

Salviano Filipe Silva Pinto Soares

Processamento Digital de Sinal

(Conteúdos e métodos de organização científica e de execução pedagógica)

Relatório sobre os Conteúdos e os Métodos de Organização Científica e de Execução Pedagógica da Unidade Curricular ***Processamento Digital de Sinal***, do ***Mestrado em Engenharia Eletrotécnica e de Computadores***, no âmbito de Provas de Agregação de acordo com o Artigo 8.º, alínea b), que se refere a alínea b) do Artigo 5.º do Decreto-Lei n.º 239/2007 de 19 de junho de 2007.

Universidade de Trás-os-Montes e Alto Douro
Escola de Ciências e Tecnologia
Departamento de Engenharias
julho 2020

ÍNDICE

1 – RESUMO	3
1.1 – <i>Introdução</i>	5
2 – APRESENTAÇÃO E ENQUADRAMENTO DO TEMA.....	9
2.1 – <i>Processamento Analógico versus Digital</i>	9
2.2 – <i>Vantagens do DSP sobre o ASP.....</i>	11
2.3 – <i>Análise e Filtragem de sinais.....</i>	12
3 – A UNIDADE CURRICULAR EM OUTRAS INSTITUIÇÕES DO ENSINO SUPERIOR	13
4 – ALINHAMENTO DA UNIDADE CURRICULAR NO CURSO.....	15
4.1 – <i>Objetivos de aprendizagem da Unidade Curricular PDS.....</i>	17
4.2 – <i>Conteúdos programáticos.....</i>	18
4.2.1 – <i>Planificação da atividade letiva</i>	18
4.3 – <i>Demonstração da coerência dos conteúdos programáticos com os objetivos de aprendizagem da UC</i>	19
4.4 – <i>Metodologias de Ensino (Avaliação incluída).....</i>	20
4.4.1 – <i>Obtenção de frequência à unidade curricular</i>	21
4.4.2 – <i>Sistema de Avaliação.....</i>	21
5 – BIBLIOGRAFIA RECOMENDADA PARA A UC PDS	24
6 – CONSIDERAÇÕES FINAIS.....	26
7 – REFERÊNCIAS BIBLIOGRÁFICAS	27
Anexo_A.....	30
Anexo_B.....	31
Anexo_C.....	32
Anexo_D	33
Anexo_E.....	34
Anexo_F.....	35

1 – RESUMO

O relatório aqui apresentado corresponde a um programa da Unidade Curricular (UC) de **Processamento Digital de Sinal (PDS)**, que se pretende venha a integrar o novo plano curricular do Mestrado em Engenharia Eletrotécnica e de Computadores (MIEEC) sucedâneo do Mestrado Integrado em Engenharia Eletrotécnica e de Computadores cinco (5) anos para as Licenciatura (3 anos) e o Mestrado (2 anos) em Engenharia Eletrotécnica e de Computadores que decorre da nova legislação de 2018 relativa ao novo Regime Jurídico de Graus e Diplomas inscrita no Decreto-Lei nº65/2018 de 16 de agosto, imposta às ofertas educativas da UTAD tal como a todas as outras instituições de ensino superior do país. Referia-se que por Despacho da ECT o candidato foi nomeado para a Comissão que elaborou a proposta para o curso de Mestrado e recomendado como Professor Regente da UD de Processamento Digital de Sinal em função da sua experiência na área de mais de vinte (anos).

Atualmente a UC consta da estrutura curricular do Mestrado Integrado em Engenharia Eletrotécnica e de Computadores (MIEEC) da UTAD, existe já uma UC denominada Processamento Digital de Sinal (DR, 2.ª série - N.º 169 - 31 de agosto de 2015). O MIEEC da UTAD entrou em funcionamento no ano letivo 2015/2016 implementando o conceito de ciclo de estudos integrado conducente ao grau de Mestre, conforme estabelecido na Lei de Bases do Ensino Superior, surgindo na sequência de uma reformulação do anterior modelo adaptado ao Modelo de Bolonha que assentava em dois ciclos de estudos. Com esta reformulação, o plano de estudos do curso passou a ter um tronco comum de três anos seguido de três percursos formativos alternativos denominados de ramos, Eletrónica e Instrumentação; Automação Industrial e Telemática. Existindo em cada ramo um conjunto de UC comuns e uma área de especialização à qual estão associadas UC opcionais.

Desde a génese dos cursos de Electrotecnia na UTAD em 1990, a UC de PDS, sendo considerada de referência, fez sempre parte das disciplinas obrigatórias estando atualmente, como já referido, nos três ramos a ser lecionada no 1º semestre do 4º ano (7º semestre do curso). Segundo o autor, este continua a ser o momento adequado para a inserção da nova UC no *curriculum*, atendendo à especificidade dos assuntos a abordar e ao nível de conhecimentos necessários adquiridos anteriormente pelos alunos nomeadamente no que diz respeito a conceitos fundamentais de Matemática e Teoria de Sistemas de Sinais numa abordagem contínua que passa também pelos Sistemas de

Electrónica Analógica ou os Sistemas Digitais nos primeiros anos do curso cujos temas voltam a ser estudados em PDS agora em realizações discretas onde se discutem os teoremas e regras fundamentais inerentes ao processamento digital dos dados face a dicotomia dos paradigmas analógico *versus* digital que se sintetizam no termo que entretanto se consolidou na comunidade científica da área: *digital twin*.

Ao longo das últimas décadas, a área de PDS desenvolveu-se quer ao nível teórico e tecnológico tendo sido um fator fundamental a medida crescente da disponibilidade computacional. Uma outra razão importante para o seu sucesso foi a sua implementação na indústria e na sociedade em geral em grande parte devido à evolução dos custos de utilização de software (SW) e hardware (HW) que se tornaram mais baixos fruto também do progresso e desenvolvimento tecnológico.

Para além da área da Electrotecnia, novas tecnologias e inúmeras aplicações em áreas distintas, como por exemplo a Medicina ou a Biotecnologia, tiram hoje partido de algoritmos de PDS aumentando assim a exigência de conhecimentos de base aos futuros Engenheiros Electrotécnicos o que só por si confirma a necessidade fundamental da unidade curricular de **Processamento Digital de Sinal** continuar a integrar o *curriculum* de um qualquer curso na área da Engenharia Electrotécnica como é o caso do Mestrado em Engenharia Electrotécnica e de Computadores (MEEC) na UTAD já nas suas próximas edições como no passado.

Como já aludido, a unidade curricular de **Processamento Digital de Sinal**, pretende complementar as estratégias de resolução de problemas de engenharia numa perspectiva analógica, fazendo agora uso do conhecimento dos conceitos fundamentais de sinais e sistemas discretos bem como de técnicas nos domínios do tempo e da frequência: análise e filtragem. Numa outra perspectiva é também objetivo que desenvolva competências para a prossecução de estudos avançados na área de PDS.

1.1 – Introdução

No âmbito das Provas de Agregação, o presente relatório incide sobre a UC de Processamento Digital de Sinal/*Digital Signal Processing*, que se pretende faça parte do plano de estudos do **Mestrado em Engenharia Eletrotécnica e de Computadores** da UTAD.

O candidato tem já uma vasta experiência nesta área adquirida, por um lado, ao longo dos mais de vinte (20) anos de regência da UC entretanto robustecida pelo seu mestrado em 1995 e doutoramento em 2005 na área e por outro adquirida ao longo da sua experiência em gestão de projetos de investigação, em orientações alunos bolseiros ERASMUS, de fim de curso de Licenciatura, de Mestrado ou Doutoramento podendo muito do seu trabalho produzido na área de PDS ser testemunhado nas publicações com alunos validadas *entre pares* cujos resultados foram naturalmente sucedâneos à lecionação de PDS: contam-se alguns exemplos em,

- Silveira, L. *et al.* Soares, S., (2020), *Repair of the recurrent laryngeal nerve – Experimental study*, Revista Portuguesa de Cirurgia, Sociedade Portuguesa de Cirurgia, II Série, Volume 46:19-31, ISSN: 1646-6918. (<https://revista.spcir.com/index.php/spcir>)
- Neves, F., Soares, S., Assunção, P., (2018) – *Optimal voice packet classification for enhanced VoIP over priority-enabled networks* – *Journal of Communications and Networks*, 20(6), pg. 554-564, Dec. 2018. DOI:10.1109/JCN.2018.000088,
- Silva, S., Soares, S., Cabral, M., Neves, F., Assunção, P., (2017) – *A dynamic programming algorithm to select optimal high-priority voice segments using Arduino* – 17th IEEE International Conference on Smart Technologies, EUROCON 2017 , Conference Proceedings, pg. 271-276, OHRID, R. Macedonia;
- Silva, S., Valente, A., Soares, S., Cabral, M., Paiva, J., Bartolomeu, P., (2016) – *Morse Code Translator Using the Arduino Platform: Crafting the*

- Future of Microcontrollers* – Proceedings of Science and Information Conference 2016 (SAI 2016 – www.conference.thesai.org), pg. 675-680, July 13-15, London, United Kingdom. ISBN: 978-1-4673-8460-5. DOI: 10.1109/SAI.2016.7556055;
- Silva, S., Soares, S., Valente, A., Marcelino, S. (2015) – *Digital Sound Processing using Arduino and MatLab* – Proceedings of Science and Information Conference 2015, SAI 2015, pg. 1184-1191, July 28-3, London, United Kingdom. ISBN:978-1-4799-8546-3;
 - Reis, M., Soares, S., Cardeal, S., Morais, R., Peres, E., Ferreira, P. (2014) – *Teaching Fourier series expansions in undergraduate education with the help of the FouSE Android application* – International Journal of Interactive Mobile Technologies (IJIM). Vol. 8(1), pg. 26-31, ISSN: 1865-7923.
 - Costa, M.; Barroso, J.; Soares, S.; (2012) – *Signal Processing Interpolation Educational Workbench* – *Computer Application in Engineering Education*, 20(2), pg. 356-363, DOI:10.1002/cae.20402;
 - Pereira, L., Faustino J., Soares M., Soares S. (2011) – *Simulation of a Reverberation Room* – Forum Acusticum 2011, pg. 1947-1952 Aalborg – Denmark. Edited by Danish Acoustical Society – DAS, 27 de June – 01 July. ISBN: 978-84-694-1520-7.
 - Lebre, P.; Soares, S.; Silva, S.; Silveira, L. (2010) – *Avaliação da voz após cirurgia ao NLR* – in 12º Encontro da Secção Portuguesa da *Audio Engineering Society*, 8 de Outubro, UA, Aveiro.
 - Neves, F.; Soares, S.; Reis, M. J.C.S.; Tavares, F.; Assunção, P.. (2008) – *VoIP Reconstruction Under a Minimum Interpolation Algorithm* – In 2008 IEEE International Symposium on Consumer Electronics, Vols. (1-2), pg. 51-53. CD-ROM. ISBN 978-1-4244-2422-1.

- Neves, F.; Soares, S.; Assunção, P. (2007) – *O Algoritmo de Papoulis-Gerchberg e a Reconstrução de Voz Comutada* – Engenharias 07, Inovação&Desenvolvimento, CD-ROM;
- Salviano Filipe Silva Pinto Soares, *Algoritmos de Reconstrução de Sinais e Correção de Erros*, Tese apresentada à UA para a obtenção do grau de Doutor em Engenharia Electrotécnica sob orientação do Prof. Dr. Paulo Jorge dos Santos Gonçalves Ferreira, Maio de 2005.
- Salviano Filipe Silva Pinto, *Técnicas de Reconstrução de Sinal aplicadas à Transmissão de Voz*, Dissertação apresentada à UA para a obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações sob orientação do Prof. Dr. Paulo Jorge dos Santos Gonçalves Ferreira, Dezembro de 1995.

Atendendo à importância que da área de Teoria de Sinal consubstanciada na presença mandatória da disciplina de PDS, apesar de constar de praticamente todos os cursos conceituados de Engenharia Electrotécnica em todo o Mundo, enumeram-se na seção 3 algumas das principais universidades nacionais e estrangeiras que têm incluída nos seus planos curriculares PDS.

A inclusão desta área na formação de mestres em Engenharia Eletrotécnica e de Computadores da UTAD continua a revelar-se não só oportuna mas necessária, pois para além do já referido quanto à transversalidade do tema, todo o progresso provido pelo desenvolvimento tecnológico nomeadamente a nível computacional, deve também ser incluído no processo educativo tornando o ensino de PDS mais efetivo o que resultará num enriquecimento à sua aprendizagem e compreensão como será apresentado nas metodologias de ensino através da realização de um conjunto de trabalhos em MatLab, propostas de Tutoriais, promoção de Seminários Temáticos ou à realização/assistência de cursos *on-line* síncronos e/ou assíncronos ao longo da leção dos conteúdos programáticos que o *Admirável Mundo Novo* nos oferece muito útil a plenamente adequado ao contexto de Pandemia Mundial dos nossos dias.

A UC de Processamento Digital de Sinal, será proposta para o 1º semestre do 1º ano do mestrado, dado que nesta fase da sua formação os alunos possuem já determinados

requisitos e conhecimentos nomeadamente nas áreas da Matemática, Sistemas e Sinais, Sistemas Digitais ou a Electrónica Analógica.

Este relatório encontra-se organizado da seguinte forma. Depois do Resumo e Introdução, na Secção 2 é feita a Apresentação do tema da UC como complementar ao paradigma do processamento analógico de sinal. De seguida é feito um levantamento em termos nacionais e internacionais sobre o ensino do PDS nas instituições de ensino superior. Nas secções 4 e 5 é respetivamente enquadrar-se a UC no curso bem como discriminam-se as propostas de desenvolvimento curricular de Processamento Digital Sinal num curso de Engenharia Electrotécnica e de Computadores, completando-se na secção 6 com a Bibliografia recomendada. Conclui-se com breves considerações finais e ulteriormente apresentam-se as Referências Bibliográficas que serviram à escrita deste documento bem como em Anexos toda a documentação de suporte a este documento.

2 – APRESENTAÇÃO E ENQUADRAMENTO DO TEMA

No mundo moderno, estamos rodeados de uma grande variedade de sinais de várias formas. Alguns são naturais, mas muitos parte são artificiais e produzidos pelo Homem. Alguns são necessários e úteis (voz humana), outros agradáveis (música) enquanto muitos outros são indesejados ou desnecessárias em determinadas situações (ruído). Num contexto de engenharia podemos considerar um sinal como uma entidade que transporta informação, desejada e não desejada podendo desde logo definir-se o processamento de sinal como sendo a tentativa de extrair/realçar informação considerada útil a partir desse sinal.

2.1 – Processamento Analógico versus Digital

Muitos dos sinais que encontramos na prática e todos os do mundo real são predominantemente analógicos. Estes sinais que existem para todos os valores de amplitude e no tempo podem ser processados usando redes elétricas com dispositivos passivos (resistências) ou elementos ativos (condensadores ou bobines) admitindo-se para tal um Processamento Analógico dos Sinais (ASP).

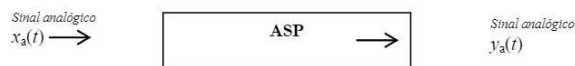


Fig. 2.1: Processamento Analógico de Sinal

Numa outra abordagem por vezes denominada de discreta/digital, os sinais podem também ser processados utilizando HW digital contendo módulos de adição, multiplicadores, elementos lógicos ou mesmo processadores dedicados para o efeito sendo para isso necessário converter os sinais analógicos em digitais onde são tomadas para a representação apenas valores finitos no tempo sendo a amplitude função do número de bits disponíveis. O diagrama de blocos representa genericamente o Processamento Digital de Sinais.

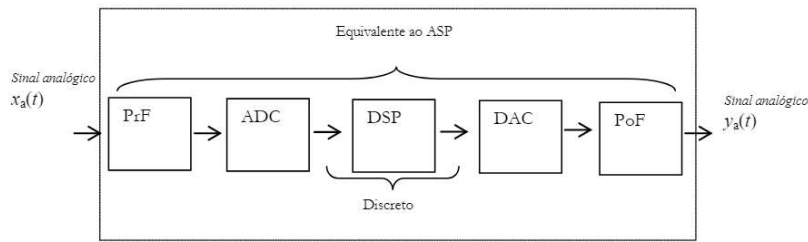


Fig. 2.2: Processamento Digital de Sinais

onde

- **PrF:** Pré-filtro ou filtro *anti aliasing* que torna o sinal analógico de banda limitada de modo a prevenir o *aliasing*;
- **ADC:** Conversor analógico/digital que transforma o sinal de entrada numa sequência discreta;
- **DSP:** Processador Digital de Sinal;
- **DAC:** Conversor digital/analógico que realiza a transformação inversa da ADC produzindo um sinal em escada a partir de um conjunto de números binários;
- **PoF:** Pós filtro Passa-Baixo que transforma o sinal em escada no sinal analógico de saída.

Por análise dos diagramas de blocos quando se confrontam os dois paradigmas a opção pelo processamento digital parece ser mais complicada pois exige mais das questões técnicas com a inevitável passagem dos dados para domínios de representação diferentes pelo que se pode colocar a questão: porquê optar na maior parte das vezes hoje em dia pelo DSP? A resposta está nas várias vantagens que DSP quando se compara com o processamento analógico da informação.

