uso em posteriores processos de extração de parâmetros com intuito de identificação e ou classificação.
- (n)

O algoritmo detecção de eventos extremos - EVD requer que alguns parâmetros sejam definidos, a priori e empiricamente, para funcionar corretamente, são estes:

- Tamanho da janela para cálculo da transformada
- Filtros e valores dos filtros
- Valor do limiar ou método alternativo (p.ex. média móvel)

Esses parâmetros dependem das características próprias dos eventos de interesse tais como a duração e conteúdo frequencial (banda de frequência específica).

4.10. Testes e resultados

Inicialmente, nos capítulos 4.10.1 e 4.10.2, foi apresentado o funcionamento do algoritmo de detecção baseado em valores extremos (4.9), para dois cenários diferentes (Figura 4.11) descritos a seguir:

- (a) Cenário 1: O sinal de entrada, um segmento de áudio de 12 segundos, contendo vocalizações de baleia jubarte, é filtrado para conter unicamente informação abaixo dos $1500Hz$; posteriormente é feita a detecção e remoção de ruído dos eventos detectados
- (b) Cenário 2: Um filtro adicional para atenuar ruído na banda de $300Hz$ e até $500Hz$ (rejeita banda) é aplicado; posteriormente é feita a detecção e remoção de ruído dos eventos detectados.

Figura 4.11.: Cenários de teste do Algoritmo de detecção EVD

(a) Pre-processando o sinal com filtro passa-baixa.

(b) Pre-processamento com filtro passa-baixa e rejeita-banda

Fonte: s

Com o segmento de áudio escolhido é possível testar os cenários estabelecidos previamente, já que o sinal contém ruído uniformemente distribuído na banda citada, . Fica evidente que deve existir um conhecimento prévio (análise exploratória) dos sinais a serem analisados, visando garantir uma detecção mais eficiente.

4.10.1. Resultados e discussão detecção cenário 1

Na Figura 4.13 são apresentados o espectrograma do sinal original, contendo vocalizações de baleia jubarte, assim como a função de detecção obtida com o algoritmo EVD (linhas vermelhas).

No caso apresentado, os eventos de interesse são vocalizações de baleia jubarte, que ocorrem em uma banda de frequência que se encontra poluída por ruído distribuído uniformemente entre os 300Hz e 500Hz aproximadamente (Figura 4.13-a).

Pode-se observar que, apesar desse ruído se sobrepor às vocalizações, estas últimas possuem uma maior intensidade, o que faz com que sejam visíveis no espectrograma (Figura 4.13-a).

Na Figura 4.13-b, a função de detecção obtida pelo algoritmo proposto é representada por uma linha vermelha. Na mesma figura, em cor azul, é apresentada uma média móvel calculada a partir da função de detecção. Essa média móvel funciona como limiar.

Os segmentos da função de detecção que superam o limiar são considerados eventos detectados. Na Figura 4.13-c o sinal quadrado (linha azul) indica o começo e final de cada evento. Essa informação é usada posteriormente para segmentação do arquivo de áudio original em porções de áudio contendo eventos isolados.

Figura 4.13.: (a) Espectrograma de segmento de sinal com presença de vocalizações. (b) Função de detecção (linha vermelha) e threshold calculado por média móvel (linha azul). (c) Eventos detectados (linha azul)

Fonte: o autor (2015))

No exemplo apresentado, o algoritmo de detecção EVD conseguiu detectar sete eventos. No entanto, e apesar da detecção ser coerente com o visualizado no espectrograma, há eventos que não foram detectados com sucesso (elipses amarelas na Figura 4.13-a), ou detectados quase no limite do limiar (quadrados amarelos Figura 4.13-a e b).

Esses eventos que não foram detectados ou detectados de forma “fraca”, apesar de estarem localizados numa banda de frequência diferente do ruído dominante, possuem intensidade ou nível de energia similar ao ruído dominante (ruído distribuído homogeneamente na banda de $300 - 500\text{Hz}$).

Essa característica de proximidade da intensidade desses eventos com relação ao ruído,

os classifica dentro dos intervalos definidos nas eq. 4.7 e 4.8 ou seja, não pertencem à categoria de outliers ou eventos extremos; portanto, não são considerados na construção da função de detecção.

Para melhorar o desempenho da detecção, uma alternativa é atenuar o ruído presente na banda de frequência de $300 - 500Hz$ até níveis similares ao ruído de fundo, dessa forma favorecendo aqueles eventos de menor intensidade presentes em outras bandas de frequência.

Esse procedimento e os resultados do processamento com atenuação da banda de $300 - 500Hz$ são apresentados no item a seguir.

4.10.2. Resultados e discussão detecção cenário 2

Como descrito no item anterior, no sinal de exemplo apresentado há um ruído uniformemente distribuído na banda de $300 - 500Hz$, que dificulta a detecção de eventos nessa banda de frequência.

No gráfico da DEP do sinal original, apresentado na Figura 4.14, pode-se observar que as frequências com maior intensidade correspondente à banda mencionada anteriormente, têm uma potência aproximada de $102.5dB$ rel $1V\mu Pa$, enquanto que o ruído de fundo está próximo dos $95dB$ rel $1V\mu Pa$.

Foi aplicado um filtro rejeita-banda para atenuar o sinal em aproximadamente $7dB$ na faixa de $300Hz$ até $500Hz$. Na Figura 4.15 são apresentadas as curvas de DSP, antes (linha preta) e depois da aplicação do filtro (linha vermelha). Pode-se observar que a potência na banda filtrada (em amarelo) apresenta valores mais próximos do ruído de fundo (entre $95Hz$ e $100Hz$).