2.2 – Vantagens do DSP sobre o ASP

Uma das maiores desvantagens dos ASP reside entre outras, na sua dificuldade em realizar tarefas complicadas de processamento de sinal que se pode traduzir também em pouca flexibilidade ou na dificuldade no desenho dos sistemas conduzindo regra geral a soluções finais do produto mais dispendiosas. Por outro lado, a opção por DSP permite a realização aplicações mesmo num computador pessoal pois o seu poder de cálculo é já muito considerável. Algumas vantagens dos DSP são:

- regra geral as aplicações podem ser desenvolvidas e testados num PC genérico evidenciando características de portabilidade;
- em contraste com os sistemas analógicos são imunes a fatores físicos como é o caso da temperatura;
- as operações num DSP são facilmente reprogramáveis em tempo real bastando para isso alterar o SW;
- a possibilidade de criar produtos de baixo custo em suporte VLSI constitui também uma possibilidade que pode ser explorada com as realizações discretas.

Apesar de existirem também algumas limitações nomeadamente no que diz respeito à velocidade de processamento a altas frequências, à máxima frequência de amostragem disponível ou mesmo a melhor QoS das respostas em frequência de alguns sistemas analógicos em particular e em determinados regimes nomeadamente em baixas frequências como é caso dos Sistemas de Áudio analógicos, as vantagens enumeradas fazem com que hoje em dia a opção por um processamento digital de sinais se torne a primeira escolha em muitos sistemas tecnológicos com aplicações na Eletrónica de Consumo, nas Telecomunicações ou mesmo as áreas afins como a Medicina.

2.3 – Análise e Filtragem de sinais

No processamento digital de sinais podemos agrupar as tarefas em dois grandes grupos: a Análise de Sinal ou a Filtragem de Sinais.

- **Análise de Sinal:** tarefas relacionadas com o estabelecimento de medidas das propriedades dos sinais: algumas aplicações:
 - Análise espectral (frequência/fase);
 - Reconhecimento de fala/orador;
 - Identificação de alvos.
- **Filtragem de Sinais:** os sistemas que realizam esta tarefa normalmente designam-se por filtros: algumas aplicações:
 - remoção de ruído indesejável;
 - remoção de interferências;
 - separação em bandas de frequência;
 - modelação espectral.

Tanto a Análise como a Filtragem poderão ser realizadas nos domínios do tempo ou da frequência sendo complementares como no caso dos sintetizador digital de voz onde numa primeira instância o sinal é analisado e estudadas as suas características no tempo para numa segunda fase se utilizar um filtro digital para gerar a voz sintetizada ou mesmo em algoritmos de reconstrução de sinal como é o caso do *Papoulis-Gerbberg*.

Desta forma considera-se necessário dotar os conteúdos a propor para a UC PDS de aspetos relacionados com:

- descrições básicas de sinais/sistemas discretos no tempo;
- a análise em frequência com a transformada de *Fourier*, de sinais/sistemas discretos no tempo;
- a amostragem e reconstrução de sinais analógicos;
- a análise da transformada Z como abordagem na frequência;
- a transformada de *Fourier* discreta (DFT) bem como algoritmos rápidos para a sua realização como é o caso da transformada rápida de *Fourier* (FFT);
- a descrição de estruturas de realização de filtros digitais;
- o projeto de filtros digitais com resposta impulsional infinita (IIR);
- o projeto de filtros digitais com resposta impulsional finita (FIR).

3 – A UNIDADE CURRICULAR EM OUTRAS INSTITUIÇÕES DO ENSINO SUPERIOR

Relativamente ao perspectivado em outras instituições de ensino superior, ainda que de forma não exaustiva, nesta secção comparam-se a UC de PDS lecionadas em algumas instituições de ensino superior tanto portuguesas como internacionais onde se conclui como já referido que não só PDS mas outras UC de Teoria de Sinal são mandatórias nos currículos dos cursos mais conceituados em Engenharia Electrotécnica em todo o mundo. Os elementos indicados nesta secção e apresentados na tabela 1, baseiam-se na informação disponibilizada na *world wide web* pelas respetivas instituições de ensino, para o ano letivo 2018/2019.

Tabela 1 – UC de instituições portuguesas e internacionais.

Identificador	Nome da UC	Universidade	Curso	Ano / Semestre	ECTs	Professor Regente 2018/2019
P1	PDS	FEUP	Mestrado Integrado em Engenharia Eletrotécnica e de Computadores	3º Ano / 2º Sem	6.0	Anibal Ferreira / Sérgio Cunha
P2	PDS	UA	Mestrado Integrado em Engenharia Eletrotécnica e de Computadores	3º Ano / 2º Sem	6.0	José Vieira / Ana Tomé
P3	PDS	IST	Mestrado Integrado em Engenharia Eletrotécnica e de Computadores	3º Ano / 2º Sem	6.0	Jorge Marques / Maria Silveira
P4	PDS	UTAD	Mestrado Integrado em Engenharia Eletrotécnica e de Computadores	4º Ano / 1º Sem	6.0	Salviano Soares
E1	DSP	University of Essex	Master's Degree Electrical Engineering			Dr Nikolaos Thomos
E2	DSP	MIT	Master's Degree Electrical Engineering			Dr. Alan V. Oppenheim
E3	DSP	Universidade de Cambridge	Master's Degree Electrical Engineering			Dr Markus Kuhn

A tabela 2 pretende comparar o plano curricular desta UC de PDS, proposta neste documento, com UC similares à de PDS em outras instituições nacionais e estrangeiras, foram analisadas as UC referenciados na tabela 1.

Tabela 3 - Comparação dos Conteúdos programáticos.

Tópico	P1	P2	P3	P4	E1	E2	E3
I. Sinais e sistemas discretos (SD); Sistemas recursivos e não recursivos; Resposta impulsional; Sistemas com resposta impulsional finita (FIR) e infinita (IIR); Classificação de SD quanto à linearidade, invariância à translação, estabilidade causalidade; (SLIT). Convolução discreta; Resposta em frequência; Transformada de <i>Fourier</i> (FT); propriedades.	✓	✓	✓	✓	✓	✓	✓
II. Amostragem: reconstrução perfeita; Sistemas <i>sample and hold</i> .	✓	✓	✓	✓	✓	✓	✓
III. Transformada de Fourier discreta (DFT); Transformada rápida de <i>Fourier</i> (FFT).	✓	✓	✓	✓	✓	✓	✓
IV. Transformada z: Convergência, Propriedades e Inversa; Relação com a transformada de Fourier; Função de transferência.	✓	✓	✓	✓	✓	✓	✓
V. Projecto de filtros digitais do tipo IIR; Invariância da resposta impulsional; Transformação bilinear.	✓	✓	✓	✓	✓	✓	✓
VI. Projecto de filtros digitais do tipo FIR; Sistemas com fase linear; Método da janela; Método da amostragem da função de transferência.	✓	✓	✓	✓	✓	✓	✓

Como se pode depreender pela tabela 2, existe uma uniformização nos planos curriculares das diferentes UC nos diversos cursos, quer nas universidades nacionais quer nas universidades estrangeiras. No entanto, julga-se que a UC proposta neste relatório abrange os tópicos fundamentais de PDS. Além disso, existem temas nos programas de outras instituições que podem ser lecionadas em diferentes unidades curriculares. As tabelas 3 e 4 apresentam as comparações das metodologias de ensino e das metodologias de avaliação implementadas nas diferentes UC.

Tabela 4 - Comparação das metodologias de ensino.

Metodologias	P1	P2	P3	P4	E1	E2	E3
Exposição Teórica	✓	✓	✓	✓	✓	✓	✓
Trabalhos Práticos	✓	✓	✓	✓	✓	✓	✓
Laboratórios	✓	✓	✓	✓	✓	✓	✓

Tabela 5 - Comparação das metodologias de avaliação.

Metodologias de avaliação	P1	P2	P3	P4	E1	E2	E3
Avaliação distribuída com exame final	65%	60%	70%	70%	60%	60%	70%
Avaliação distribuída sem exame final	Não	Não	Não	Não	Não	Não	?
Avaliação contínua	Sim	Sim	Sim	Sim	Sim	Sim	?
Participação presencial	5,25%	N/A	N/A	N/A	N/A	N/A	?
Trabalhos práticos	14,87%	40%	30%	30%	40%	40%	?
Trabalho escrito	14,88%						?

4 – ALINHAMENTO DA UNIDADE CURRICULAR NO CURSO

A Engenharia Eletrotécnica e de Computadores encontra-se na vanguarda das áreas chave do desenvolvimento, sendo transversal a todas as áreas da engenharia e estando no centro das decisões estratégicas ligadas ao desenvolvimento sustentado e à melhoria da qualidade de vida. Assim, o Mestrado (2º ciclo) em Engenharia Eletrotécnica e de Computadores, onde se enquadra a UC aqui apresentada, deverá ter como grandes objetivos a formação de profissionais com um elevado nível de qualidade técnica, científica e ética, capazes de responder às necessidades de modernização da economia à escala global neste domínio, tendo em conta os ramos de especialização (Automação, Controlo e Energia; Eletrónica e Instrumentação; Telemática) com a UC PDS mandatória aos três ramos. Devendo ainda fornecer uma formação sólida que habilite aqueles que o desejarem a prosseguirem os seus estudos num 3º ciclo (Doutoramento) e uma carreira na área da investigação.

Desde o encontro realizado em Bolonha, em 1998, que as universidades europeias iniciaram um processo de criação de uma área comum de ensino superior europeu, com vista a permitir uma maior mobilidade dos alunos entre as várias instituições europeias e o consequente reconhecimento dos graus por todas as instituições. O denominado *Processo de Bolonha* introduz um sistema de ensino baseado em três ciclos, em que o 1º ciclo corresponde a uma Licenciatura (ou Bacharelato), de 3 ou 4 anos, o 2º ciclo corresponde ao Mestrado, e o 3º ciclo ao Doutoramento. Além desta alteração, o Processo de Bolonha pressupõe uma mudança no paradigma de ensino, introduzindo um modelo de ensino centrado no aluno. Este modelo, baseia-se no desenvolvimento de competências onde se incluem as competências de natureza genérica, mas também as competências de natureza específica, associadas à área de formação e onde a componente experimental e de projeto desempenham um papel importante. A tabela 6 apresenta o plano de estudos proposto para o Mestrado (2º ciclo) em Engenharia Eletrotécnica e de Computadores onde se testemunha que também no Ramo de Especialização em Automação, Controlo e Energia onde é evidente a preocupação em atingir os objetivos atrás citados.

Tabela 6 - Proposta de Plano de Estudos do Mestrado (2º ciclo) em Engenharia Eletrotécnica e de Computadores – Ramo de Especialização em Automação, Controlo e Energia

Unidade Curricular	Regime	Ano Semestre	Horas de Trabalho		ECTS
			Total	Contacto	
Processamento Digital de Sinal	O	1 / 1S	162	T-30; PL-30; OT-4,5	6
Automação Industrial	O	1 / 1S	162	T-30; PL-30; OT-4,5	6
Otimização e Algoritmos	O	1 / 1S	81	TP-30; OT-3,0	3
Gestão de Projetos em Engenharia	O	1 / 1S	81	TP-30; OT-3,0	3
Sistemas de Microcontroladores	O	1 / 1S	162	T-30; PL-30; OT-4,5	6
Opção A-1	OP	1 / 1S	162	T-30; PL-30; OT-4,5	6
Robótica	O	1 / 2S	162	T-30; PL-30; OT-4,5	6
Controlo Digital	O	1 / 2S	81	TP-30; OT-3,0	3
Gestão e Qualidade da Energia Elétrica	O	1 / 2S	162	T-30; PL-30; OT-4,5	6
Sistemas de Mobilidade Elétrica	O	1 / 2S	81	TP-30; OT-3,0	3
Sistemas Inteligentes	O	1 / 2S	162	T-30; PL-30; OT-4,5	6
Opção A-2	OP	1 / 2S	162	T-30; PL-30; OT-4,5	6
Preparação para a Dissertação	O	2 / 1S	81	OT-33	3
Energias Renováveis e produção Descentralizada	O	2 / 1S	162	T-30; PL-30; OT-4,5	6
Instalações Elétricas Industriais	O	2 / 1S	162	T-30; PL-30; OT-4,5	6
Controlo Avançado	O	2 / 1S	162	T-30; PL-30; OT-4,5	6
Controlo Inteligente	O	1 / 2S	81	TP-30; OT-3,0	3
Opção A-3	OP	1 / 2S	162	T-30; PL-30; OT-4,5	6
Dissertação	O	2 / 2S	810	T-30; PL-30; OT-4,5	30
Opções A-1					
Segurança em Instalações Elétricas	OP	1 / 1S	162	T-30; PL-30; OT-4,5	6
Gestão de Redes e Sistemas Distribuídos	OP	1 / 1S	162	T-30; PL-30; OT-4,5	6
Processamento Digital de Imagem	OP	1 / 1S	162	T-30; PL-30; OT-4,5	6
Opções A-2					
Transporte e Distribuição de Energia	OP	1 / 2S	162	T-30; PL-30; OT-4,5	6
Computação Natural	OP	1 / 2S	162	T-30; PL-30; OT-4,5	6
Computação móvel e ubíqua	OP	1 / 2S	162	T-30; PL-30; OT-4,5	6

Opções A-3					
Sistemas Energéticos Emergentes	OP	2 / 1S	162	T-30; PL-30; OT-4,5	6
Aprendizagem automática	OP	2 / 1S	162	T-30; PL-30; OT-4,5	6
Redes Avançadas de Computadores	OP	2 / 12S	162	T-30; PL-30; OT-4,5	6

T - Teóricas; PL – Práticas Laboratoriais; OT – Orientação Tutorial; O – Obrigatória; OP – Optativa

Nota: Esta é a proposta para o plano de estudos, aquando da elaboração deste documento (janeiro de 2019).

A proposta de Mestrado (2º ciclo) aqui apresentada, tem 120 ECTS, desenvolvendo-se ao longo de quatro semestres letivos (dois anos), conferindo o grau de Mestre em Engenharia Eletrotécnica e de Computadores. A estratégia adotada para a definição do conjunto de unidades curriculares oferecidas aos alunos, tem em conta o ramo de especialização escolhido, possibilitando ainda a existência de diferentes UC opcionais, em função das preferências e interesses profissionais de cada aluno. Tratando-se de um curso de 2º ciclo, pressupõe-se que os alunos admitidos, possuam uma base sólida de formação inicial na área da Engenharia Eletrotécnica e de Computadores, que lhes permita a aquisição de competências específicas na área de especialização escolhida.

Os temas a abordar no curso são variados e cobrem as áreas cientificamente emergentes, nomeadamente em termos de investigação. A UC Dissertação tem uma importância fundamental neste plano de estudos, não só pelo peso que apresenta face às restantes (33 ECTS), mas sobretudo por permitir a integração dos conhecimentos adquiridos ao longo do curso, podendo ser desenvolvida junto de empresas ou grupos de investigação.

4.1 – Objetivos de aprendizagem da Unidade Curricular PDS

Nesta Unidade Curricular pretende-se dotar os alunos de conceitos fundamentais de sinais e sistemas discretos bem como de técnicas de Processamento Digital de Sinal (PDS) nos domínios do tempo e da frequência: análise e filtragem.

Pretende-se ainda apresentar problemas relacionados com aplicações de PDS em Teoria da Informação e Telemática, Codificação Digital (áudio, imagem ou vídeo) e a Internet das Coisas (IoT).

É objetivo também desenvolver as competências para a prossecução dos estudos avançados na área de PDS.

4.2 – Conteúdos programáticos

O conteúdo programático proposto adaptado ao espírito de Bolonha onde aluno o centro de todo o processo de ensino-aprendizagem tem em conta a prossecução dos objetivos enunciados para a unidade curricular. Assim propõe-se o seguinte programa:

- **I.** Sinais e sistemas discretos (SD); Sistemas recursivos e não recursivos; Resposta impulsional; Sistemas com resposta impulsional finita (FIR) e infinita (IIR); Classificação de SD quanto à: linearidade, invariância à translação, estabilidade causalidade (SLIT). Convolução/correlação discretas; Resposta em frequência; Transformada de Fourier de sinais discretos: propriedades;
- **II.** Amostragem: reconstrução perfeita; Sistemas *sample and hold*;
- **III.** Transformada Discreta de Fourier (DFT); Transformada rápida de Fourier (FFT). Convolução circular. Técnicas de *Overlap-add* e *Overlap-save*;
- **IV.** Transformada z: Convergência, Propriedades e Inversa; Relação com a transformada de Fourier; Função de transferência;
- **V.** Projeto de filtros digitais do tipo IIR; Invariância da resposta impulsional; Transformação bilinear.

O conteúdo programático apresentado está organizado de forma encadeada para possibilitar uma compreensão progressiva das matérias abordadas na unidade curricular e contribuir para os objetivos definidos encontrando-se estruturado em cinco grandes tópicos capítulos, abordados e explorados nas 15 semanas previstas de aulas, perfazendo um total de 30 horas teóricas e 30 horas práticas.

4.2.1 – Planificação da atividade letiva

Tal como referido anteriormente, o semestre desenvolve-se ao longo de 15 semanas. Esta UC dedicará duas horas semanais às aulas teóricas e duas às aulas práticas laboratoriais. As duas tipologias de aulas, teórica e prática laboratoriais, desenvolver-se-ão sempre em articulação para permitir aplicar os conceitos apreendidos nas aulas teóricas às aulas práticas. Os métodos de ensino utilizados são baseados nas práticas preconizadas pelo Processo de Bolonha, na experiência adquirida pelo candidato na

lecionação de unidades curriculares afins, bem como na experiência adquirida através da gestão e participação em projetos e orientação de trabalhos académicos em áreas afins. Tal, permite que os conhecimentos sejam transmitidos, aliando as melhores e mais atuais referências nesta área, com uma perspectiva pragmática e de aproximação aos contextos reais em que o aluno se vai ver envolvido no seu quotidiano profissional futuro.

Na primeira semana será efetuada a apresentação das normas de funcionamento da UC, dos objetivos a alcançar, do conteúdo programático, do método de avaliação e da bibliografia recomendada. Serão ainda apresentados os temas a desenvolver pelos alunos em grupos de dois elementos, relativos aos trabalhos finais, a desenvolver durante o semestre. Nas semanas seguintes são abordados os cinco tópicos que constam do conteúdo programático. A última aula, será dedicada à apresentação oral do projeto final por parte dos alunos, momento que se pretende seja motivador e de debate sobre os temas abordados.

As aulas Práticas Laboratoriais acompanham e complementam os temas abordados na componente teórica com a resolução de problemas e de trabalhos de grupo.

4.3 – Demonstração da coerência dos conteúdos programáticos com os objetivos de aprendizagem da UC

A realização de muitos sistemas em Engenharia pode ser aproximada por sistemas de Processamento Discreto com soluções semelhantes às obtidas com Processamento Analógico.

Muitos dos tópicos abordados nas Unidades Curriculares habituais nos primeiros anos duma licenciatura na área, nomeadamente, Matemática, Sistemas de Sinais, Eletrónica ou Sistemas de Controlo, cujos conteúdos estão predominantes orientados para nos sistemas analógicos, servem de referência e constituem os antecedentes fundamentais para que a aproximação das soluções preconizadas nos conteúdos de PDS sejam devidamente estabelecidas e medidas em laboratório sendo assim confrontadas com as dos sistemas analógicos equivalentes (desenho, tamanho ou custo computacional).

Como já referido e de modo enriquecer a coerência das metodologias de ensino fomentando o progresso na aprendizagem, foram desenvolvidos ao longo dos anos uma

série de Tutoriais, criados um conjunto de Seminários Temáticos proferidos por especialista a nível nacional bem como estimulada a realização/assistência de cursos *on-line* síncronos e/ou assíncronos ao longo da leção dos conteúdos programáticos legado que contribuiu em boa medida para a instalação em 2018 do *Calibration Audio MIDI Lab* (CAM_LAB@utad.pt) no Edifício de Engenharias I da UTAD sob a responsabilidade do autor que pretende entre outros desígnios, servir os alunos que pretendam cumprir todas as etapas para o sucesso numa UC de PDS onde acomoda entre outros, alguns protótipos e experiências¹, artigos científicos que foram passando de geração em geração de alunos há praticamente trinta anos de experiência de ensino na área (ver Anexo_D)

4.4 – Metodologias de Ensino (Avaliação incluída)

A apresentação de conteúdos teóricos é acompanhada com sessões de resolução de problemas que envolvem a discussão dos conceitos, resultados e aplicações, onde se pretende que os estudantes desempenhem um papel ativo em todo o método de ensino pois só completando as propostas de aprendizagem diversificadas e distribuídas ao longo de todo o período de aulas que compreendem a realização dos tutoriais, a participação em Seminários temáticos bem como o acompanhamento de cursos *on-line* síncronos ou assíncronos e mesmo a participação em fóruns, se tornarão alunos mais competentes e habilitados para concluir com sucesso a Unidade Curricular.

O sistema de avaliação que a seguir se descreve está de acordo com o Regulamento Pedagógico da UTAD (Regulamento n.º 136/2018, publicado na 2ª série do Diário da República N.º 41, de 27 de fevereiro de 2018). De acordo com aquele regulamento, o aluno terá à disposição 3 modos de avaliação: Avaliação Contínua (Modo 1), Avaliação Complementar (Modo 2) e Avaliação por Exame Final (Modo 3). Será, no entanto, dada preferência ao Modo 1, Avaliação Contínua. Para realizar qualquer das provas de avaliação previstas, o aluno deverá, obrigatoriamente, proceder à sua inscrição no SIDE (Sistema de Informação de Apoio ao Ensino, <https://side.utad.pt/>) até 48 horas antes da

¹<https://www.youtube.com/watch?v=jpiqimVV2Zc&v=1s>
https://www.youtube.com/watch?v=ZuHLA_nX1iuc
<https://www.youtube.com/watch?v=QKRHtnXMumc>
<https://www.youtube.com/watch?v=8IQSNHIJQ7k>

realização da prova. Esta imposição prende-se com o facto de, desta forma, ser possível otimizar os recursos a afetar/disponibilizar.

4.4.1 – Obtenção de frequência à unidade curricular

Independentemente do modo de avaliação escolhida pelo aluno, este só poderá ser avaliado se tiver obtido frequência à UC. Considera-se que o aluno obteve frequência à UC quando frequentar no mínimo 70% das horas de contacto previstas a cada componente (70% TP + 70% PL). Constituem exceção a esta regra os alunos que se encontrem nas condições definidas nos n.ºs 1 e 2 do Artigo 19º das Normas Pedagógicas da UTAD. Os alunos com regimes especiais deverão contactar o docente nos primeiros dez dias úteis do semestre para definir o sistema de avaliação que lhes será aplicado. No caso em que o aluno não está matriculado na UC pela primeira vez, considera-se que este obteve frequência, caso tenha realizado, com classificação igual ou superior a 8,5 valores, a componente prática (trabalhos práticos) no ano letivo anterior.

4.4.2 – Sistema de Avaliação

A avaliação da UC é contínua (exceto nos casos previstos nas Normas Pedagógicas em vigor na UTAD), sendo dividida em duas componentes:

1. Componente teórica (NT), que incentive o aluno a fazer um estudo sobre toda a matéria lecionada: com dois momentos de avaliação na forma de testes escritos individuais compostos por questões de índole teórica com inclusão de problemas teórico-práticos, estando formatados para uma duração média de uma hora e trinta minutos. A média aritmética obtida nesta provas contará com **70% para o valor da classificação final (NF)**. As provas de avaliação escrita podem ser realizadas preferencialmente em avaliação contínua ou em exame, caso o aluno não tenha obtido aprovação por frequência e esteja admitido a exame, ou caso tenha um estatuto especial. No Anexo_A, apresentam-se, a título de exemplo, duas provas de avaliação contínua (Frequências) a primeira tipicamente a meio do semestre incidindo *Análise no Tempo* e a segunda no fim do semestre é mais focada nos tópicos de *Análise em Frequência*, bem como um exemplo de um Exame Global. Refira-se ainda que todos os anos são disponibilizados conjunto de exercícios teórico práticos propostos logo no

início do semestre e abrangem todas as componentes das matérias lecionadas. Outros recomendados para auto-estudo normalmente servem de referência para os problemas mais complexos que surgem nos exames em cada ano (ver Anexo_B);

2. Componente prática (NP), que englobe os diferentes assuntos estudados ao longo do semestre letivo através de uma prática experimentada que pode ser complementada por prática simulada. Os alunos em grupo de dois (2) terão que realizar três trabalhos práticos, onde dois nas aulas laboratoriais de contacto (PDS_Lab1 e PDS_Lab 2) e orientados um Projeto final extra-aula (PDS_Lab3) a apresentar no fim do semestre, onde a classificação obtida na prática contará com **30% para o valor da classificação final (NF)**.

Todos os trabalhos serão são documentados com um relatório escrito e alvo de apresentação por grupo em três sessões de aprendizagem colaborativa a meio do semestre, no fim e um outro final antes do semestre terminar durante numa das datas da Época de Exames: regra geral na época Normal permitindo o enriquecimento ainda durante a Época de Recurso para os grupos que assim o entenderem. Cada trabalho será avaliado numa escala de zero a vinte valores e contribuem parceladamente para a classificação final prática da seguinte forma:

$$\text{NP} = 25\% \text{ PDS_Lab1} + 25\% \text{ PDS_Lab2} + 50\% \text{ PDS_Lab3}$$

A assiduidade e empenho na preparação e desenvolvimento dos trabalhos serão classificados por observação direta do professor (avaliação contínua de desempenho).

Os temas dos trabalhos a desenvolver são os seguintes onde os dois primeiros PDS_Lab1 e PDS_Lab2 mandatórios e o PDS_Lab3 acordado de entre os tópicos abaixo apresentados ou proposto por cada Grupo de Alunos de acordo com os seus interesses em particular:

PDS_Lab1 – *A realização das Convolução e Correlação Discretas;*

PDS_Lab2 – *Realização da Transformada Rápida de Fourier com a função `fft.m` do MatLab;*

PDS_Lab3 – *Descodificando Morse / OuvER Imagens Meteorológicas – WEFAX / A Sinalização telefónica e a Descodificação de DTMF / Identificação de filtros FIR e IIR usando o método dos Mínimos Quadrados (LMS) / Proposto pelos alunos.*

No Anexo_C estão reproduzidos alguns exemplares dos protocolos desenvolvidos.

Modo 1: Avaliação Contínua

Conforme referido anteriormente, neste modo de avaliação, o aluno será avaliado através da realização de dois testes realizado no final do período letivo, que será composto por perguntas de índole teórica com inclusão de problemas teórico-práticos, estando formatado para uma duração média de uma hora e trinta minutos. O aluno ficará aprovado, neste regime, se obtiver uma classificação final (**NF**) superior ou igual a 9,5 valores quando aplicada a média ponderada da Nota Teórica (70%), Classificação da Nota Prática (30%). Ou seja, para apuramento da nota final (**NF**), será aplicada a seguinte expressão:

$$NF = 0.7 * NT + 0.3 * NP$$

Onde:

NT – Classificação das Frequências;

NP – Classificação da Componente Prática;

Modo 2: Avaliação Complementar

Podem submeter-se a este modo de avaliação os alunos que, não tendo obtido aprovação no Modo 1, cumpram as condições de frequência e admissão à avaliação complementar, Época Normal ou de Recurso, previstas no regulamento pedagógico. Os alunos podem submeter-se a nova avaliação apenas nas componentes a que obtiverem classificação negativa inferior ao mínimo exigido no Modo 1. A Nota Final é calculada através da expressão:

$$NF = 0.7 * NT + 0.3 * NP$$

Onde:

NT – Classificação da Avaliação Complementar;

NP – Classificação da Componente Prática;

Modo 3: Avaliação por Exame

Os alunos que não tenham obtido aprovação no Modo 1 mas que cumpram os critérios de admissão ao Modo 2, ou que possam optar por este modo, no caso dos

alunos com estatuto especial, são admitidos ao Modo 3, Avaliação por Exame. A prova de exame é composta por perguntas de índole teórica com inclusão de problemas teórico-práticos, estando formatada para uma duração média de uma hora e trinta minutos, sendo classificada para 20 valores. Aos alunos com estatuto especial e que se tenham proposto ao Modo3, conjuntamente com esta prova ser-lhes-á exigida a realização da Componente Prática semelhante à realizada em modo síncrono mas ajustada agora de maneira assíncrona dependendo dos casos. A Nota Final é calculada através da expressão:

$$NF = 0.7 * NT + 0.3 * NP$$

Onde:

NT – Classificação do Exame;

NP – Classificação da Componente Prática;

Nota: As omissões a este documento regem-se pelas disposições dos estatutos da UTAD.

5 – BIBLIOGRAFIA RECOMENDADA PARA A UC PDS

De modo a concluir com sucesso a Unidade Curricular, como já referido, propõem-se um conjunto alargado de propostas de aprendizagem que compreendem a participação em Seminários temáticos, o acompanhamento de *links/cursos on-line* síncronos ou assíncronos, a realização Tutoriais (Anexo_D), bem como todo o material Pedagógico de apoio que foi desenvolvido ao longo dos anos onde se contam Apontamentos para apoio às Aulas Teóricas, Aulas Teórico-Práticas com a produção de um conjunto de exercícios Teórico-Práticos a apresentar nas aulas de contacto e a realizar em ensino autónomo e a entregar no fim do semestre (Anexo_B), aos Mini-Projetos a realizar durante as aulas laboratoriais, Projeto Final (Anexo_C) bem como referências a alguns Artigos científicos sucedâneos no âmbito de trabalho desenvolvido em PDS já publicados (Anexo_E).

- **Bibliografia Teórica**
 - Kuc, R., (1988) – *Introduction to Digital Signal Processing* – McGraw-Hill Education (ISE Editions); International Ed edition, ISBN: 978-0071005432;
 - Oppenheim, A., Shaffer, R., (2009) – *Discrete-Time Signal Processing* – Prentice-Hall Signal Processing Series, ISBN: 978-0131988422;
 - Apontamentos do Professor;
- **Apoio Prática e Teórico-Prática**
 - Ingle, V., Proakis, J., (2007) – *Digital Signal Processing using MatLab* – Vinay K. Ingle, John G. Proakis, CEngage Learning ISBN: 978-9386668110;
 - Ambardar, A., Borghesani, C., (1997) – *Mastering DSP Concepts using MatLab* –, Ashok Ambardar, Prentice Hall, ISBN: 978-0135349762;
- **Seminários Temáticos** – (*Audio Talks*- Grupo de Áudio da AES/NEUTAD)
 - Soares, S., (2011-2020) – *A Audição e o Sistema Auditivo Humano; Áudio, do Analógico ao Digital* – UTAD;
 - Pereira, L., (2011-2020) – *How Rooms acts as a Sound Equalizer* – Diretor da Yacoustic, Lda, UK;
 - Soares, S., (2013-2020) – *A norma MIDI: uma aplicação de síntese de um clarinete num canal em MatLab* – UTAD;
 - Fonseca, N., (2017-2020) – *All you need to know about 3D Audio – Sound Particles CEO*;
 - Pinho, A., (2019) – *Coding and Compression Data – IEETA Director*;
 - Moreira, E., (2019) – *Audio Production Equipment, Systems and Techniques* – Produtor musical na Ruby Discos,
 - Silva, N., Silva, P., (2020) – *Audio and Video on Television* – *Audio Talks-2020* – Operadores de Áudio/Vídeo na RTP-Norte.
- **Bibliografia Complementar**
 - Lima, J.J. Pedroso (2012) – *Ouvindo Onda e Vibrações. Aspectos físicos e biofísicos* – Imprensa da Universidade de Coimbra, ISBN:978-989-26-0143-4;
 - Fonseca, N., (2012) – *Introdução à Engenharia do Som* – FCA, Editora Informática ISBN: 9789727227280;
 - Silva, S., Valente, A., Soares, S. et al, (2016) – *Morse Code Translator Using the Arduino Platform: Crafting the Future of Microcontrollers* – Proceedings of Science na Information Conference 2016, London, United Kingdom. ISBN: 9781467384605. DOI: 10.1109/SAI.2016.7556055.
 - Silva, S., Soares, S., et al (2015) – *Digital Sound Processing using Arduino and MatLab* – Proceedings of Science Information Conference SAI 2015 London, United Kingdom. ISBN:9781479985463. IEEE Catalog Number: CFP 15SAAUSB.

- Links

- Demeine, E., (2015) – *Divide&Conquer:FFT* – <https://www.youtube.com/watch?v=iTMn0Kz18tg>, MIT OpenCourseWareVideo course;
- <https://monocasual.github.io/vocoder/>, Vocoder on-line;
- <https://www.kevraudio.com/forum/viewtopic.php?t=150348>, How to build a drum kit with Adobe;
- <https://www.youtube.com/watch?v=bTqn7SeQQ7g>, Implantes Cocleares - C.H.Coimbra;
- <https://www.youtube.com/watch?v=gWpWlQw7HWU>, Audio Dither Explained
- <https://www.youtube.com/watch?v=1C8yfeature=endscreen&v=gfsennndNQ8>, Ringtones no adults.

6 – CONSIDERAÇÕES FINAIS

Como já referido, nos nossos dias o desenvolvimento continua acelerado da área do PDS e vai muito para além da área da Electrotecnia onde inúmeras aplicações têm surgido nas mais variadas áreas desde a Medicina ou Biotecnologia até à Música passando pelas Finanças e tiram partido de algoritmos de PDS persistindo assim a exigência de conhecimentos de base em Teoria de Sinal nomeadamente no paradigma discreto aos futuros Engenheiros Electrotécnicos, Engenheiros Informáticos ou áreas emergentes como é o caso em BioInformáticaⁱⁱⁱ o que só por si confirma a necessidade fundamental da unidade curricular de **Processamento Digital de Sinal** continuar a integrar o *currículum* de quaisquer cursos da Engenharia Electrotécnica que se pretenda afirmar como tal como é o caso do Mestrado em Engenharia Eletrotécnica e de Computadores (MEEC) na UTAD.