Figura 4.14.: Espectro de potência para o segmento de áudio analisado na Figura 4.13

Fonte: o autor (2015))

Figura 4.15.: Espectro de potência do sinal analisado antes (linha preta) e depois de filtragem (linha vermelha)

Fonte: o autor (2015))

O efeito imediato do filtro rejeita-banda aplicado anteriormente pode ser observado no espectrograma da Figura 4.16-a. Os eventos acústicos ficam mais evidentes e o ruído de fundo agora é mais homogêneo.

Como consequência, o algoritmo de detecção proposto, agora é capaz de detectar aqueles eventos que não foram detectados anteriormente (elipses amarelas), ou onde a distância entre a função de detecção e o limiar era mínima (quadrados amarelos). De forma quantitativa, no exemplo ilustrado, a detecção passou de 7 para 10 eventos.

Figura 4.16.: (a) Espectrograma do sinal após filtragem com equalização. (b) Função de detecção e limiar do sinal filtrado. (c) Eventos detectados (linha azul)

Fonte: o autor (2015))

Embora a detecção feita pelo algoritmo EVD seja do tipo não supervisionada, é possível condicioná-la de forma a aceitar ou rejeitar eventos sob certas restrições, como

Tabela 4.2.: Resultados comparativos do detector EVD e o EDet (Mellinger, 2002)

Detecções	Algoritmo EDet		Algoritmo EVD	
	Eventos	Porcentagem	Eventos	Porcentagem
Com sucesso	2.503	67,68	2.754	71,32
Falsos positivos	640	17,31	623	16,13
Falsos negativos	555	15,01	484	12,55
Total	3.698	100	3.861	100

por exemplo duração do evento ou distância entre eventos.

4.10.3. Desempenho do Algoritmo de detecção EVD

O desempenho do algoritmo EVD foi avaliado mediante comparação com detector de energia [EDet] implementado no Ishmael (Mellinger, 2002) e detecção feita por inspeção visual. Na detecção visual, considerada o padrão para as comparações, foram identificados 3.524 eventos.

Na Tabela 4.2 são apresentados os resultados quantitativos das detecções obtidas com cada um dos métodos comparados. Pode-se observar que o algoritmo EVD apresenta melhores resultados tanto na detecção de eventos com sucesso, assim como menores porcentagens de erros por falsos positivos e falsos negativos.

A eficácia dos detectores foi calculada como a razão entre os eventos detectados com sucesso e os eventos detectados por inspeção visual. O EVD foi ligeiramente melhor na detecção com 78,14% frente a 71,02% do detector de energia. A pequena margem poderia indicar que não há diferença significativa entre um método ou outro.

$$Desempenho_{EVD} = \frac{2.574}{3.254} \times 100 = 78,14\% \quad (4.9)$$

$$Desempenho_{EDet} = \frac{2.503}{3.254} \times 100 = 71,02\% \quad (4.10)$$

Até agora, os resultados das detecções usando EVD e EDet foram analisados isoladamente, como métodos competidores. No entanto, e com intuito de melhorar a detecção, os resultados de cada um desses métodos foram agrupados, e eventos duplicados foram descartados, obtendo assim um novo conjunto de resultados.

Na Tabela 4.3 são apresentados os eventos detectados com sucesso, para cada método. Na mesma tabela são apresentados os resultados da união das detecções do *EVD* e o

Tabela 4.3.: Eventos detectados com sucesso em cada um dos métodos, comparados isoladamente e em conjunto

Método	Dataset									Total
	1	2	3	4	5	6	7	8	9	
Visual	442	437	351	336	294	384	439	436	405	3.524
EVD	334	314	272	265	257	314	343	322	333	2.754
EDet	290	334	235	214	228	296	335	281	290	2.503
EVD + EDet	367	362	289	283	267	267	375	356	351	2.999
Percentagens (%)										
EVD	75,6	71,9	77,5	78,9	87,4	81,8	78,1	73,9	82,2	78.1
EDet	65,6	76,4	67,0	63,7	77,6	77,1	76,3	64,4	71,6	71.0
EVD + EDet	83,0	82,8	82,3	84,2	90,8	90,9	85,4	81,7	86,7	85.1

EDet, assim como a participação percentual de cada um deles.

Em termos gerais, para todos os casos houve um incremento na porcentagem de eventos detectados com sucesso (Tabela 4.3). No total consolidado, foram detectados 85,1% do total de eventos, frente a 78,1% do EVD e 71,0% do *EDet* obtidos anteriormente.

4.11. REMOÇÃO DE RUÍDO EM ESPECTROGRAMA DE EVENTOS ACÚSTICOS SEGMENTADOS

O conceito de valores extremos utilizado no algoritmo de detecção EVD pode também ser usado como método de remoção de ruído em espectrogramas de eventos acústicos segmentados.

Na Figura 4.18 é apresentado um exemplo de remoção de ruído no espectrograma de uma vocalização de baleia jubarte, a qual foi detectada e extraída do conjunto de dados analisado anteriormente.

No espectrograma é visualizada a presença de dois eventos relacionados com vocalização. Um na banda de $400 - 500Hz$ e de maior intensidade, e outra de menor intensidade, na banda de $100 - 250Hz$ (Figura 4.18-a). Igualmente pode-se observar a presença de ruído uniformemente distribuído na banda de $100Hz$ até quase $1.250Hz$.

Figura 4.18.: (a) Espectrograma de uma vocalização de baleia jubarte. (b) contorno extraído usando EVD e filtro de medianas 2D.

Fonte: o autor (2015))

Após a execução do algoritmo de remoção de ruído, na matriz do espectrograma do evento segmentado (Figura 4.18-b), são conservados unicamente os valores identificados como extremos, e que correspondem à vocalização contida no espectrograma inicial.

Na Figura 4.19 é apresentado o resultado do remoção de ruído, na forma de Boxplot e de gráfico de Densidade de probabilidade. Em ambas representações é possível observar como o método é capaz de diferenciar os valores extremos (áreas em amarelo), dos valores associados com o ruído de fundo e o ruído uniformemente distribuído explicado anteriormente, e, portanto, detectar as vocalizações do exemplo.

Figura 4.19.: Boxplot (a) e densidade de probabilidade (b) do espectrograma para o evento acústico da Figura 4.18. Em amarelo os valores extremos calculados

Fonte: o autor (2015))

4.12. CONCLUSÕES

Neste capítulo foi proposto um método alternativo para detecção de eventos bioacústicos em gravações de PAM submarino, baseado no conceito estatístico de outliers e valores extremos. A mesma abordagem estatística foi usada para realizar remoção de ruído em espectrogramas dos eventos detectados previamente.

O método proposto obteve sucesso na detecção de eventos com desempenho similar a um detector de energia implementado no Ishmael (Mellinger, 2002). No entanto, as porcentagens de falsos positivos e falsos negativos foram menores no método proposto.

Apesar do desempenho do detector EVD ter sido levemente superior, é necessário mencionar que, ajustando-se os parâmetros de cada um dos métodos, os resultados podem variar e ser mais precisos em um ou outro método. Isso acontece com todos os detectores existentes, e de forma geral, dependendo do cenário, uns funcionam melhor que outros. Por tanto, a proposta não deve ser considerada uma concorrente do método de energia usado para comparação, e sim um método alternativo que visa aumentar a confiabilidade na detecção.

O procedimento proposto pode ser usado para analisar não apenas vocalizações de

mamíferos marinhos, mas também espécies terrestres (mamíferos, insetos, pássaros, etc.).

Mais estudos são recomendados para incluir outros métodos de detecção de valores extremos que permitam determinar a viabilidade da abordagem proposta ou melhorar o desempenho em remoção e extração de contornos de vocalizações em espectrogramas.

Adicionalmente foi proposto um algoritmo para análise e visualização de grandes volumes de dados de PAM submarino, contribuindo assim para um tema com falta de opções na literatura. Ele apresentou um desempenho ótimo, obtendo relações de tempo de processamento vs. tempo do sinal de até de $9000\times$ para cálculo do espectrograma e $6000\times$ para o cálculo do SPL.

5. CONCLUSÕES GERAIS

Conforme estabelecido nos objetivos, neste trabalho foi desenvolvido um equipamento de monitoramento acústico passivo submarino, denominado OceanPod. O equipamento foi usado de forma satisfatória em condições reais de trabalho, e em vários locais ao longo do litoral brasileiro.

Graças à tecnologia empregada, o OceanPod pode ser facilmente adaptado a outras condições de operação como por exemplo em monitoramento acústico terrestre, simplesmente substituindo o hidrofone por um microfone e, opcionalmente, trocando o sistema de baterias por uma fonte permanente, como painéis solares ou conexão à rede elétrica.

Durante as jornadas de monitoramento realizadas, foram adquiridas gravações de acústica submarina em paisagens acústicas diferenciadas, permitindo a criação de um banco de dados acústico submarino com vários tipos de eventos como, por exemplo, sons de baleias (baleia jubarte), golfinhos (boto cinza), sirenios (peixe boi), embarcações de pequeno, mediano e grande porte, detonações, atividades de dragagem e vários outros não identificados.

Visando maior aproveitamento das capacidades do computador de placa única usada no projeto, foram avaliados vários métodos de criação de envelopes do sinal, que mostraram a viabilidade de se realizar detecção em tempo real usando a mesma tecnologia embarcada no OceanPod.

Foi proposto um método alternativo de detecção de eventos acústicos em gravações de PAM, baseado no conceito estatístico de outliers e/ou valores extremos. O método obteve resultados similares ao detector de eventos baseado em energia do sinal, o qual está implementado no software Ishmael e Panguard.

O mesmo conceito (valores extremos) foi usado como ferramenta de remoção de ruído em espectrogramas de eventos bioacústicos no processo de detecção, mostrando resultados satisfatórios na segmentação do espectrograma (favorecendo os contornos dos eventos de maior intensidade).

6. TRABALHOS FUTUROS

Visando um melhor aproveitamento das capacidades computacionais presentes no OceanPod, sugere-se:

Aumentar a capacidade do OceanPod como unidade de *MAP* multipropósito; em outras palavras, não limitar a aquisição a sinais acústicos unicamente. É possível conectar outros tipos de sensores para medir, por exemplo, temperatura e pressão e, já que esses parâmetros requerem uma F_s de poucos Hz , a capacidade de armazenamento e consumo energético não deveriam ser afetadas de forma substancial.

Implementação de algoritmos de detecção e identificação em tempo real para um ou vários tipos de eventos acústicos.

O banco de dados consolidado, possibilita análises que permitiriam extrair e caracterizar eventos acústicos, que poderiam ser utilizados como modelo ou “templates” em treinamento de redes neurais, ou sistemas de identificação e classificação automáticas.

Outro trabalho interessante seria o desenvolvimento de um software com algoritmos de detecção de eventos, mas elaborado inteiramente em Python, garantindo dessa forma a portabilidade a qualquer SO e sem as restrições de licenças, ou uso de linguagens mais complexas.