ⁱⁱⁱNo Anexo_F apresenta-se artigo *Genomic Signal Processing* proposto como projeto final de PDS-Edição 2019/2020 e realizado por alunos do curso de Mestrado de BioInformática Aplicado às Ciências da Vida

7 – REFERÊNCIAS BIBLIOGRÁFICAS

- [Cambridge, 2019] <https://www.cl.cam.ac.uk/teaching/1112/DSP/>;
- [Chandra, 2015] Chandra, V., & Mills, K. A.; (2015) – *Transforming the Core Business of Teaching and Learning in Classrooms through ICT – Technology, Pedagogy and Education*, 24(3), 285-301;
- [Costa, 2012] Costa, M.; Barroso, J.; Soares, S.; (2012) – *Signal Processing Interpolation Educational Workbench – Computer Application in Engineering Education*, 20(2), pg. 356-3633, DOI:10.1002/cae.20402;
- [ECT, 2006] Departamento de Engenharias. Dossier para Acreditação do Curso junto da Ordem dos Engenheiros. Universidade de Trás-os-Montes e Alto Douro, Abril 2006;
- [ECT, 2018] Regulamento pedagógico da UTAD. Regulamento n.º 136/2018, publicado na 2ª série do Diário da República N.º 41, de 27 de fevereiro de 2018;
- [ESSEX, 2019] <https://www1.essex.ac.uk/modules/default.aspx?coursecode=CE335&level=6&period=AU>;
- [Evans, 2013] Evans, C. (2013) – *Making Sense of Assessment Feedback in Higher Education – Review of Educational Research*, Vol. 83, No. 1, 70-120. DOI: 10.3102/0034654312474350 © 2013 AERA. (<https://journals.sagepub.com/doi/full/10.3102/0034654312474350>)
- [FEUP, 2019] https://sigarra.up.pt/feup/pt/ucurr_geral/ficha_uc_view?pv_ocorrencia_id=420274;
- [IST, 2019] <https://fenix.tecnico.ulisboa.pt/disciplinas/PDS5645111326/2018-2019/2-semester/lab-assignments>;
- [Jenkins, 2009] Jenkins, H. *et al* (2009) – *Confronting the Challenges of Participatory Culture, Media Education for the 21st Century* – The John D. and Catherine T. MacArthur Foundation Reports on Digital Media and Learning, ISBN 978-0-262-51362-3;
- [Hulme, 2020] Kukulska-Hulme, A. *et al* (2020) – *Innovating Pedagogy 2020*– Open University Innovation Report 8. Milton Keynes: The Open University;
- [Lebre, 2010] Lebre, P.; Soares, S.; Silva, S.; Silveira, L. (2010) – *Avaliação da voz após cirurgia ao NLR* – 12º Encontro da Secção Portuguesa da *Audio Engineering Society*, 8 de Outubro, UA-Aveiro;
- [Lafarriere, 2012] Lafarriere, T., Law, N., & Montané, M. (2012) – *An international knowledge building network for sustainable curriculum and pedagogical innovation* – *International Education Studies*, 5(3), 148-160;
- [Lakkala, 2015] Lakkala, M., & Ilomaki, L. (2015) – *A case study of developing ICT-supported pedagogy through a collegial practice transfer process* – *Computers & Education*, 90, 1-12. DOI: 10.1016/j.compedu.2015.09.001;
- [MIT, 2019] <https://ocw.mit.edu/resources/res-6-008-digital-signal-processing-spring-2011/video-lectures/>;
- [Neves, 2018] Neves, F., Soares, S., Assunção, P. (2018) – *VoIP Technologies - Chapter Title: Enhanced VoIP by signal reconstruction and voice*

- quality assessment* – Book edited by Shigeru Kashiara. Intech -Open Access Publisher, Chapter 3, pg. 45-78. ISBN 978-1-4244-2422-1;
- [Neves, 2008] Neves, F.; Soares, S.; Reis, M. J.C.S.; Tavares, F.; Assunção, P.. (2008) – *VoIP Reconstruction Under a Minimum Interpolation Algorithm* – In 2008 IEEE International Symposium on Consumer Electronics, Vols. (1-2), pg. 51-53. CD-ROM. ISBN 978-1-4244-2422-1;
- [Neves, 2007] Neves, F.; Soares, S.; Assunção, P. (2007) – *O Algoritmo de Papoulis-Gerchberg e a Reconstrução de Voz Comutada* – Engenharias 07, Inovação&Desenvolvimento, CD-ROM;
- [Norman, 1996] Norman, D. *et al* (1996) – *Learner-Centered Education* – *Communications of the ACM*, 39(4), pg. 24-27. (<https://doi.org/10.1145/227210.227215>);
- [Oxford, 2019] <https://www.conted.ox.ac.uk/courses/digital-signal-processing-theory-and-application>;
- [Pereira, 2011] Pereira, L., Faustino J., Soares M., Soares S. (2011) – *Simulation of a Reverberation Room* – Forum Acusticum 2011, pg. 1947-1952 Aalborg – Denmark. Edited by Danish Acoustical Society – DAS, 27 de June – 01 July. ISBN: 978-84-694-1520-7;
- [Rabiner, 1978] Rabiner, R., Schafer, R. (1978) – *Digital Processing of Speech Signals* – Pearson; US edition, ISBN: 013213603-1;
- [Reis, 2014] Reis, M., Soares, S., Cardeal, S., Morais, R., Peres, E., Ferreira, P. (2014) – *Teaching Fourier series expansions in undergraduate education with the help of the FouSE Android application* – International Journal of Interactive Mobile Technologies (IJIM). Vol. 8(1), pg. 26-31, ISSN: 1865-7923;
- [Silva, 2017] Silva, S., Soares, S., Cabral, M., Neves, F., Assunção, P., (2017) – *A dynamic programming algorithm to select optimal high-priority voice segments using Arduino* – 17th IEEE International Conference on Smart Technologies, EUROCON 2017, Conference Proceedings, pg. 271-276, OHRID, R. Macedonia;
- [Silva, 2016] Silva, S., Valente, A., Soares, S., Cabral, M., Paiva, J., Bartolomeu, P., (2016) – *Morse Code Translator Using the Arduino Platform: Crafting the Future of Microcontrollers* – Proceedings of Science and Information Conference 2016 (SAI 2016 – www.conference.thesai.org), pg. 675-680, July 13-15, London, United Kingdom. ISBN: 978-1-4673-8460-5. DOI: 10.1109/SAI.2016.7556055;
- [Silva, 2015] Silva, S., Soares, S., Valente, A., Marcelino, S. (2015) – *Digital Sound Processing using Arduino and MatLab* – Proceedings of Science and Information Conference 2015, SAI 2015, pg. 1184-1191, July 28-3, London, United Kingdom. ISBN:978-1-4799-8546-3;
- [Silveira, 2020] Silveira, L. *et al.* Soares, S., (2020) – *Repair of the recurrent laryngeal nerve – Experimental study* – Revista Portuguesa de Cirurgia, Sociedade Portuguesa de Cirurgia, II Série, Volume 46:19-31, ISSN: 1646-6918;

[Sharples, 2019] Sharples, M. (2018) – *Practical Pedagogy: 40 New Ways to Teach and Learn* – London-Routledge, ISBN 9781138599819;

[UA, 2019] <https://www.ua.pt/#/pt/uc/2759>.

Anexo_A

Avaliações Teórica Teórico Prática

Frequências (Avaliação Contínua)
&
Exame Global

PARTE I

Indique se são Verdadeiras (V) ou Falsas (F) as seguintes afirmações

Uma das propriedades dos sistemas discretos é repetibilidade	
Num sistema recursivo, a saída $y[n]$, depende apenas da entrada $x[n]$	
Todos os sistemas realizáveis são causais	
Um filtro IIR tem resposta impulsional, $h[n]$, finita	
Num sistema linear, a saída é alterada em amplitude e em fase mantendo-se a frequência	
A transformada de Fourier de um sistema existe sempre	
$H(e^{j\omega})$, é uma função discreta de período 2π	
Um filtro anti-aliasing é um filtro passa baixo com frequência de corte igual a metade da frequência de amostragem	
A técnica de oversampling (sobre-amostragem) não contribui para um melhor desempenho dos filtros	
A frequência de Nyquist ($f_n = f_s/2$ Hz) corresponde no domínio digital a 2π	

PARTE II

1. Determine se o sistema descrito pela transformação $T\{x[n]\} = x[n]^{-1}$ é linear e estável? Justifique.
2. Seja $h[n]$ a resposta impulsional de um SD com resposta em frequência $H(e^{j\omega})$. Determine, em função de $H(e^{j\omega})$ a resposta em frequência do SD com resposta impulsional $(-1)^n h[n]$.
3. Sabendo que o sistema auditivo humano tem uma largura de banda com uma $f_{\max} \sim 20\text{kHz}$,
 - a) determine o **número mínimo de amostras** que devemos tomar para processar digitalmente um sinal áudio sem erros;
 - b) explique sucintamente para que servem os circuitos **sample&hold**.

PARTE III

4. Considere o sistema discreto (SLIT) definido pela equação às diferenças:

$$y[n] = \frac{x[n-1] + x[n] + x[n+1]}{3}$$

Determine

- a) a forma fechada da sua resposta em frequência, $H(e^{j\omega})$.
- b) o desenho gráfico do módulo e fase de $H(e^{j\omega})$.
- c) a resposta do sistema quando a entrada é $x[n] = u[n+1] - u[n-2]$.
- d) a saída do sistema quando a entrada é $x[n] = \cos\left(\frac{8\pi}{3}n\right)$.

Colaão

Parte I – 5 Parte II – 1) 2 2) 1,5 3) 1,5 4) 1,5 Parte III – 4a) 1,5 4b) 3 4c) 3 4d) 1

Formulário

Sequência

Transformada de Fourier

ℳo caso de uma sequência contínua (analógica)

$h_a(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} H(j\Omega) e^{j\Omega t} d\Omega$	$H(j\Omega)$
$h_a(t)$	$H(j\Omega) = \int_{-\infty}^{+\infty} h_a(t) e^{-j\Omega t} dt$

ℳo caso de uma sequência discreta (digital)

$h[n]$	$H(e^{jw}) = \sum_{n=-\infty}^{+\infty} h[n] e^{-jwn}$
$h[n] = \frac{1}{2\pi} \int_{-\pi}^{+\pi} H(e^{jw}) e^{jwn} dw$	$H(e^{jw})$

Algumas relações matemáticas e trigonométricas úteis

- $\sum_{n=-\infty}^{+\infty} |h[n]| = \dots, h[n_{-1}], h[n_0], h[n_1], \dots \begin{cases} S_n = u_1 \frac{1 - \text{razão}^{n^2 \text{ termos}}}{1 - \text{razão}}, \text{ progressão geométrica} \\ S_n = \frac{u_1 - u_n}{2} \times n^2 \text{ termos}, \text{ progressão aritmética} \end{cases}$
- $\cos w = \frac{e^{jw} + e^{-jw}}{2} ; \sin w = \frac{e^{jw} - e^{-jw}}{2j}$
- $H(e^{jw}) = \sum_{n=-\infty}^{+\infty} h[n] e^{-jwn} = \sum_{n=-\infty}^{+\infty} h[n] \cos(wn) + j \sum_{n=-\infty}^{+\infty} h[n] \sin(wn)$
- $H(e^{-jw}) = \sum_{n=-\infty}^{+\infty} h[n] e^{jwn} = \sum_{n=-\infty}^{+\infty} h[n] \cos(wn) - j \sum_{n=-\infty}^{+\infty} h[n] \sin(wn)$
- $H(e^{jw}) = H_R(e^{jw}) + jH_I(e^{jw})$
- $|H(e^{jw})| = \sqrt{H(e^{jw})H^*(e^{jw})}$
- $H(e^{jw}) = \frac{N(e^{jw})}{D(e^{jw})}, \begin{cases} |H(e^{jw})| = \sqrt{\frac{N(e^{jw})N^*(e^{jw})}{D(e^{jw})D^*(e^{jw})}} \\ \arg H(e^{jw}) = \tan^{-1} \frac{H_I(e^{jw})}{H_R(e^{jw})} \end{cases}$

PARTE I

Indique se são Verdadeiras (V) ou Falsas (F) as seguintes afirmações

A transformada de Fourier discreta (DFT) é a amostragem da Transformada de Fourier em N pontos desigualmente espaçados	
O produto de DFT equivale no tempo à convolução circular	
Não é possível calcular uma DFT de tamanho 2N à custa de duas de tamanho N	
O produto de DFT pode em certas condições ser aproximado à convolução linear	
A complexidade computacional da FFT é $\mathcal{O}(N \log_2 N)$	
A Transformada inversa de z é única	
Um sistema que tenha transformada de Z é sempre estável	
Um filtro FIR, regra geral tem menor custo computacional quando comparado com IIR	
Num filtro FIR é possível encontrar a Forma Direta II (minimiza os atrasos)	
A transformação bilinear é uma das metodologias para sintetizar um filtro IIR	

PARTE II

1. Indique para que serve a DFT e explique em que condições existe *aliasing* no domínio dos tempos?
2. Mostre que a transformada Z da parte real de, $x[n]$

$$\frac{1}{2}[X(z) + X^*(z^*)]$$

3. Pretende-se calcular a convolução de dois sinais discretos $x[n]$ e $h[n]$ com comprimento 100 e 500 respetivamente. **Compare sob o ponto de vista do n° de operações aritméticas** o cálculo direto e o cálculo recorrendo ao algoritmo da *fft*.

PARTE III

4. Considere o **SDLI** definido pela função de transferência,

$$H(z) = \frac{1 - a^{-1}z^{-1}}{1 - az^{-1}}, a \in \mathbb{R}$$

- Para que gamas de valores de “ a ” o sistema é **estável**? Justifique
- Para $0 < |a| < 1$, represente o diagrama de **pólos e zeros** bem como a região de convergência (**ROC**) considerando o sistema **causal**.
- Admitindo o sistema causal, determine as **duas primeiras** amostras da resposta do sistema à entrada

$$x[n] = \delta[n + 1] - \delta[n - 1]$$

- Considere a seguinte equação às diferenças.

$$y[n] - \frac{3}{4}y[n - 1] + \frac{1}{8}y[n - 2] = x[n] + \frac{1}{3}x[n - 1]$$

Apresente o diagrama de blocos que **realiza o sistema**

- na forma que **minimize os atrasos**;
- em associação série (cascata) apenas com **seções de 1ª ordem**

PARTE I

Indique se são Verdadeiras (V) ou Falsas (F) as seguintes afirmações

Num sinal digital a variável dependente é função do número de bits de quantização	
Uma das propriedades dos sistemas discretos é a imunidade ao ruído elétrico	
Em todos os sistemas causais a saída só depende de entradas .	
Um filtro digital FIR é constituído por resistências e condensadores	
Um sistema linear é aquele em que se pode aplicar o teorema da sobreposição, $T\{\alpha x_1[n] + \beta x_2[n]\} = \alpha T\{x_1[n]\} + \beta T\{x_2[n]\}$	
Um SDLI é estável se $b[n]$ for absolutamente integrável	
Existem sistemas realizáveis não causais .	
A resposta em frequência de um SDLI, $H(e^{j\omega})$ é igual à transformada de Fourier da sua resposta impulsional, $b[n]$.	
A resposta em frequência é uma função contínua e de período 2π	
A frequência de amostragem corresponde, no domínio digital, a 2π .	

PARTE II

1. Indique **duas formas** diferentes de **descrever** um sistema discreto colocando em evidência as **relações** entre elas.
2. Determine se o sistema descrito pela transformação $T\{x[n]\} = x[n]^2$ é **linear**? Justifique.
3. Admita que uma sequência $x[n]$ tem a transformada de Fourier, $X(e^{j\omega})$. Relacione a transformada de Fourier de $x^*[n]$ com $X(e^{j\omega})$.
4. Da amostragem a 10kHz de um determinado sinal contínuo $x_c(t)$ resultou o sinal discreto $x[n] = \sin\left(\frac{\pi}{6}n\right)$. Determine dois possíveis sinais contínuos que satisfaçam essa condição, um deles correspondendo à amostragem **com aliasing** e outro à amostragem **sem aliasing**.

PARTE III

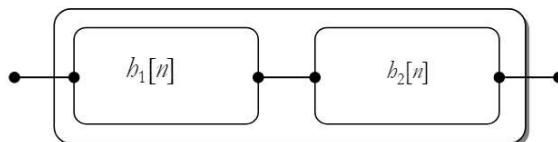
5. Suponha o sistema discreto definido pela equação às diferenças

$$y[n] = \frac{x[n] + 2x[n-1] + x[n-2]}{4}.$$

Determine:

- a) a forma fechada da sua resposta em frequência $H(e^{j\omega})$.
- b) graficamente (**módulo e fase**) de $H(e^{j\omega})$.
- c) a resposta impulsional **dum sistema** constituído pela **associação série** de dois sistemas idênticos a $y[n]$ sabendo que

$$h_T[n] = b_1[n] * b_2[n]$$



PARTE I

Indique se são Verdadeiras (V) ou Falsas (F) as seguintes afirmações

A DFT é uma função contínua e periódica de período N (N -nº de pontos da DFT)	
No algoritmo da <i>fft</i> a ordem das amostras deve ser em <i>bit reverse</i>	
Na sequência sinusoidal $x[n] = \cos(\Omega n T + \phi)$ a primeira amostra da DFT é zero	
Zeros são os zeros do numerador da função de Transferência	
Com a expansão em frações parciais é possível calcular a transformada inversa de z	
Quando existe a transformada de Fourier, a ROC da transformada de z inclui a circunferência de raio unitário	
A ROC de um sistema causal é para fora do pólo mais interior	
A associação em cascata (série) de dois sistemas, $H_1(z)$ e $H_2(z)$, equivale à sua soma $H_1(z) + H_2(z)$	
Um filtro FIR é sempre estável	
O método da invariância da resposta impulsional é um método para sintetizar filtros digitais IIR	

PARTE II

- Comente a seguinte informação: "Um filtro *anti-aliasing* é um filtro passa baixo com frequência de corte igual a metade da frequência de amostragem (frequência de Nyquist)".
- No cálculo da DFT de uma sequência, em que condições a frequência de Nyquist **não é observada**: para um n° de pontos da DFT, **par** ou **ímpar**? Justifique. (Relembre o trabalho nº2, "A realização da *fft* do MatLab").
- Prove que sendo $w_N = e^{-j\frac{2\pi}{N}}$,

$$w_N^{k(N-n)} = (w_N^{kn})^*$$
- O que é um **filtro digital**? Porque é que os **filtros FIR** não se conseguem representar em configurações **mais eficientes**, quando comparados com os IIR? Justifique.

PARTE III

10. Considere o SDLI definido pela função de transferência,

$$H(z) = \frac{1 - a^{-1}z^{-1}}{1 - az^{-1}}, a \in \mathbb{R}$$

- Para que gamas de valores de “ a ” o sistema é **estável**? Justifique
- Para $0 < |a| < 1$, represente o diagrama de **pólos e zeros** bem como a região de convergência (**ROC**) considerando o sistema **causal**.
- Admitindo o sistema causal, determine as **duas primeiras** amostras da resposta do sistema à entrada

$$x[n] = -\delta[n] + \delta[n-2]$$

- Considere a seguinte função de transferência.

$$H(z) = \frac{(1 - z^{-1})^3}{1 - \frac{5}{8}z^{-1} + \frac{1}{16}z^{-2}}$$

Apresente o diagrama de blocos que **realiza o sistema**

- na forma que **minimize os atrasos**;
- numa estrutura paralela (apenas com **seções de 1ª ordem**).

Anexo_B

Exercícios Teórico-Práticos
(de contacto e para auto-estudo)

&

Propostos para realização extra-aulas
(a entregar aquando da Defesa dos projetos finais)

SINAIS E SISTEMAS DISCRETOS (SD)

Sinais e sistemas discretos; Sistemas recursivos e não recursivos; Resposta impulsional; Classificação de Sistemas quanto à Linearidade, Invariância à Translação, Estabilidade e Causalidade; Sistemas com resposta impulsional finita (FIR) e infinita (IIR); Convolução discreta.

1. Determine a resposta impulsional, $h[n]$, dos seguintes sistemas discretos:

a) $y[n] = 0.4x[n] + 0.3x[n-1] + 0.2x[n-2] + 0.1x[n-3]$

b) $y[n] = x[n] + ay[n-1]; a \in \mathbb{R}$.

2. Verifique se os seguintes sistemas são ou não lineares:

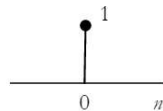
a) $y[n] = \frac{1}{3}(x[n+1] + x[n] + x[n-1])$

b) $y[n] = x^2[n]$.

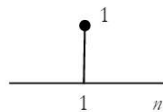
3. Calcular a resposta impulsional do seguinte sistema, para os impulsos:

$$y[n] = \begin{cases} \frac{1}{n+1}y[n-1] + x[n] & ; \quad n \geq 0 \\ 0 & ; \quad - \end{cases}$$

a) $\delta[n]$



b) $\delta[n-1]$



4. Considere os seguintes sistemas:

a) $T(x[n]) = \alpha x[n] + \beta; \alpha, \beta \in \mathbb{R}$

b) $T(x[n]) = x[n - n_0]; n_0 \in \mathbb{N}$

Conclua acerca da sua estabilidade, causalidade, linearidade e invariância à translação.

5. Admita um SD com resposta impulsional $h[n] = 2^{-n}u[n]$. Qual será a saída do sistema quando a entrada é $x[n] = u[n] - u[n - 8]$.

6. Demonstre que se $h[n] = \alpha^n u[n]$ e $x[n] = \beta^n u[n]$ a saída do sistema é

$$y[n] = (k_1 \alpha^n + k_2 \beta^n) u[n], \text{ onde } k_1 = \frac{\alpha}{\alpha - \beta} \text{ e } k_2 = -\frac{\beta}{\alpha - \beta}$$

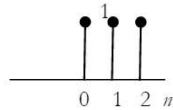
7. Admita a seguinte saída de um sistema não recursivo:

$$y[n] = \frac{1}{3}x[n+1] + \frac{1}{3}x[n] + \frac{1}{3}x[n-1].$$

Calcule a resposta desse sistema quando a entrada é:

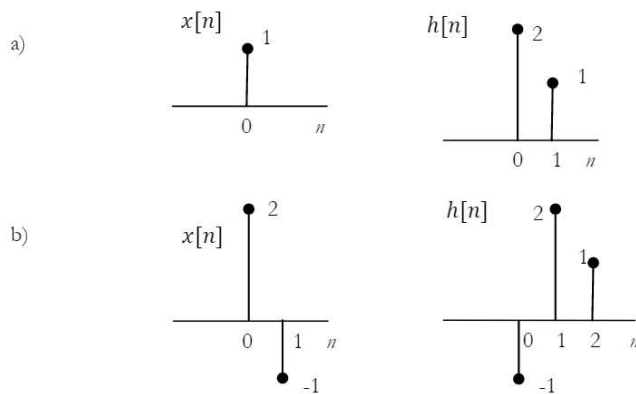
a) $x[n] = \delta[n]$

b) $x[n] = \delta[n] + \delta[n-1] + \delta[n-2]$



c) Verifique a estabilidade deste sistema.

8. Para as seguintes sequências, use a convolução linear discreta para calcular a resposta à entrada $x[n]$ quando a resposta impulsional é a indicada:



9. Considere o sistema discreto

$$y[n] = \frac{3x[n] + 2x[n-1] + x[n-2]}{6}$$

a) Determine a sua resposta impulsional $h[n]$.

b) Qual será a resposta do sistema quando a entrada é $x[n] = u[n] - u[n-4]$.

RESPOSTA EM FREQUÊNCIA E TRANSFORMADA DE FOURIER

Transformada de Fourier de sinais discretos: representação gráfica, módulo/fase; Transformada inversa de Fourier; A Resposta em Frequência $H(e^{j\omega})$ e a transformada de Fourier da resposta impulsional; Associação série e paralelo de SD; Propriedades da transformada de Fourier. Resposta em frequência de sistemas recursivos causais.

27. Calcule a transformada de Fourier dos seguintes sistemas com resposta impulsional:

- a) $h[n] = \begin{cases} \frac{1}{3} & ; \quad -1 \leq n \leq 1 \\ 0 & ; \quad - \end{cases}$
- b) $h[n] = a^n u[n]; a \in \mathbb{R}$
- c) $h[n] = r^n \cos(n \omega_0) u[n]; 0 < \omega_0 \leq 2\pi \wedge r \in \mathbb{R}$.

28. Desenhe o módulo e fase da transformada de Fourier de:

$$\delta[n-k] = \begin{cases} 1 & ; \quad n = k \\ 0 & ; \quad - \end{cases}$$

29. Calcule a resposta em frequência e represente-a em módulo e fase para o seguinte sistema definido pela equação às diferenças:

$$y[n] = \frac{1}{4}(x[n] + 2x[n-1] + x[n-2])$$

30. Calcule a resposta em frequência e represente-a graficamente, do sistema que tem a seguinte resposta impulsional:

$$h[n] = \begin{cases} \frac{1}{3} & ; \quad 0 \leq n \leq 2 \\ 0 & ; \quad - \end{cases}$$

31. Calcule a resposta impulsional do sistema que tem como resposta em frequência:

$$H(e^{j\omega}) = \begin{cases} 1 & ; \quad |\omega| \leq \omega_0 \\ 0 & ; \quad - \end{cases}$$

32. Calcular a resposta impulsional do filtro passa-banda ideal. (*Sugestão: partir do filtro passa-baixo ideal*)

33. Desenhe o módulo e a fase do seguinte sistema:

$$H(e^{j\omega}) = \frac{1}{3}(1 + 2\cos\omega)$$

34. Represente graficamente o módulo e fase da saída do sistema que tem como resposta impulsional:

$$h[n] = \begin{cases} a^n & ; \quad n \geq 0 \\ 0 & ; \quad - \end{cases}$$

quando a entrada é

$$x[n] = \begin{cases} \frac{1}{3} & ; \quad -1 \leq n \leq 1 \\ 0 & ; \quad - \end{cases}$$

35. Calcule a resposta em frequência do seguinte sistema e represente-a graficamente:

$$h[n] = \begin{cases} -1 & ; n = -2 \\ 1 & ; n = 2 \\ 0 & ; - \end{cases}$$

36. Suponha que utiliza o sistema discreto $y[n] = \frac{1}{T}(x[n] - x[n-1])$ para aproximar o diferenciador analógico $H_a(j\Omega) = j\Omega$.

- Determine a resposta em frequência $H(e^{j\omega})$ do sistema discreto.
- Represente graficamente (módulo e fase) $H(e^{j\omega})$ e $H_a(j\Omega)$ indicando em que condições a aproximação se pode considerar razoável.
- Determine a resposta impulsional do sistema discreto constituído pela associação em série de dois sistemas idênticos ao dado.

37. A parte real da transformada de Fourier de um sinal causal é: $X_R(e^{j\omega}) = 1 + \cos \omega$.

Determine a parte imaginária dessa mesma transformada $X_I(e^{j\omega})$.

38. Calcule a transformada inversa de $H(e^{j\omega}) = \frac{1}{3}(1 + 2\cos\omega)$.

39. Considere o sistema discreto:

$$y[n] = \frac{x[n] + x[n+3]}{2}$$

- Determine a sua resposta impulsional.
- Determine a sua resposta em frequência e represente-a graficamente.

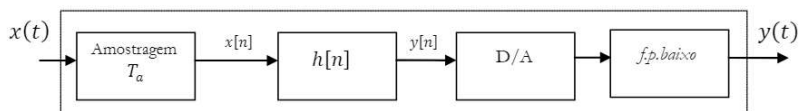
TEOREMA DA AMOSTRAGEM

A relação entre os espectros do sinal contínuo e do sinal discreto obtido a partir da amostragem. Relação entre a frequência analógica (Ω) e a frequência digital (ω): $\omega = \Omega T$ rad. Critério para reconstrução perfeita: $f_a \geq 2f_{\max}$. Frequência crítica (Nyquist). Aliasing no domínio da frequência; Reconstrução do sinal analógico: circuitos "sample and hold".

23. Enuncie em que condições um sinal contínuo $x_a(t)$ pode ser representado por um discreto $x[n]$ obtido por amostragem sem *aliasing*.

24. Da amostragem a 1000Hz de um determinado sinal contínuo $x_a(t)$, resultou o sinal discreto $x[n] = \cos\left(\frac{\pi}{8}n\right)$. Determine dois possíveis sinais contínuos que satisfaçam essa condição, um deles correspondente a uma amostragem sem *aliasing* e outro à amostragem com *aliasing*.

25. O diagrama de blocos representa um filtro digital, onde T_a é o intervalo entre cada amostra (assumir que T_a é suficientemente pequeno para prevenir o *aliasing*).



- a) Se $h[n]$ tiver uma frequência de corte $\frac{\pi}{8}$ rad e $\frac{1}{T_a} = 10$ kHz, qual será a frequência de corte do sistema?
- b) E se $\frac{1}{T_a} = 20$ kHz?

26. Considere o filtro analógico passa-baixo RC elementar com uma frequência de corte a 3dB f_c com a resposta impulsional $h_a(t)$ e o sistema discreto com resposta impulsional $h[n]$ relacionada com $h_a(t)$ por $h[n] = Th_a(t)_{t=nT}$:

- Determine a resposta em frequência $H(e^{j\omega})$ do sistema discreto.
- Represente graficamente o $|H(e^{j\omega})|$ para $\frac{1}{T} = 2f_c, 4f_c, 8f_c$
- Determine a equação às diferenças que realiza o sistema discreto.

TRANSFORMADA DE Z

Transformada de Z: representação gráfica dos pólos e zeros e Região de Convergência (ROC); Estabilidade no domínio Z. Transformada inversa de Z: método da divisão longa, da decomposição em frações parciais e do integral linha (resíduos); A relação com a transformada de Fourier.

27. Calcule a transformada Z das seguintes sequências:

- a) $h[n] = \begin{cases} \frac{1}{3} & ; \quad -1 \leq n \leq 1 \\ 0 & ; \quad - \end{cases}$
- b) $h[n] = \begin{cases} a^n & ; \quad n \geq 0 \\ 0 & ; \quad - \end{cases}$
- c) $h[n] = \begin{cases} r^n \cos(\omega_0 n) & ; \quad n \geq 0 \\ 0 & ; \quad - \end{cases} \quad (0 < \omega_0 \leq 2\pi \wedge r \in \mathbb{R})$

28. Calcule, aplicando as propriedades da transformada Z, a saída do sistema

- a) $y[n] = \frac{1}{3}(x[n+1] + x[n] + x[n-1])$
- b) $y[n] = x[n] + ay[n-1]$
- c) $y[n] = a_1 y[n-1] + a_2 y[n-2] + b_0 x[n] + b_1 x[n-1]$

29. Calcular a região de convergência (ROC) e a transformada de Z para as seguintes funções:

- a) $\delta[n]$
- b) $-\left(\frac{1}{2}\right)^n u[-n-1]$
- c) $\left(\frac{1}{2}\right)^n u[-n]$
- d) $\delta[n-1]$
- e) $\delta[n+1]$
- f) $\left(\frac{1}{2}\right)^n (u[n] - u[n-10])$

30. Analise o seguinte sistema quanto à estabilidade:

$$y[n] = x[n] + ay[n-1]$$

31. Determine a ROC e a resposta impulsional do sistema discreto com função de transferência:

$$H(z) = \frac{1 - z^{-1}}{1 - 2z^{-1} + 0.75z^{-2}}$$

a) De modo que o sistema seja causal.

b) De modo que o sistema seja estável.

32. Calcule a transformada inversa de Z da seguinte função limitada à esquerda:

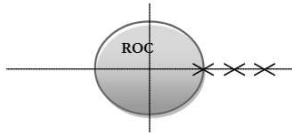
$$X(z) = \frac{-8z^{-1} + 18}{z^2 - 5z^{-1} + 6}$$

33. Sendo

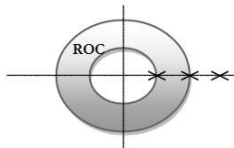
$$X(z) = \frac{z}{z - \frac{1}{2}} + \frac{z}{z - \frac{1}{3}} + \frac{z}{z - \frac{1}{4}}$$

calcule a transformada inversa de Z nos quatro casos seguintes:

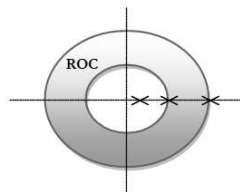
a)



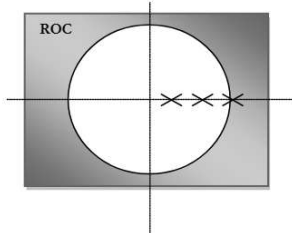
b.



c)



d)



34. Determine a transformada inversa de Z , usando (1) o método do integral linha; (2) o método da decomposição em frações parciais; (3) o método da divisão longa:

a) $\frac{1}{1+\frac{1}{2}z^{-1}}; |z| > \frac{1}{2}$

b) $\frac{1}{1+\frac{1}{2}z^{-1}}; |z| < \frac{1}{2}$

35. Determine a transformada inversa de Z :

$$X(z) = 1 - z^{-1} + 2z^{-2}; |z| \neq 0$$

36. Determine a transformada inversa de Z da seguinte expressão:

$$X(z) = \frac{z}{z - \frac{1}{2}} + \frac{z}{z - 2}; \frac{1}{2} < |z| < 2$$

37. Calcular a transformada inversa:

$$H(z) = \frac{1 - b^2 z^{-2}}{1 + a^2 z^{-2}}; |z| > a$$

38. Considere um SLIT causal com:

$$H(z) = \frac{1 - a^{-1} z^{-1}}{1 - a z^{-1}}; a \in \mathbb{R}$$

- Para que valores de a o sistema é estável?
- Se $0 < |a| < 1$, desenhe o diagrama de pólos e zeros indicando a ROC.
- $H(z)$ deve ser colocado em série com outro sistema de função de transferência $H_2(z)$ de tal forma que a função de transferência total seja unitária.
Admitindo $0 < |a| < 1$ e $H_2(z)$ é estável, determine a resposta impulsional $h_2[n]$.
- Demonstre graficamente que $H(z)$ é um passa-tudo, isto é, que a magnitude da resposta em frequência é constante.

39. Calcular $h[n]$ pelo método da decomposição em frações parciais:

$$H(z) = \frac{z^{-1}}{1 - 2.5z^{-1} + z^{-2}}; \quad \frac{1}{2} < |z| < 2$$

40. Analise a seguinte função de transferência quanto à estabilidade.

$$H(z) = \frac{z^{-1}}{1 - 2.5z^{-1} + z^{-2}}$$

Calcule $h[n]$ pelo método dos resíduos, considerando a função estável.

41. A função de transferência de um SD causal é:

$$H(z) = \frac{0.375}{z^2 - 0.75z + 0.125}$$

- Escreva a equação às diferenças que realiza o sistema.
- Verifique se o sistema é estável.
- Determine a sua resposta impulsional.
- Represente graficamente, de uma maneira aproximada, a amplitude da sua resposta em frequência.

EXERCÍCIOS PROPOSTOS – REALIZAÇÃO EXTRA-AULA

Prazo de entrega durante a época de exames do 1º semestre

1. Sabendo que por definição a convolução discreta bidimensional é

$$y[m, n] = x[m, n] \otimes h[m, n] = \sum_{k=-\infty}^{+\infty} \sum_{r=-\infty}^{+\infty} x[k, r] h[m - k, n - r]$$

calcule a convolução de

$$\begin{array}{c}
 x[m, n] \quad \uparrow m \\
 \begin{array}{cccccc}
 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 1 & 1 & 1 & 1 & 0 \\
 0 & 1 & 1 & 1 & 1 & 0 \\
 0 & 1 & 1 & 1 & 1 & 0 \\
 0 & 1 & 1 & 1 & 1 & 0
 \end{array} \quad \xrightarrow{n}
 \end{array}$$

$$\begin{array}{c}
 h[m, n] \quad \uparrow m \\
 \begin{array}{ccccc}
 0 & 0 & 0 & 0 & 0 \\
 0 & -\frac{1}{9} & -\frac{1}{9} & -\frac{1}{9} & 0 \\
 0 & -\frac{1}{9} & 1 & -\frac{1}{9} & 0 \\
 0 & -\frac{1}{9} & -\frac{1}{9} & -\frac{1}{9} & 0 \\
 0 & 0 & 0 & 0 & 0
 \end{array} \quad \xrightarrow{n}
 \end{array}$$

2. Seja $h_a(t)$ a resposta impulsional dum filtro linear, invariante e contínuo no tempo, e $h_d[n]$ a resposta impulsional de um filtro linear, invariante e discreto no tempo.