Referências Bibliográficas

- Abbot, T. a., Premus, V. E., and Abbot, P. a. (2010). A real-time method for autonomous passive acoustic detection-classification of humpback whales. *The Journal of the Acoustical Society of America*, 127(5):2894–2903.
- Au, W. W. L. and Hastings, M. C. (2008). *Principles of Marine Bioacoustics*. Springer-Verlag New York, 1 edition.
- Au, W. W. L., Mobley, J., Burgess, W. C., Lammers, M. O., and Nachtigall, P. E. (2000). SEASONAL AND DIURNAL TRENDS OF CHORUSING HUMPBAC WHALES WINTERING IN WATERS OFF WESTERN MAUI. *Marine Mammal Science*, 16(3):530–544.
- Aucremanne, L., Brys, G., Hubert, M., Rousseeuw, P. J., and Struyf, A. (2004). A Study of Belgian Inflation, Relative Prices and Nominal Rigidities using New Robust Measures of Skewness and Tail Weight. In Hubert, M., Pison, G., Struyf, A., and Van Aelst, S., editors, *Theory and Applications of Recent Robust Methods SE - 2*, Statistics for Industry and Technology, pages 13–25. Birkhäuser Basel.
- Baumgartner, M. F. and Mussoline, S. E. (2011). A generalized baleen whale call detection and classification system. *The Journal of the Acoustical Society of America*, 129(5):2889–2902.
- Binder, C. M. and Hines, P. (2012). Applying automatic aural classification to cetacean vocalizations. In *ECUA 2012 11th European Conference on Underwater Acoustics*, volume 17, page 070029, Edinburgh, Scotland.
- Bioacoustics, A. (2015). Avisoft-SASLab Pro.
- Bittle, M. and Duncan, A. (2013). A review of current marine mammal detection and classification algorithms for use in automated passive acoustic monitoring . (November).
- Brown, J. C. and Smaragdis, P. (2009). Hidden Markov and Gaussian mixture models for automatic call classification. *The Journal of the Acoustical Society of America*, 125(6):EL221–L224.
- Burgess, W. C. (2000). The bioacoustic probe: A general-purpose acoustic recording tag. *The Journal of the Acoustical Society of America*, 108(5):2583–2583.

- Caldas-Morgan, M., Alvarez-Rosario, A., and Rodrigues Padovese, L. (2015). An Autonomous Underwater Recorder Based on a Single Board Computer. *PloS one*, 10(6):e0130297.
- Charif, R. and Pitzrick, M. (2008). Automated detection of Cerulean Warbler songs using XBAT data template detector software. . . . *Research Program Cornell Laboratory of Ornithology*
- Chen, Z., Newsome, T., McKenzie, J., Batterham, P., Burtner, P., Qualls, C., Woollacott, M., Burgess, W., Tyack, P., Le Boeuf, B., and Costa, D. (1998). A programmable acoustic recording tag and first results from free-ranging northern elephant seals. *Deep Sea Research Part II: Topical Studies in Oceanography*, 45(7):1327–1351.
- Chiu, M.-H., Wang, C.-C., Liu, J.-Y., and Chang, C.-W. (2007). Design and Application of Autonomous Underwater Acoustic Recorder. In *Underwater Technology and Workshop on Scientific Use of Submarine Cables and Related Technologies, 2007. Symposium on*, pages 107–112.
- Clemins, P., Trawicki, M., Adi, K., Tao, J. T. J., and Johnson, M. (2006). Generalized Perceptual Features for Vocalization Analysis Across Multiple Species. *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, 1:253–256.
- Diaz Lopez, B. (2011). Whistle characteristics in free-ranging bottlenose dolphins (*Tursiops truncatus*) in the Mediterranean Sea: Influence of behaviour. *Mammalian Biology - Zeitschrift für Säugetierkunde*, 76(2):180–189.
- Dugan, P., a.N. Rice, Urazghildiiev, I., and Clark, C. (2010). North Atlantic Right Whale acoustic signal processing: Part I. comparison of machine learning recognition algorithms. *Applications and Technology Conference (LISAT), 2010 Long Island Systems*, pages 0–5.
- Erbe, C. (2000). Detection of whale calls in noise: performance comparison between a beluga whale, human listeners, and a neural network. *The Journal of the Acoustical Society of America*, 108(1):297–303.
- Figuerola, H. (2007). Xbat - extensible bioacoustic tool. *Manual*.
- Giannakopoulos, T. and Pikrakis, A. (2014). *Introduction to Audio Analysis*. Elsevier.
- Gillespie, D., Castellote, M., Maginnis, A., Garner, C., and Small, R. (2013). Real-time monitoring for beluga whales (*delphinapterus leucas*) in the eagle river, alaska using a pambuoy detection system. Technical report.
- Gillespie, D. and Chappell, O. (2002). An automatic system for detecting and classifying the vocalisations of harbour porpoises. *Bioacoustics*, 1(37):37–61.

- Gillespie, D., Mellinger, D., Gordon, J., McLaren, D., Redmond, P., McHugh, R., Trinder, P., Deng, X., and Thode, A. (2008). Pamguard: semiautomated, open source software for real-time acoustic detection and localisation of cetaceans. In *Proceedings of the Conference on Underwater Noise Measurement: Impact and Mitigation 2008, Southampton, UK, 14-15 Oct 2008*, volume 30 of *Proceedings of the Institute of Acoustics*, pages 54–62. Curran Associates, Red Hook, NY, USA.
- Harland, E. and Armstrong, M. (2004). The real-time detection of the calls of cetacean species. *Canadian Acoustics*, 32(2):76–82.
- Hubert, M. and Vandervieren, E. (2008). An adjusted boxplot for skewed distributions. *Computational Statistics and Data Analysis*, 52(12):5186–5201.
- Kandia, V. and Stylianou, Y. (2006). Detection of sperm whale clicks based on the Teager–Kaiser energy operator. *Applied Acoustics*, 67(11–12):1144–1163.
- Koay, T. B., Seeking, P. J., Chitre, M., Tan, S. P., and Hoffmann-Kuhnt, M. (2006). Advanced PANDA for high speed autonomous ambient noise data collection and boat tracking - system and results. In *OCEANS 2006 - Asia Pacific*, pages 1–9.
- Limited, C. (2011). C-pod user guide. Manual.
- Ma, T., Collison, N., Theriault, J., Hood, J., Pecknold, S., and Bougher, B. (2010). Detection of precise time events for marine mammal clicks. *Canadian Acoustics*, 38(3):30–31.
- Madsen, P. T., Payne, R., Kristiansen, N. U., Wahlberg, M., Kerr, I., and Møhl, B. (2002). Sperm whale sound production studied with ultrasound time/depth-recording tags. *Journal of Experimental Biology*, 205(13):1899–1906.
- Mallawaarachchi, A., Ong, S. H., Chitre, M., and Taylor, E. (2008). Spectrogram denoising and automated extraction of the fundamental frequency variation of dolphin whistles. *The Journal of the Acoustical Society of America*, 124(2):1159–1170.
- Mellinger, D. K. (2002). *ISHMAEL: Integrated system for holistic multi-channel acoustic exploration and localization*.
- Mellinger, D. K. (2004). A comparison of methods for detecting right whale calls. *Canadian Acoustics*, 32(2):55–65.
- Mellinger, D. K. and Clark, C. W. (1997). Methods for automatic detection of mysticete sounds. *Marine and Freshwater Behaviour and Physiology*, 29(1-4):163–181.
- Mellinger, D. K. and Clark, C. W. (2000). Recognizing transient low-frequency whale sounds by spectrogram correlation. *The Journal of the Acoustical Society of America*, 107(6):3518–3529.

- Mellinger, D. K., Martin, S. W., Morrissey, R. P., Thomas, L., and Yosco, J. J. (2011). A method for detecting whistles, moans, and other frequency contour sounds. *The Journal of the Acoustical Society of America*, 129(6):4055–4061.
- Mercado, E., I. and Kuh, A. (1998). Classification of humpback whale vocalizations using a self-organizing neural network. In *Neural Networks Proceedings, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference on*, volume 2, pages 1584–1589 vol.2.
- Nanaware, S., Shastri, R., Joshi, Y., and Das, A. (2014). Passive acoustic detection and classification of marine mammal vocalizations. In *Communications and Signal Processing (ICCSP), 2014 International Conference on*, pages 493–497.
- Natrella, M. (2010). *NIST SEMATECH eHandbook of Statistical Methods*. NIST SEMATECH.
- Neunaber, B. (2015). Improve your root-mean calculations.
- of Ornithology, C. L. (2010). Raven is a software program for the acquisition, visualization, measurement, and analysis of sounds.
- Oppenheim, A. V., Schafer, R. W., and Buck, J. R. (1999). *Discrete-time Signal Processing (2Nd Ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.
- Oswald, J. N., Barlow, J., and Norris, T. F. (2003). Acoustic Identification of Nine Delphinid Species in the Eastern Tropical Pacific Ocean. *Marine Mammal Science*, 19(1):20–37.
- Oswald, J. N., Rankin, S., Barlow, J., and Lammers, M. O. (2007a). A tool for real-time acoustic species identification of delphinid whistles. *The Journal of the Acoustical Society of America*, 122(1):587–595.
- Oswald, J. N., Rankin, S., Barlow, J., and Lammers, M. O. (2007b). A tool for real-time acoustic species identification of delphinid whistles. *The Journal of the Acoustical Society of America*, 122(1):587–595.
- Parnum, I., McCauley, R., Cato, D., Thomas, F., a.J. Duncan, and Johnson, M. (2011). Detection of beaked whale clicks in underwater noise recordings.pdf.
- Philpott, E., Englund, a., Ingram, S., and Rogan, E. (2007). Using T-PODs to investigate the echolocation of coastal bottlenose dolphins. *Journal of the Marine Biological Association of the UK*, 87(01):11.
- Pratt, W. K. (2007). *Digital Image Processing: PIKS Scientific Inside*. Wiley.
- Rickwood, P. and Taylor, A. (2008). Methods for automatically analyzing humpback song units. *The Journal of the Acoustical Society of America*, 123(3):1763–1772.

- Scripps Whale Acoustic Lab. (2007). Triton User Guide.
- Smith, S. W. (1997). *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Pub.
- Sousa-Lima, R. S., Norris, T. F., Oswald, J. N., and Fernandes, D. P. (2013). A review and inventory of fixed autonomous recorders for passive acoustic monitoring of marine mammals. *Aquatic Mammals*, 39(1):23–53.
- Stafford, K. M., Fox, C. G., and Clark, D. S. (1998). Long-range acoustic detection and localization of blue whale calls in the northeast Pacific Ocean. *The Journal of the Acoustical Society of America*, 104(6):3616–3625.
- S.V. Borisov, D.G. Kovzel, A. R. V. U. (2008). Transmitting autonomous underwater acoustic recorder. *XX Session of the Russian Acoustical Society*.
- Systems, M. I. (2015). Maxim integrated systems.
- Tasker, M. L. (2000). Marine mammals and noise, by w.j. richardson, c.r. greene, c.i. malme and d.h. thomson. academic press, london and san diego, 1998, 576 pp. price: £29.95. isbn 0 12588 441 9. *Aquatic Conservation: Marine and Freshwater Ecosystems*, 10(2):152–153.
- The MathWorks, I. (2015). Measure the Power of a Signal.
- Wang, C. C., Hsiao, Y. H., and Huang, M. C. (2009). Development of MSP430-based ultra-low power expandable underwater acoustic recorder. *Ocean Engineering*, 36(6-7):446–455.
- Ward, J., Fitzpatrick, M., DiMarzio, N., Moretti, D., and Morrissey, R. (2000). New algorithms for open ocean marine mammal monitoring. *OCEANS 2000 MTS/IEEE Conference and Exhibition. Conference Proceedings (Cat. No.00CH37158)*, 3:1749–1752.
- Wiggins, S. (2003). Autonomous acoustic recording packages (arps) for long-term monitoring of whale sounds. *Marine Technology Society Journal*, 37(2):13–22. n/a.
- Zaugg, S., Schaar, M. V. D., Houégnigan, L., and André, M. (2012). A framework for the automated real-time detection of short tonal sounds from ocean observatories. *Applied Acoustics*, 73(3):281–290.
- Zaugg, S., Van Der Schaar, M., Houégnigan, L., Gervaise, C., and André, M. (2010). Real-time acoustic classification of sperm whale clicks and shipping impulses from deep-sea observatories. *Applied Acoustics*, 71(11):1011–1019.
- Zimmer, W. M. X. (2011). *Passive acoustic monitoring of cetaceans*. Cambridge University Press.