✓ Se $h_a(t) = \begin{cases} e^{-at} & ; \quad t \geq 0 \\ 0 & ; \quad t < 0 \end{cases}$

determine a resposta em frequência do filtro analógico $H(j\Omega)$ e esboce a respetiva magnitude (módulo).

✓ Se $h_d[n] = h_a(t)_{t=nT}$, sendo $h_a(t)$ a descrita na alínea anterior, determine a resposta em frequência do filtro digital $H(e^{j\omega})$ e esboce a sua magnitude.

✓ Para um dado valor de a , determine o valor mínimo do módulo de $H(e^{j\omega})$ em função de T .

3. Considere um SLIT com resposta impulsional:

$$h[n] = \begin{cases} a^n & ; \quad n \geq 0 \\ 0 & ; \quad - \end{cases}, \quad |a| < 1$$

e a entrada
$$x[n] = \begin{cases} 1 & ; \quad 0 \leq n \leq (N-1) \\ 0 & ; \quad - \end{cases}$$

✓ Determine a saída $y[n]$ calculando a convolução discreta entre $h[n] \otimes x[n]$.

✓ Determine a saída $y[n]$ calculando a transformada Z inversa do produto das transformadas Z da entrada e da resposta impulsional. (*admita $x[n]$ não causal*).

4. O sistema com função de transferência

$$H(z) = \frac{z^{-1} - a}{1 - az^{-1}}$$

é um *all-pass*, isto é, a resposta em frequência tem magnitude igual a 1.

- ✔ Desenhe uma realização deste sistema na forma direta II (que minimiza os atrasos).
- ✔ A equação às diferenças que relacionada a entrada e a saída do sistema *all-pass* pode ser expressa por

$$y[n] = a(y[n-1] - x[n]) + x[n-1].$$

Desenhe uma realização desta equação que requeira dois ramos de atraso mas apenas um ramo que multiplique por +1 ou -1.

- ✔ Considere outro sistema *all-pass* com função de transferência

$$X(z) = \frac{z^{-1} - a}{1 - az^{-1}} \times \frac{z^{-1} - b}{1 - bz^{-1}}$$

Desenhe uma implementação deste sistema, por associação série de duas realizações do tipo da alínea b), de tal forma que só sejam necessários 3 ramos de atraso.

Formulário

Sequência	Transformada de Fourier
O caso de uma sequência contínua (analógica)	
$h_a(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} H(j\Omega) e^{j\Omega t} d\Omega$	$H(j\Omega)$
$h_a(t)$	$H(j\Omega) = \int_{-\infty}^{+\infty} h_a(t) e^{-j\Omega t} dt$

O caso de uma sequência discreta (digital)

$h[n]$	$H(e^{jw}) = \sum_{n=-\infty}^{+\infty} h[n] e^{-jwn}$
$h[n] = \frac{1}{2\pi} \int_{-\pi}^{+\pi} H(e^{jw}) e^{jwn} dw$	$H(e^{jw})$
$x[n] = x_R[n] + jx_I[n]$	$X(e^{jw})$
$x^*[n]$	$X^*(e^{-jw})$
$x^*[-n]$	$X^*(e^{jw})$
$x_R[n]$	$X_e(e^{jw})$, conjugada simétrica de $X(e^{jw})$
$jx_I[n]$	$X_o(e^{jw})$, conjugada anti-simétrica de $X(e^{jw})$

quando $x[n]$ é uma sequência real

$x_e[n] = \frac{x[n] + x^*[n]}{2}$, parte par de $x[n]$	$X_R(e^{jw})$
$x_o[n] = \frac{x[n] - x^*[n]}{2}$, parte ímpar de $x[n]$	$jX_I(e^{jw})$

Transformada de z de $x[n]$

$x[n]$	$X(z) = \sum_{n=-\infty}^{+\infty} x[n] z^{-n}$
$x[n] = \frac{1}{2\pi j} \oint z^{n-1} dz$	$X(z)$

Transformada discreta de Fourier $x[n]$

$h[n]$	$H(k) = \frac{1}{N} \sum_{n=-\infty}^{+\infty} h[n] e^{-j \frac{2k\pi}{N} n}, 0 \leq k < N$
$\tilde{h}[n] = \frac{1}{N} \sum_{k=-\infty}^{+\infty} H(k) e^{j \frac{2k\pi}{N} n}, 0 \leq n < N$	$H(k)$

Algumas relações matemáticas e trigonométricas úteis

- $\sum_{n=-\infty}^{+\infty} |h[n]| = \dots, h[n_{-1}], h[n_0], h[n_1], \dots \begin{cases} S_n = u_1 \frac{1 - \text{razão}^{n^2 \text{ termos}}}{1 - \text{razão}}, \text{ progressão geométrica} \\ S_n = \frac{u_1 - u_n}{2} \times n^2 \text{ termos}, \text{ progressão aritmética} \end{cases}$
- $\cos w = \frac{e^{jw} + e^{-jw}}{2}$
- $\sin w = \frac{e^{jw} - e^{-jw}}{2j}$
- $H(e^{jw}) = \sum_{n=-\infty}^{+\infty} h[n] e^{-jwn} = \sum_{n=-\infty}^{+\infty} h[n] \cos(wn) + j \sum_{n=-\infty}^{+\infty} h[n] \sin(wn)$
- $H(e^{-jw}) = \sum_{n=-\infty}^{+\infty} h[n] e^{jwn} = \sum_{n=-\infty}^{+\infty} h[n] \cos(wn) - j \sum_{n=-\infty}^{+\infty} h[n] \sin(wn)$
- $H(e^{jw}) = H_R(e^{jw}) + jH_I(e^{jw})$
- $|H(e^{jw})| = \sqrt{H(e^{jw})H^*(e^{jw})}$
- $H(e^{jw}) = \frac{N(e^{jw})}{D(e^{jw})}, \begin{cases} |H(e^{jw})| = \sqrt{\frac{N(e^{jw})N^*(e^{jw})}{D(e^{jw})D^*(e^{jw})}} \\ \arg H(e^{jw}) = \tan^{-1} \frac{H_I(e^{jw})}{H_R(e^{jw})} \end{cases}$

Anexo_C

Avaliação Prática

Mini-Projetos de Contacto
&
Propostas de Projetos Finais (*@home*)

TUTORIAL

CRIAÇÃO DE SEQUÊNCIAS E OPERAÇÕES ALGÉBRICAS DISCRETAS EM MATLAB

Para concretizar o trabalho prático nº1 é necessário numa fase inicial criar uma nova Álgebra que permita **gerar** sequências discretas permitindo **operações** entre elas (adição algébrica, multiplicação/divisão, translação e rotação), **decompô-las** (parte par e ímpar p.ex) ou calcular **medidas objetivas** (energia ou potência).

1. Inicie as seguintes sequências:

- Impulso unitário

$$\delta[n - n_0] = \begin{cases} 1, n = n_0 \\ 0, n \neq n_0 \end{cases}$$

```
function [x,n]=impseq(n0,n1,n2)
%Gera o impulso unitário
%-----
%[x,n]=impseq(n0,n1,n2)

if((n0<n1) || (n0>n2) || (n1>n2))
    error('arguments must satisfy n1<=n0<=n2')
end

n=[n1:n2];
x=[zeros(1,(n0-n1)),1,zeros(1,(n2-n0))];
%[x]=[ (n-n0)==0];
```

- Degrau unitário

$$u[n - n_0] = \begin{cases} 1, n \geq n_0 \\ 0, n < n_0 \end{cases}$$

```
function [x,n]=stepseq(n0,n1,n2)
%Gera o degrau unitário
%-----
%[x,n]=stepseq(n0,n1,n2)

if((n0<n1) || (n0>n2) || (n1>n2))
    error('arguments must satisfy n1<=n0<=n2')
end

n=[n1:n2];
%[x]=[zeros(1,(n0-n1)),ones(1,(n2-n0+1))];
x=[(n-n0)>=0];
```

- Exponencial real

$$x[n] = a^n, \forall n; a \in \mathbb{R}$$

```
function [x,n]=realseq(a,n1,n2)
%Gera uma exponencial real
%-----
%[x,n]=realseq(a,n1,n2)

if(a==NaN)
    error('a must be real')
end

n=[n1:n2];
x=(a).^n;
```

- Exponencial complexa

$$\dagger x[n] = e^{(\sigma + j\omega_0)n}, \forall n$$

- (MatLab) >> n=[0:10]; x = exp((2+3j)*n)

- Sinusoide

$$\dagger x[n] = \cos(\omega_0 n + \theta), \forall n$$

- (MatLab)>> n=[0:10]; x = 3*cos(0.1*pi*n+pi/3)

- Aleatória

- (MatLab)>> x = rand(1,20)

- Periódica

$$\dagger$$
 (MatLab)


```
>>x=[1,2,3]
>>Periodo=3
>>x_periodico=x*ones(1,Periodo)
>>x_periodico=x_periodico(:)
>>x_periodico=x_periodico'
```

2. Operações com seqüências

Adição/Subtração

$$\nabla \quad \{x_1[n]\} \pm \{x_2[n]\} = \{x_1[n] \pm x_2[n]\}$$

```
function [y,n] = sigadd(x1,n1,x2,n2)
% implements y(n) = x1(n)+x2(n)
% -----
% [y,n] = sigadd(x1,n1,x2,n2)
% y = sum sequence over n, which includes n1 and n2
% x1 = first sequence over n1
% x2 = second sequence over n2 (n2 can be different from n1)
%
n = min(min(n1),min(n2)):max(max(n1),max(n2)); % duration of y(n)
y1 = zeros(1,length(n)); y2 = y1; % initialization
y1(find((n>=min(n1))&(n<=max(n1))=1))=x1; % x1 with duration of y
y2(find((n>=min(n2))&(n<=max(n2))=1))=x2; % x2 with duration of y
y = y1+y2; % sequence addition
%y = y1-y2; % sequence difference
```

Multiplicação/Divisão

$$\nabla \quad \{x_1[n]\} ./ \{x_2[n]\} = \{x_1[n] ./ x_2[n]\}$$

```
function [y,n] = sigmult(x1,n1,x2,n2)
% implements y(n) = x1(n)*x2(n)
% -----
% [y,n] = sigmult(x1,n1,x2,n2)
% y = * sequence over n, which includes n1 and n2
% x1 = first sequence over n1
% x2 = second sequence over n2 (n2 can be different from n1)
%
n = min(min(n1),min(n2)):max(max(n1),max(n2)); % duration of y(n)
y1 = zeros(1,length(n)); y2 = y1; % initialization
y1(find((n>=min(n1))&(n<=max(n1))=1))=x1; % x1 with duration of y
y2(find((n>=min(n2))&(n<=max(n2))=1))=x2; % x2 with duration of y
y = y1.*y2; % sequence addition
%y = y1./y2; % sequence division
```

▼ Translação

$$\Downarrow \quad y[n] = \{x[n - k]\}$$

```
function [y,n] = sigshift(x,m,n0)
% implements y(n) = x(n-n0)
% -----
% [y,n] = sigshift(x,m,n0)
%
n = m+n0; y = x;
```

▼ Rotação

$$\Downarrow \quad y[n] = \{x[-n]\}$$

```
function [y,n] = sigfold(x,n)
% implements y(n) = x(-n)
% -----
% [y,n] = sigfold(x,n)
%
y = fliplr(x); n = -fliplr(n);
```

▼ Decomposição em parte par e ímpar

$$\Downarrow \quad x[n] = x_e[n] + x_o[n] = \left\{\frac{1}{2}[x[n] + x[-n]]\right\} + \left\{\frac{1}{2}[x[n] - x[-n]]\right\}$$

```
function [xe, xo, m] = evenodd(x,n)
% Real signal decomposition into even and odd parts
% -----
% [xe, xo, m] = evenodd(x,n)
%
if any(imag(x) ~= 0)
    error('x is not a real sequence')
end
m = -fliplr(n);
m1 = min([m,n]); m2 = max([m,n]); m = m1:m2;
nm = n(1)-m(1); n1 = 1:length(n);
x1 = zeros(1,length(m));
x1(n1+nm) = x; x = x1;
xe = 0.5*(x + fliplr(x));
xo = 0.5*(x - fliplr(x));
```

• Somatório das amostras

$$\Updownarrow \quad \sum_{n_1}^{n_2} x[n] = x[n_1] + \dots + x[n_2]$$

▪ (MatLab) >> sum(x(n1:n2))

• Produto das amostras

$$\Updownarrow \quad \prod_{n_1}^{n_2} x[n] = x[n_1] \times \dots \times x[n_2]$$

▪ (MatLab) >> prod(x(n1:n2))

• Energia de um sinal (sequência)

$$\Updownarrow \quad \varepsilon_x = \sum_{-\infty}^{\infty} x[n]x^*[n] = \sum_{-\infty}^{\infty} |x[n]|^2$$

▪ (MatLab) >> Ex=sum(x.*conj(x));

▪ (MatLab) >> Ex=sum(abs(x).^2);

- Potência de um sinal (sequência)

$$\dagger \quad P_x = \frac{1}{N} \sum_0^N |x[n]|^2$$

Exercício 1:

Gere e apresente num gráfico as seguintes sequências para os intervalos indicados:

a) $x[n] = 2\delta[n+2] - \delta[n-4], -5 < n < 5.$

```
n=[-5:5];
x=2*impseq(-2,-5,5)-impseq(4,-5,5);
subplot(2,2,1);stem(n,x);title('Sequência da alinea a)')
xlabel('n'),ylabel('x[n]')
```

b) $x[n] = n[u[n] - u[n-10]] + 10e^{-0.3(n-10)}[u[n-10] - u[n-20]], 0 < n < 20.$

```
n=[0:20];
x1=n.*(stepseq(0,0,20)-stepseq(10,0,20));
x2=10*exp(-0.3*(n-10)).*(stepseq(10,0,20)-stepseq(20,0,20));
x=x1+x2;
subplot(2,2,2);stem(n,x);title('Sequência da alinea b)')
xlabel('n'),ylabel('x[n]')
```

c) $x[n] = \cos(0.04\pi n) + 0.2w[n], 0 < n < 50, w[n]$ é uma sequência gaussiana aleatória

```
n=[0:50];
x=cos(0.04*pi*n)+0.2*randn(size(n));
subplot(2,2,3);stem(n,x);title('Sequência da alinea c)')
xlabel('n'),ylabel('x[n]')
```

d) $\tilde{x}[n] = \{ \dots 5, 4, 3, 2, 1, \mathbf{5}_{(n=0)}, 4, 3, 2, 1, 5, 4, 3, 2, 1, \dots \}, -10 < n < 9.$

```
n=[-10:9];
x=[5:-1:1];
xperiodico=x*ones(1,4);
xperiodico=(xperiodico(:));
subplot(2,2,4);stem(n,xperiodico);title('Sequência da alinea d)')
xlabel('n'),ylabel('x[n]')
```


Exercício 2:

Admita a sequência

$$x[n] = \{1, 2, 3, 4, 5, 6, 7, 6, 5, 4, 3, 2, 1\}, -2 < n < 10.$$

Gere e apresente num gráfico as seguintes sequências:

a) $x_1[n] = 2x[n-5] - 3x[n+4].$

```
n=[-2:10];
x=[1:7,6:-1:1];

[x11,n11]=sigshift(x,n,5);
[x12,n12]=sigshift(x,n,-4);
[x1,n1]=sigadd(2*x11,n11,-3*x12,n12);

subplot(2,1,1);stem(n1,x1);title('Sequência da alínea a)')
xlabel('n');ylabel('x1[n]')
```

b) $x_2[n] = x[3-n] + x[n]x[n-2].$

```
n=[-2:10];
x=[1:7,6:-1:1];

[x21,n21]=sigfold(x,n);[x21,n21]=sigshift(x21,n21,3);
[x22,n22]=sigshift(x,n,2);
[x22,n22]=sigmult(x,n,x22,n22);
[x2,n2]=sigadd(x21,n21,x22,n22);

subplot(2,1,2);stem(n2,x2);title('Sequência da alínea b)')
xlabel('n');ylabel('x2[n]')
```

Exercício 3:

Crie o sinal complexo

$$x[n] = e^{(-0.1+j0.3)n}, -10 < n < 10$$

e represente-o graficamente: módulo (magnitude), fase, parte real e parte imaginária:

```
n = [-10:1:10]; alpha = -0.1+0.3j;
x = exp(alpha*n);
subplot(2,2,1); stem(n,real(x));title('real part');xlabel('n')
subplot(2,2,2); stem(n,imag(x));title('imaginary part');xlabel('n')
subplot(2,2,3); stem(n,abs(x));title('magnitude part');xlabel('n')
subplot(2,2,4); stem(n,(180/pi)*angle(x));title('phase part');xlabel('n')
```

Exercício 4:

Considere

$$x[n] = u[n] - u[n - 10],$$

Decomponha-o nas suas componentes par e ímpar representando-as graficamente.

```
n = [0:10];
x = stepseq(0,0,10)-stepseq(10,0,10);
[xe,xo,m] = evenodd(x,n);
subplot(1,1,1)
subplot(2,2,1); stem(n,x); title('Rectangular pulse')
xlabel('n'); ylabel('x(n)'); axis([-10,10,0,1.2])
subplot(2,2,2); stem(m,xe); title('Even Part')
xlabel('n'); ylabel('xe(n)'); axis([-10,10,0,1.2])
subplot(2,2,4); stem(m,xo); title('Odd Part')
xlabel('n'); ylabel('xe(n)'); axis([-10,10,-0.6,0.6])
```

TUTORIAL

A DIGITALIZAÇÃO O TEOREMA DA AMOSTRAGEM E A RECONSTRUÇÃO PERFEITA

Pretende-se rever os principais conceitos associados à operação digitalização através de análise de Sons no formato digital nomeadamente através do cálculo da frequência natural da voz individual de cada aluno. A concretização de experiências devem levar ao entendimento claro de conceitos tais como: resolução ($f_s, 2^n$) Hz/bits; frequência de amostragem; fator de sobre-amostragem (*over-sampling*); níveis de quantização; redundância estatística e perceptual; algoritmos de compressão; largura de banda; requisitos da voz e do áudio; mascaramento no tempo (pré/pós) e na frequência (ou simultâneo); codificadores perceptuais; MP3 – *Constant Bit Rate* (CBR) e *Variable Bit Rate* (VBR); Bit-Stream de áudio.