A. Publicações e trabalhos apresentados em congresso internacional

Como autor principal

Título: Real-Time Performance of Energy Based Underwater Acoustic Event Detectors Embedded in a Single-Board Computer Autores: Alvarez-Rosario, A., Padovese, L.R. Congresso: IEEE Oceanic Engineering Society- IEEE/OES Acoustics in Underwater Geosciences Symposium, Rio Acoustics 2015. Data de apresentação: Agosto 30 de 2015

Como Coautor

Título: Sound Pressure Levels Measurements and Transmission Loss Models for Explosions in Coastal Waters Autores: Paro, A., Alvarez-Rosario, A., Padovese, L.R. Congresso: IEEE Oceanic Engineering Society- IEEE/OES Acoustics in Underwater Geosciences Symposium, Rio Acoustics 2015. Data de apresentação: Agosto 29 de 2015

Título: An Autonomous Underwater Recorder Based on a Single Board Computer Autores: Caldas-Morgan, M., Alvarez-Rosario, A., Padovese, L.R. Jornal: Plos One Data de publicação: 15 de Junho de 2015 DOI: 10.1371/journal.pone.0130297

B. Funções implementadas em Python para avaliar cada um dos métodos de cálculo de envelopes a partir de um sinal de áudio.

Script: Funtions.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# -----
# Bibliotecas a serem usadas pelas funções
# -----
import numpy as np
import scipy.signal as sig
import scipy.fftpack as fft
import matplotlib.mlab as mlab
# -----
#                               Funções implementadas
# -----
def fullRMS(interval, window_size):
    total_size = interval.size
    out = np.zeros(total_size)

    for i in np.arange(total_size):
        initial = max(0, i+1-(window_size/2))
        final = min(total_size, i+1+(window_size/2))
        step = final - initial
        val = np.dot(interval[initial:final],
            interval[initial:final])/float(step)
        out[i] = np.sqrt(val)

    return out
# -----

def fastRMS(interval, window_size):
```

```
total_size = interval.size
out = np.zeros(total_size)

lastRMS = 0
lastStep = 0
previus = 0
next = 0

for i in np.arange(total_size):
    initial = max(0, i+1-(window_size/2))
    final = min(total_size, i+1+(window_size/2))
    step = final - initial

    if (i == 0):
        val = np.sqrt(np.dot(interval[initial:final],
            interval[initial:final])/float(step))

    else:
        if initial > 0:
            previus = interval[initial-1]
        else:
            previus = 0

        if final < total_size:
            next = interval[final]
        else:
            next = 0

    val = (lastRMS*lastStep + np.abs(next) - np.abs(previus))
    / float(step)

    lastRMS = val
```

C. Funções implementadas em Python para avaliar cada um dos métodos de cálculo de envelopes, considerando as possíveis combinações de frequências de amostragem, tamanho de janela e porcentagens de sobreposição, usando as funções implementadas no ANEXO B

Script: Envelope.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# -----
# Bibliotecas a serem usadas pelas funções
# -----
from scipy.io import wavfile
fromTicTocGen import TicTocGen
from Functions import *
import numpy as np
import os
# frompylab import *
# -----
# Função para impressão de resultados em computador de escritório
# -----
def plotSignal(frequency, dataset, out, funcName, timeStamp,
               windowSize, overlap):

    imagesFolder = "../Images/"
    windowString = ""
    windowPrintString = ""
    overlapString = ""
    overlapPrintString = ""

    if windowSize != None:
```

```

windowString = "      Window size: " + str(windowSize)
windowPrintString = "Window size: " + str(windowSize)

if overlap != None:
    overlapString = "      overlap: " + str(overlap)
    overlapPrintString = "\toverlap: " + str(overlap)

print (windowPrintString + overlapPrintString + "\t:", timeStamp )

    # close()
    # plot(abs(dataset), color='blue')
    # plot(out, color='red')
    # name = "Frecuency: " + str(frecuency) + "      Function: " +
    funcName + windowString + overlapString
    # xlabel(name)

    # show()
    # savefig(imagesFolder + name + ".png")

def main():
# -----
#      Definição de variáveis de entorno para execução do teste
# -----
    samplesDir = "../Muestras/"
    functions = ["fullRMS", "fastRMS", "recursiveRMS",
"movingAverage", "fourierConvolve", "hilbertEnvelope",
"overlapRMS", "spectralEnergyEnvelope"]

    functions = ["spectralEnergyEnvelope"]
    functionsWindows = ["fullRMS", "fastRMS", "recursiveRMS",
"movingAverage", "fourierConvolve", "overlapRMS",
"spectralEnergyEnvelope"]

    functionsOverlap = ["overlapRMS"]
    functionsFrequency = ["spectralEnergyEnvelope"]
    windowSizes = [128, 256, 512, 1024, 2048, 4096, 8192]
    overlaps = [0, 0.25, 0.5, 0.75]

# -----
#      Carregando função de medição do tempo de execução
# -----

tictocker = TicTocGen()          # Para poder usar as funções
# ticetoc