Propriedades Objectivas do Som

- 📖 Com o gravador “Edirol R-09 24-bit-Wave/MP3” digitalize a voz individual com a resolução máxima do dispositivo (96000Hz; 24bits) no formato sem perdas “.wav”.
 - ☐ Registe numa tabela a **frequência natural** da voz e o **tamanho de cada ficheiro** para os seguintes pares (96000 Hz; 24 bits) , (88200 Hz; 24 bits); (44100 Hz; 24 bits) (22050 Hz; 24 bits) (11025 Hz; 24 bits) (6000 Hz; 24 bits) (3000 Hz; 24 bits); (1000 Hz; 24 bits). Comente os resultados. Deve justificar-se a razão pela qual a frequência natural e o tamanho variam (ou não) de cada conteúdo sonoro.
 - ☐ Registe numa tabela a **frequência natural** da voz e o **tamanho de cada ficheiro** para os seguintes pares (88200 Hz; 24 bits); (88200 Hz; 16 bits); (88200 Hz; 8 bits). Comente os resultados. Deve justificar-se a razão pela qual a frequência natural e o tamanho variam (ou não) de cada conteúdo sonoro.
- 📖 Sabendo que a característica única do sinal de voz humana relacionada com a variação dos impulsos glóticos permite medir, por exemplo, a sua frequência natural, repita a questão anterior mas agora utilizando as “**Demos do Edirol**”.

Avaliação Subjectiva do Som

- 📖 Identifique onde se localizam os **pontos críticos** ($f_s, 2^n$) Hz/bits associados à percepção da degradação da qualidade do Som para os exemplos anteriores. Deve justificar a razão pela qual esses **pontos variam em função da largura de banda** de cada conteúdo sonoro.

Aplicações e curiosidades

- 📖 A digitalização potenciou entre muitas outras, a área de desenvolvimento o uso de codificadores que visam a representação eficiente dos dados nomeadamente conhecidos por algoritmos de compressão com o principal objetivo de reduzir a redundância estatísticas e/ou perceptual. Investigue quais os mecanismos naturais estão associados aos fenómenos de mascaramento no tempo (pré/pós) ou na frequência (ou simultâneo) [1] e apresente um conteúdo sonoro que seja um exemplo **da vantagem da compressão** em *bits* MP3/mp3PRO® Encoder evidenciando de uma forma clara **as diferenças** entre ConstantBitRate (CBR) e VariableBitRate (VBR) bem como o interesse na utilização de vários valores para o **Bitrate**.
- 📖 A inserção de ruído como técnica para reduzir ruído intitulada Dithering (Audio bit Depth- [2]) consiste na geração de ruído aleatório com baixa intensidade mascarando ao ouvido humano, as transições de alta frequência originadas pela diminuição da resolução do número de bits na representação de um sinal digital: produza exemplos que concretizem este conceito.
- 📖 Na figura abaixo apresenta-se uma imagem dum clássico de referência na área dos leitores de CD que faz uso da técnica de Over-Sampling [3] (sobre-amostragem). Sabendo que over-sampling significa digitalizar acima do critério mínimo ou seja, com uma frequência de amostragem maior da crítica/Nyquist ($2f_{max}$) e assumindo a disponibilidade de HW ao nível das ADC do sistema, justifique o interesse da utilização desta estratégia na reprodução do sinal áudio com base no teorema da Amostragem estudado.



Referências:

- [1] https://en.wikipedia.org/wiki/Auditory_masking, julbo 2020.
- [2] https://en.wikipedia.org/wiki/Audio_bit_depth, julbo 2020.
- [3] <https://medium.com/@skwareman/marantz-cd-94-cd-player-part-6-e44c779421c7>, julbo 2020.



TUTORIAL

MANIPULAÇÃO DE CONTEÚDOS MULTIMÉDIA

Pretende-se com este tutorial realizado em colaboração com a Escola Secundária de Arouca, utilizar um conjunto de ferramentas *open source* e *on-line* e free de manipulação de conteúdos Multimédia com o objetivo de criar um documento audiovisual, um vídeo legendado, que deverá estar disponível ao público.

<https://www.y2mate.com/pt/> - **Y2mate YouTube Downloader** is a leading downloader that focuses on free HD video services. You can find massive amount of video & music, and also download content from various websites, including but not limited to Youtube, Facebook, Instagram. Easy, fast, and small, Y2mate YouTube Downloader is your best choice for video download!

<https://online-video-cutter.com/pt/> - **Online Video Cutter** - This web app comes in handy when you need to cut a small video file. It does not require installation, and it works in your browser. *Online video cutter; Crop video; Rotate video; Convert video; Video Editing online.*

<https://phonicmind.com/> - **Vocal Remover and more!** Separate vocals, drums, bass and other instruments out of your songs with our State of the Art AI.

<https://speech-to-text-demo.ng.bluemix.net/> - **Speech to Text** - The IBM Watson Speech to Text service uses speech recognition capabilities to convert Arabic, English, Spanish, French, Brazilian Portuguese, Japanese, Korean, German, and Mandarin speech into text.

<https://www.keapwing.com/editor/subtitles> - **Where Contents Creation Happens** – Subtitle Add captions and subtitles to video completely in the browser.

<https://www.youtube.com/watch?v=Ua3AIW6RVv64#action=share> - **How To Upload Videos To YouTube**

<https://online-audio-converter.com/pt/> **Audio Convert on-Line**

On-line Exercício

- a. Ouça <https://www.youtube.com/watch?v=zgaCIXAt3aY> - Raul Midón - State of Mind [Live On Late Show With Dave Lett;
- i. Faça o download do ficheiro <https://www.youtube.com/watch?v=BTv24sSNXHC>, com uma ferramenta *on-line*, e.g , <https://www.y2mate.com/pt/>;
- ii. Corte o vídeo em <https://online-video-cutter.com/pt/> no intervalo [00:11.0;00:47.0]seg, e exporte para o seu computador;
- iii. Separe a voz do vídeo em <https://phonicmind.com/> e descarregue a voz em **.mp3**;

iv. Reconheça o conteúdo da voz com a ferramenta <https://speech-to-text-demo.ng.bluemix.net/> e verifique se texto obtido no resultado é similar a um dos exemplos:

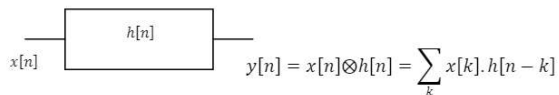
1. **Speaker 0:** *When I when I moved to New York* / **Speaker 1:** *Your.*
2. **Speaker 1:** *A few months later* / **Speaker 1:** *A few months later.*
3. **Speaker 0:** *The winter came down* / **Speaker 3:** *The winter came down.*
4. **Speaker 1:** *And it was cold* / **Speaker 2:** *You know it was cold.*
5. **Speaker 1:** *And there is* / **Speaker 3:** *And the reason why I did it just gotten higher and I didn't know how I was going to come up with it.*
6. **Speaker 0:** *Like it it just got hired and I didn't know how I was going to come up with it* / **Speaker 1:** *Yes.*
7. **Speaker 1:** *Press* / **Speaker 3:** *So I did something practical about it.*
8. **Speaker 1:** *So I did something practical about it* / **Speaker 3:** *I got a job.*
9. **Speaker 1:** *I got a job no* / **Speaker 2:** *No.*
10. **Speaker 0:** *I wrote a song* / **Speaker 3:** *I wrote a song.*
11. **Speaker 0:** *And this is called state of mind* / **Speaker 3:** *It's called state of mind.*

- b. Apresente a tradução que considere mais correta deste trecho da música de Raul Midón em português e coloque as legendas no filme utilizando a aplicação em <https://www.kapwing.com/editor/subtitles>;
- c. Qual é, para si, a mensagem desta música? (<https://www.webbostingsecretrevealed.net/pt/blog/web-copy-writing/the-who-what-where-when-and-why-of-excellent-blog-writing/>) Responda ao questionário incluso. Raul Midón (QUEM) criou esta música (QUÊ) QUANDO? ONDE? PORQUÊ? COMO? PORQUANTO?
- d. Envie o ficheiro final com o vídeo legendado para o seu canal de youtube <https://www.youtube.com/watch?v=Ua3-4W6RVv64#action=share> e envie o link para o email: salblues@utad.pt.

TRABALHO PRÁTICO N.1

A REALIZAÇÃO DA CONVOLUÇÃO E CORRELAÇÃO DISCRETAS

Um sistema discreto linear e invariante ao tempo (**LTI**), é completamente caracterizado no domínio do tempo se conhecermos a sua resposta impulsional, $h[n]$. Desta forma é possível obter a sua saída realizando a operação **convolução linear** $\otimes$ conforme se apresenta no diagrama de blocos.



1. **Projete um algoritmo** em MatLab que implemente a convolução linear **pela definição** e que permita ao mesmo tempo a sua visualização/interpretação gráfica.
2. Demonstre a robustez através das seguintes **propriedades** da convolução linear:

- ✓ **Comutativa** $\otimes$ $x_1[n] \otimes x_2[n] = x_2[n] \otimes x_1[n]$
- ✓ **Associativa** $\otimes$ $\{x_1[n] \otimes x_2[n]\} \otimes x_3[n] = x_1[n] \otimes \{x_2[n] \otimes x_3[n]\}$
- ✓ **Distributiva** $\otimes$ $x_1[n] \otimes \{x_2[n] + x_3[n]\} = x_1[n] \otimes x_2[n] + x_1[n] \otimes x_3[n]$
- ✓ **Identidade** $\otimes$ $x[n] \otimes \delta[n - n_0] = x[n - n_0]$

3. Apresente uma **realização** da convolução discreta.

A **correlação** é uma operação utilizada em muitas aplicações de processamento discreto de sinais. O seu resultado dá-nos **uma medida da semelhança** entre duas sequências. Assim, dados dois sinais discretos de energia finita, $x[n]$ e $y[n]$, define-se correlação r_{xy} como

$$r_{xy}[l] = \sum_n x[n].y[n-l]$$

No caso particular de $y[n] = x[n]$, a **auto-correlação** permite inferir a proximidade entre diferentes versões desfasadas da mesma sequência.

1. **Projete um algoritmo** em MatLab que implemente a **correlação pela definição** e que permita ao mesmo tempo uma interpretação gráfica da operação utilizando **o algoritmo da convolução linear** já desenvolvido.
2. É possível estabelecer relações entre a correlação e a convolução: demonstre com exemplos práticos as relações seguintes:

$$r_{xy}[l] = x[l] \otimes y[-l]$$

$$r_{xx}[l] = x[l] \otimes x[-l]$$

3. Apresente uma **realização** da correlação discreta.

TRABALHO PRÁTICO N.2

REALIZAÇÃO DA TRANSFORMADA RÁPIDA DE FOURIER COM A FUNÇÃO “*fft.m*” DO MATLAB

Pretende-se com este trabalho rever a transformada discreta de Fourier (**DFT**) quando aproximada pelo algoritmo da transformada rápida de Fourier (**fft**).

A concretização de experiências de análise espectral devem levar ao entendimento claro de conceitos tais como: frequência de amostragem; frequência de Nyquist; *zero padding*; *bit reverse*; n° de pontos da DFT potência de 2; simetrias da DFT; teorema da conservação da energia (Parseval); normalização da função “*fft.m*” do MatLab ou a interpretação de gráficos no domínio da frequência.

Procedimento

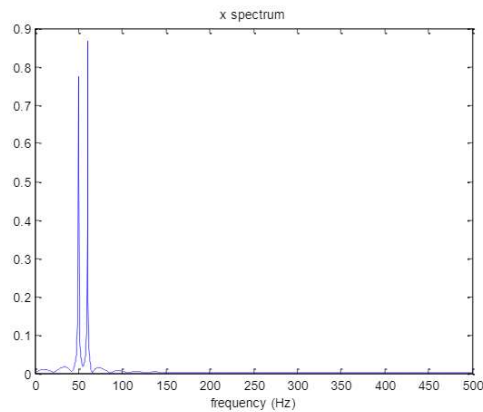
1. Inicie um vetor **x** com uma sinusóide de 200Hz, amostrada a 1 kHz e com 1 segundo de duração.
2. Calcule a transformada de Fourier do sinal **x** para o vetor **FFTX**. Para otimizar o desempenho utilize a função do MatLab ‘*fft.m*’ com um número de pontos (**NFFT**) igual à potência de dois seguinte ao tamanho do vetor **x**, acrescentado a **x** o número de zeros necessários (técnica de *zero padding*).
3. Se **NFFT** for par (como é o caso visto ser potência ou seja $NFFT=2^n$), a magnitude (valor absoluto ou módulo) de **FFTX** é simétrica sendo apenas necessário utilizar os pontos entre 0 e $(1+NFFT/2)$. **FFTX**(1) é a componente DC de **x** e **FFTX**($1+NFFT/2$) corresponde à frequência de Nyquist. O que acontece se **NFFT** for ímpar?
4. Dada a existência de simetria da DFT quando de sequências apenas reais, é possível apresentar apenas metade do espectro. Desta forma, elimine de **FFTX** os valores redundantes.
5. Calcule a magnitude do vetor **FFTX** (representado por **abs_FFTX**). Note que a cada amostra deve ser multiplicado por 2, exceto para o termo DC e para a frequência de Nyquist (f_n) caso esteja presente.
6. A função ‘*fft.m*’ do MATLAB não normaliza o resultado em função do número de pontos do sinal. É assim necessário efetuar a normalização dividindo o resultado já obtido pelo número de pontos utilizado no seu cálculo.
7. Antes de fazer o gráfico da magnitude de **FFTX**, é necessário gerar um vetor **f** com os índices do eixo da frequência. Os valores estão igualmente espaçados entre 0 e metade da frequência de amostragem ou seja (f_n).
8. Se efetuou corretamente todos os passos, está agora em condições de apresentar o gráfico da magnitude da transformada de Fourier do sinal **x** em função da frequência. Para isso use o comando **stem(f,abs_FFTX)**.
9. Escreva agora o código de uma função que aceite a sequência e a frequência de amostragem, **plot_fft_half_spectrum(x,f_s)**, que permita calcular o gráfico de metade da magnitude do espectro de um sinal em função da frequência.

Questões

Realize a seguinte experiência:

```
fs=1000; % frequência de amostragem
t=0:1/fs:2-1/fs; % vetor tempo com 2seg. (2000 amostras)
x1= sin (2* pi *t*50); % sinal com uma frequência de 50 Hz
x2= sin (2* pi *t*60.4); % sinal com uma frequência de 60.4 Hz
x= x1+ x2; % sinal com as componentes de frequência
plot_fft_half_spectrum(x,fs) % gráfico da magnitude da FFTX vs Frequência
```

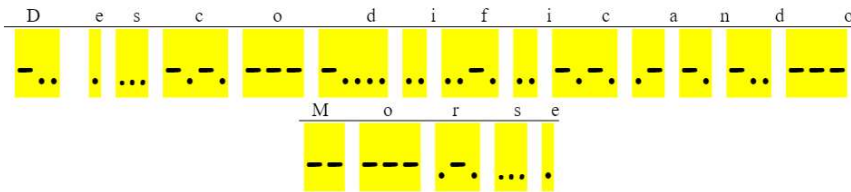
- a. Analisando o gráfico obtido consegue explicar porque é que as duas componentes de frequência **têm magnitudes diferentes**, apesar de serem geradas com sinais com a mesma energia?



- b. Será possível encontrar **outros dois sinais x_1 e x_2** de forma a que as componentes de frequência tenham a mesma energia? Justifique.

```
fs=1000; % frequência de amostragem
t=0:1/fs:2-1/fs; % vetor tempo com 2seg. (2000 amostras)
x1= sin (2* pi *t*???); % sinal com uma frequência de ??? Hz
x2= sin (2* pi *t*???); % sinal com uma frequência de ??? Hz
x= x1+ x2; % sinal com as componentes de frequência
plot_fft_half_spectrum(x,fs) % gráfico da magnitude da FFTX vs Frequência
```

- c. Escreva agora o código de uma função que apresente **o gráfico completo**, $[0; f_s]$ Hz, da magnitude do espectro de um sinal em função da frequência. (`plot_fft_full_spectrum(x,fs)`)



Objetivo principal:

- Deve ser implementado um algoritmo que **descodifique** mensagens escritas em código Morse através da interpretação de sons guardados num **ficheiro “.wav”**.