```

```

# -----
#   loop para execução iterativa do teste
# -----
testFiles = os.listdir(samplesDir)
for file in testFiles:
    fs, dataset = wavfile.read(samplesDir+file) # fsedatasetsão arraysnumpy
                                                # como os dados
    dataset = dataset / float(2**15) # Converter todos os valores para
                                    # o intervalo [-1, 1]
    print ("\n===== \n"+file+"\n=====")

    for func in functions:
        print ("\n## "+func+"\n")

    if func in functionsFrequency:
        stringFrequency = ", " + str(fs)
    else:
        stringFrequency = ""

    if func in functionsWindows:
        for window in windowSizes:
            if func in functionsOverlap:
                for overlap in overlaps:
                    tictocker.tic()
                    out = eval(func + "(dataset, window, overlap" + stringFrequency + ")" )
                    timeStamp = str(tictocker.getToc())
                    plotSignal(fs, dataset, out, func, timeStamp, window, overlap)
            else:
                tictocker.tic()
                out = eval(func + "(dataset, window" + stringFrequency + ")" )
                timeStamp = str(tictocker.getToc())
                plotSignal(fs, dataset, out, func, timeStamp, window, None)

    else:
        if func in functionsOverlap:
            for overlap in overlaps:
                tictocker.tic()
                out = eval(func + "(dataset, overlap" + stringFrequency + ")" )
                timeStamp = str(tictocker.getToc())
                plotSignal(fs, dataset, out, func, timeStamp, None, overlap)
            else:
                tictocker.tic()
                out = eval(func + "(dataset" + stringFrequency + ")" )
                timeStamp = str(tictocker.getToc())

```

```
plotSignal(fs, dataset, out, func, timeStamp, None, None)
```

```
if __name__ == '__main__':  
    main()
```

D. Algoritmo EVD

- (a) Sinal de entrada
- (b) Filtragem
- (c) DEP
- (d) Espectrograma
- (e) Detecção EVD
- (f) Envelope (função de detecção)
- (g) Detecção de Eventos
- (h) Segmentação
- (i) Detecção EVD
- (j) Filtragem
- (k) Contorno extraído do evento

```
% -----  
%                               Inicialização do entorno de trabalho  
% -----  
%  
clearall;  
clc;  
% -----  
%                               Ler dados a serem processados  
% -----  
% E escolhida a pasta onde encontram-se os arquivos a serem processados,  
% neste caso, os arquivos de áudio encontram-se na pasta C:\Artigo\Data  
%  
cd ('C:\Artigo\Data')  
nomefile = 'XC116219.wav'% Nome do arquivo a ser processado é  
% carregado na variável nomefile  
  
[sound, rate, nbits] = wavread(nomefile); % Leitura da informação do  
% arquivo de  
file_size = size(sound);      % áudio, dados, frequência  
% de amostragem  
                                % profundidade em bits do  
                                % sinal e tamanho  
                                % em número de amostras  
                                %(samples)
```

```

%
% -----
%                               Filtragem do sinal
% -----
% O sinal é filtrado de acordo com a banda de frequência que deseja
% ser analisada, neste caso é usado um filtro passa-banda do tipo
% 'equiripple', usando a função personalizada 'BPFilter.m'

DadosFiltrados = BPFilter(DataInput, FFst1, FFp1, FFp2, FFst2,
FAst1, FAp, FAst2, fs)

% -----
%                               Cálculo do Espectro de Potência do sinal
% -----
% O espectro aqui é usado como uma ferramenta de análise exploratória
% para avaliar a possibilidade de usar outros tipos de filtros,
% visando atenuação de alguma banda específica do sinal com o
% intuito de deixar \mais visíveis" os eventos objetivo da detecção.

% -----
%                               Cálculo do Espectrograma do Sinal
% -----
% É calculado e plotado o espectrograma do sinal filtrado.

F=[ ];
janela=512;
[S,F,T]=spectrogram(y,janela,0.875*janela,F,rate);
P = abs(S);

% Se desejado usar o espectrograma para avaliar os dados que estão
% sendo processados, é possível imprimir na tela com os comandos
% a seguir:

surf(T,F,10*log10(P),'edgecolor','none'); axis tight;
view(0,90);

% -----
%                               Detecção de valores extremos
% -----
% A matriz de dados P calculada previamente, é convertida em um
% vetor unidimensional com o intuito de calcular e extrair os

```

```

% outliers. É usada a função personalizada MyOutlierDetec.m
%

PP = P(:)';
[OutlierValues,Qt1,Qt3,Iqr,Out1,Out2,bigodeinf,bigodesup] =
    MyOutlierDetec(PP);

% É definido o valor limite para extrair outliers, e estes
% Extraídos posteriormente
%
limtrheshold = bigodesup;
    PPP=P>=limtrheshold;
MOutP = P.*PPP;

% -----
%Análise exploratória (Espectrograma com valores extremos)
% -----
%
surf(T,F,10*log10(MOutP ),'edgecolor','none'); axis tight;
view(0,90);

% -----
%               Aprimoramento dos valores extremos (Filtro de medianas)
% -----
%
MOutPMeanFiltered = medfilt2(MOutP, [3 3]);

% -----
%  Análise exploratória (Espectrograma com valores
%  extremos filtrados)
% -----
%
surf(T,F,10*log10(MOutPMeanFiltered ),'edgecolor','none');
axis tight;
view(0,90);

% -----
%               Recuperação dos valores originais contidos na matriz P
% -----
% São recuperados os valores originais da matriz P, para os pontos

```

```

% escolhidos como Valores Extremos, com essa informação é
% construída a função de detecção, nas etapas subsequentes.

teste=MOutPMeanFiltered>0;
teste2 = P.*teste;,
%
%
% -----
%               Construção do Envelope, ou função de Detecção
% -----
%
Envelopes.data= (mean(teste2));
SinalEnvelope = horzcat(SinalEnvelope,Envelopes(i).data) ;

% -----
%               Extração de Eventos
% -----
%
SinalEnvelope=SinalEnvelope';
SinalEnvelope = SinalEnvelope / max(SinalEnvelope);
SinalEnvelope = medfilt1(SinalEnvelope,150);
[SEnvFilasSEnvCols] = size(SinalEnvelope);

TimePerBin = (SoundSize*(NumSegmentos))/(SEnvFilas);
TimeVector = TimePerBin:TimePerBin:(SoundSize * NumSegmentos);
TimeVector = TimeVector';

Dataset = [TimeVectorSinalEnvelope];
% Detectar por ZeroCrossing
%
DatasetMax = Dataset(:,2)>0;
%plot(DatasetMax); ylim([0.75 1.5]);

[VectIniVectFin] = extraevento(DatasetMax);
VectIniTime = VectIni * TimePerBin;
VectFinTime = VectFin * TimePerBin;
Eventos = [VectIniTimeVectFinTime];

% O vetor Eventos contém o tempo onde começa e finaliza cada evento
% detectado. Para extraer o contorno isolado de um evento detectado,
% basta segmentar o arquivo e, para cada segmento, processar novamente
% até o passo (E).

```

```

% Script: BPFILTER.m

% -----
%                               Implementação de filtro passa-bandas
% -----
function [DataFiltered] = BPFILTER(DataInput, FFst1, FFp1,
FFp2, FFst2, FAst1, FAp, FAst2, fs)
% Filtro PasaBanda : Desenha um Filtro Passa-Banda
% e Filtra os dados
%     FFst1 = Banda para atenuação
%     FFp1  = Banda inferior
%     FFp2  = Banda superior
%     FFst2 = Banda para atenuação
%     FAst1 = 60 Atenuação em dB Banda inferior
%     FAp=1 = 1
%     FAst2 = 60 Atenuação em dB Banda superior
% http://stackoverflow.com/questions/5591278/high-pass-
% filtering-in-matlab

FFst1=(FFst1/(fs*0.5));
FFp1=(FFp1/(fs*0.5));
FFp2=(FFp2/(fs*0.5));
FFst2=(FFst2/(fs*0.5));

d = fdesign.bandpass(FFst1,FFp1,FFp2,FFst2,FAst1,FAp,FAst2);
Hd = design(d,'equiripple');
% -----
%                               filtrar os dados
% -----
DataFiltered = filter(Hd,DataInput);
%
end

```

E. SPL para grandes volumes de dados

```
functionBtmProcss_Callback(hObject, eventdata, handles)
tic
dBData;
functiondBData
tic
NomesFiles=dir('*.wav');
TimeSegmento=60;
SubMtzFile=[];
SubMtzDesv=[];
MtzFile=[];
MtzDesv=[];
fori=1:length(NomesFiles)
NomeFile=NomesFiles(i).name;
[Dataset,Fs]=audioread(NomeFile);
NumSegmFile=floor(length(Dataset)/(Fs*TimeSegmento));
for j=1:NumSegmFile
DataPascalIni=floor(Fs*j*TimeSegmento-Fs*TimeSegmento+1);
DataPascalFin=floor(Fs*j*TimeSegmento);
DataPascal=Dataset(DataPascalIni:DataPascalFin)*62;
DatadB=10*log10((DataPascal.^2)/(1e-6^2));
DatadB(~isfinite(DatadB))=65.54;
SubMtzFile(j)=mean(DatadB); %
SubMtzDesv(j)=std(DatadB);
end
MtzFile=vertcat(MtzFile,(SubMtzFile)');
MtzDesv=vertcat(MtzDesv,(SubMtzDesv)');
SubMtzFile=[];
SubMtzDesv=[];
end
toc
%-----
TotalTimeSeg=floor(length(NomesFiles)*length(Dataset)/Fs);
VectEixX = linspace(7,17,(length(MtzFile)));
figure(1);
set(gcf,'PaperUnits', 'Centimeters');
xSize = 26; ySize=8;
```

```
xLeft = (21-xSize)/2; yTop=(30-ySize)/2;
set(gcf,'PaperPosition',[xLeftyTopxSizeySize]);
set(gcf,'Position',[100 250 xSize*50 ySize*50],'visible','on');
plot (VectEixX,MtzFile);
xlabel ('Tempo (h)');
ylabel ('SPL (dB) rel 1 mPa');
%-----
end
    Fs
toc
end
```

F. Espectrograma para grandes volumes de dados

% O código apresentado a seguir faz parte da interface de usuário
% criada para analisar grandes volumes de dados, corresponde
% unicamente com o processo de cálculo espectrograma para grandes
% volumes de dados.

```
functionBtmSpectrogram_Callback(hObject, eventdata, handles)
tic
TimeSegmento=60;
LtsaWSize=4096;
    % Ler arquivos wav na pasta a analisar
NomesFiles=dir('*.wav');
ResultadoFinal=[];
MtxPwelch=[];
    %.....
fori=1:length(NomesFiles)
NomeFile=NomesFiles(i).name;
    NomePlot=strcat(NomeFile,'.jpg');
% =====
%           Calcular num de columns para a matrix do ltsa
% -----
ltsacols=0;
    [LtsaX, Fs, nbits] = wavread(NomeFile);
    [numfilas, numcols] = size(LtsaX);
FileDura = numfilas / Fs;
FileSegmentos=floor(FileDura/TimeSegmento);
    ltsacols=ltsacols+FileSegmentos;
% =====
%           Calcular num de filas para a matrix do ltsa
% -----
dataset=LtsaX(1:(Fs*TimeSegmento));
    [ltsa, freq]=pwelch(dataset,[],[],LtsaWSize,Fs);
[ltsarows,ltsacolseg]=size(ltsa);
%=====
%           Criar matriz que vai conter a LTSA
%-----
```



```
view(0,90);
caxis([-130, -40])
set(gcf, 'Color', 'white'); % white bckgr
xlabel ('Tempo (h)');
ylabel ('Frequência (Hz)');

%saveas(gcf, NomePlot, 'jpg');
toc
end
```