Algumas observações

- ❖ O código Morse é um sistema de representação de letras, números e sinais de pontuação através de um sinal codificado enviado intermitentemente. Foi desenvolvido por Samuel Morse e Alfred Vail em 1835, criadores do telégrafo elétrico (importante meio de comunicação a distância), dispositivo que utiliza correntes eléctricas para controlar electroímans que funcionam para emissão ou recepção de sinais.
(http://pt.wikipedia.org/wiki/C%C3%B3digo_Morse);
- ❖ O método ***Koch method CW trainer*** permite a aprendizagem da interpretação do código Morse de uma forma simples e deve ser estudado.
- ❖ Pretende-se **encontrar exemplos** que demonstrem o interesse na utilização do código Morse em situações de **transmissão em ambiente adverso**!.

© 1989 A. G. Reinhold, 14 Fresh Pond Place, Cambridge, MA 02138
K2PNK "May be freely distributed with attribution."

A SINALIZAÇÃO TELEFÓNICA E A DESCODIFICAÇÃO DE DTMF

A sinalização telefónica está presente desde o início das comunicações telefónicas – *conjunto de dígitos que um utilizador marca para obter uma ligação telefónica*. Na época das estações telefónicas manuais, a informação necessária para o estabelecimento de uma chamada era dita verbalmente à telefonista pelo chamador depois deste levantar o auscultador – Com esta informação ‘queria ligar ao senhor X da localidade Y’ a telefonista seleccionava um circuito livre para a estação destino, ou para uma estação intermédia de trânsito com acesso à estação de destino – Através do circuito seleccionado falava com a telefonista da estação seguinte passando-lhe a informação necessária para esta poder dar continuidade ao estabelecimento da chamada – Em cada estação telefónica manual a operadora fazia a ligação do circuito de entrada com o circuito de saída escolhido, através de um ligador apropriado com o nome de “cordão” – A operadora da estação de destino fazia tocar a campainha do chamado e, se este atendesse ‘punha a linha a direito’ completando o circuito entre chamador e chamado [http://bibliografia.bnportugal.pt/bnp/bnp.exe/registro?mfu=104377, janeiro 2017]



O processo de sinalização na rede digital que hoje utilizamos é em tudo idêntico ao da rede analógica conhecida por Public Switched Telephone Network (PSTN) mas realizado através de mensagens entre estações e o equipamento do utilizador. No caso do Sistema de Sinalização n.º 7 (SS7) a sinalização de acesso pode, entre outras, ser feita por combinação de duas tonalidades que representam dígitos e sinais auxiliares: **Funcionamento bitonal multifrequências (Dual Tone Multi-Frequency - DTMF)** [http://www.anacom.pt/render.jsp?contentId=200952].

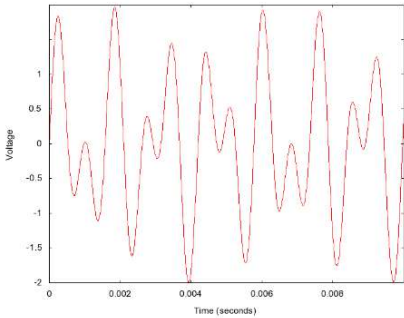
Objetivos

Através da identificação dos conteúdos das mensagens que transportam os números de telefone aquando um processo de sinalização duma rede digital, pretende-se decodificar os ficheiros ‘*.wav’ com as seguintes caraterísticas:

- a. Os ficheiros estão em várias resoluções do tipo (f_s Hz, *bits*, **Mono**);
- b. No mesmo ficheiro podem estar armazenados sequencialmente **vários tons de durações distintas**.
- c. Existem alguns exemplos que **simulam situações reais com ruído White/Brown/Pink**.

Cada DTMF, de acordo com a tabela abaixo, tem **duas frequências**: uma alta e outra baixa, normalmente assim designadas [http://en.wikipedia.org/wiki/Dual-tone_multi-frequency].

DTMF keypad frequencies (with sound clips)				
	1209 Hz	1336 Hz	1477 Hz	1633 Hz
697 Hz	1	2	3	A
770 Hz	4	5	6	B
852 Hz	7	8	9	C
941 Hz	*	0	#	D



OUVER IMAGENS METEOROLÓGICAS – WEFAX | RADIOFAX

Radiofax, também conhecido como Meteorológico e fax HF (devido a seu uso comum no rádio de ondas curtas), é um modo analógico para transmitir monocrômicas imagens. Foi o antecessor de slow-scan TV (SSTV). Antes do advento do fax comercial, ele era conhecido, mais tradicionalmente, pelo termo *radiofacsimile*. A capa do regulares NOAA publicação em frequências e horários afirma "Horários Worldwide Marine RADIOFACSIMILE de Transmissão".

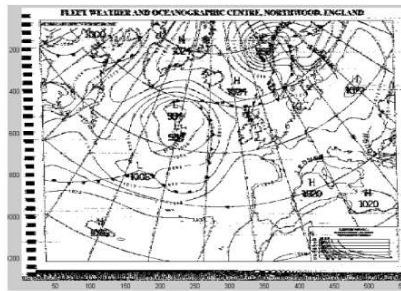
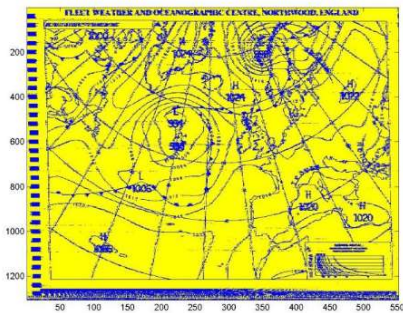
As máquinas de fax foram usados em 1950 para transmitir cartas meteorológicas em todo os Estados Unidos através de terras linhas primeiro e depois internacionalmente via rádio HF. Transmissão de rádio de mapas meteorológicos fornece uma enorme quantidade de dados hoje. Radiofax está disponível via FTP downloads de sites na Internet , como os promovidos pela Administração Oceânica e Atmosférica Nacional (NOAA). Transmissões Radiofax também são transmitidas pela NOAA de vários sites no país no programações diárias regulares. Rádio Meteorológico transmissões são particularmente úteis para o transporte, onde existem instalações limitadas de acesso à Internet na marinha e de aviação de forma a acederem informação meteorológica atual bem como previsões que complementam o planeamento das missões.



Objetivos

Através da leitura de um ficheiro *.wav*, que contém informação de uma imagem de satélite pretende-se analisar quais são as frequências que correspondem às cores branca/preta. O resultado final será a imagem que se pode associar ao som do tipo abaixo apresentado.:

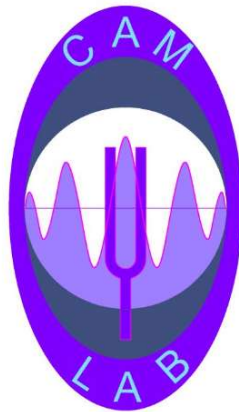
- O ficheiro *.wav* tem a resolução, $f_s=11\text{kHz}$, 8 *bits*, **Mono**;
- Cada linha da imagem da imagem corresponde a meio segundo;
- Por análise das imagens abaixo dê um exemplo duma potencialidade da utilização da pseudo-cor.

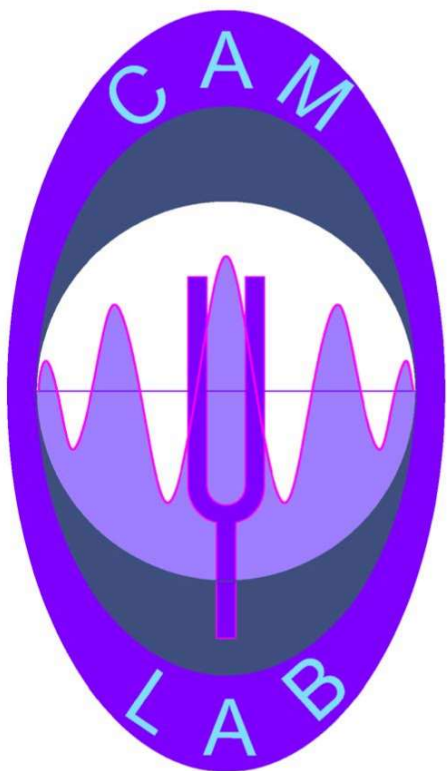


Anexo_D

Tutoriais em PDS

Calibration Audio MIDI Lab – CAM Lab
CAM_Lab@utad.pt





Calibration Audio MIDI Lab – CAM Lab

UTAD

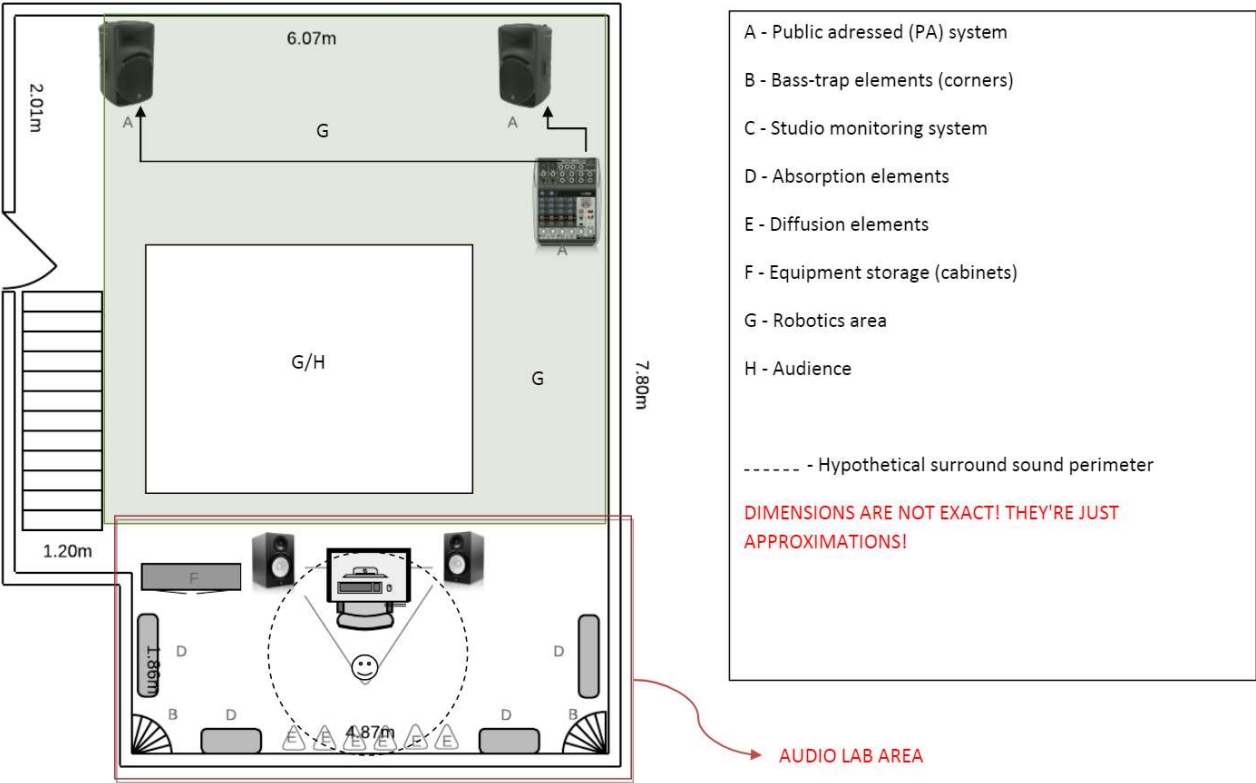
Salviano Filipe Silva Pinto Soares

Departamento de Engenharia

Escola de Ciências e Tecnologia

Edifício de Eng_I – Sala:-2.06

utad UNIVERSIDADE
DE TRÁS-OS-MONTES
E ALTO DOURO



Electro-acoustic considerations:

- **PA system (A) would consist of two active full-range speakers plus one small passive mixing desk.** Aimed at small conferences, talks, sound demos, etc. This equipment is (allegedly) already available;
- **Studio monitoring system (C) would consist of two full-range studio monitors** (already existing at UTAD);
- **Bass-trap elements would be aimed at absorption in corners** (these are some of the most problematic areas);
- **Absorption and diffusion elements (D, E) positioned as shown should be an improvement in acoustic treatment,** especially at the listener's position;
- **Some of this diffusion/absorption could actually be done with everyday-objects** (such as books, shelves, etc.), namely materials with more interesting absorption coefficients (wood, paper, cardboard, etc.). Better diffusion would be a matter of positioning relatively reflective materials (such as shelves, cabinets, etc.) in a more "random" distribution;
- **Some of the objects used in the previous point could be cabinets** for equipment storage (F);
- **The playback system could be positioned so that it allowed for further experiments such as 5.1 Surround** playback (see diagram above);
- **Some sort of false ceiling could be placed above the audio-lab area. Maybe plasterboard ("Pladur") could be an efficient solution.** It could also create an "air box" area in between the false ceiling and the original one, which would be acoustically beneficial;

Audio rig considerations



A - A functioning computer would be needed. Windows or MAC OS with the following basic specs:

- CPU Intel Core i5/i7 (quadcore or octacore)
- Minimum 8 GB RAM
- 1TB HDD (128 GB SSD ideally)
- Firewire (IEEE 1394) PCI-express port (or else a USB audio interface would be needed)
- A Digital Audio Workstation software educational license (much cheaper!) could be bought.

B - An audio interface (preferably USB) with a minimum of 2 channels (more, if possible) would allow ADC/DAC conversion, pre-amplification of microphone signals and audio playback through headphones and/or speakers.

- Affordable interfaces already accept high sample rates (up to 192 KHz) and industry-standard bit rates (16, 24bit).
- Preamps are very acceptable for this price-range. Obviously there's no need for state-of-the-art quality for this purpose.
- An interface of this type would still be very portable for other purposes.

C - An affordable MIDI controller keyboard would allow MIDI recording and control-changes in real-time.

- Most MIDI controllers now run via USB. They are also USB powered.

D - Most microphones would work perfectly using this kind of interface.

- Having a pair of condenser microphones plus another couple dynamic microphones would be enough to start with.

E - (Almost) Full-range near-field active monitors are sufficient. Also no need for very large drivers (woofers). 5" or 7" is enough for the room.

F - Affordable closed headphones are sufficient.

CAM_LAB UTAD (Infrastructure reinforcement proposal) - Salviano F.S. Pinto Soares July, 2020

Available equipment inventory (in progress)

Item	Description	Quantity	Obs
StudioMaster Session Mix 16-2	Analogue mixing desk	1	To be tested
Alto AMX 140 FX	Mixing desk	1	Could be used in PA system
Edirol FA-66	Firewire audio interface	1	Mini-IEEE 1394 connector
Edirol M5A	Studio monitors	2	To be tested
Edirol MA-15D	Small studio monitors (unbalanced inputs)	2	Fully operational
Alto EQU 231 VU MkII	Graphic EQ (19" rack unit)	1	Fully operational
Alto RTA RSA 27	Real-time analyser (19" rack unit)	1	Fully operational
Alto DI box	DI Box	1	To be tested
Vinyl player	Vinyl player	1	RCA output?
	Dynamic microphone	1	
	Measurement microphone		Flat response
Creative Soundblaster Omni 5.1	5.1 hi-fi system	1	5 satellites + 1 subwoofer + USB card
Misc cables (RCA, XLR, TRS, TS)	-	-	-
White Board		1	
Edirol	Portable Recorder		
Universal Audio Apollo Twin USB	2-channel audio interface	1	
Behringer Deepmind 6	6-voice analogue synthesizer	1	
Nektar impact LX61x	61-key MIDI controller keyboard	1	
Pioneer Hi-fi system		1	
Shure SM58	Dynamic microphone	1	
Electro-Voice PL35	Dynamic instrument mics	2	

CAM_LAB UTAD (Infrastructure reinforcement proposal) - Salviano F.S. Pinto Soares July, 2020

Item	Description	Quantity	Obs
Sony Vaio	Laptop PC	1	
Toshiba Atom N28	Leptop PC	1	
SES224Q	External recorder	1	
Samsung TFT LCD	LCD screen	1	
HP Pavillion	Laptop PC	1	
Toshiba Atom N280	Laptop PC	1	
Apple Macbook Air	Laptop PC	1	
Western Digital External HDD		1	
Rack Tour Pro 16L	19 rack unit mobile hardcase	1	
Cromatic Tuner		1	
PC Picoscope	Oscilloscope	1	
Edirol MA7D	Studio monitors	2	
Neutrik EG	Reference microphone	1	
Sony VPL CX150	Projector	1	
Edirol Wave	MP3 recorder	1	
JTS wireless system		1	

 CAM_LAB UTAD (Infrastructure reinforcement proposal) - Salviano F.S. Pinto Soares July, 2020

Equipment worth acquiring

Item	Description	Units	Price (total, Thomann)	Obs.
Miditech Minicontrol-32	MIDI controller keyboard	1	55€	Simple MIDI controller (via USB)
Superlux HD660	Studio headphones (closed)	1	30€	Good price-performance
Behringer C-4 stereo set	Pair of studio condenser microphones	1 (contains 2 mics)	40€	For basic stereo recording
Millenium	Mics stands	2	20€	-
Adam Hall SDB SB190	Stereo bar	1	6,90€	-
Behringer UMC404HD	USB Audio interface (4 chan.)	1	104€	Firewire is outdated. USB is the present and future.
The T.akustic SA-P80 6pcs set	Absorption panels	1 (contains 6 pieces!)	59€	Standard 80mm thick polyurethane panels
XLR cables				

Available software:

Name	Description	Obs
Melda productions Free Bundle	Misc. audio processing plugins	
Izotope Elements 8	Mastering suite	Basic features
Avid Pro Tools 11	DAW	
Flux Free plugins	Audio analysis	
HOFA misc. plugins		
HOFA CD-burn.DDP.Master	Mastering/Authoring software	
Slate Digital Virtual MixRack	Mixing plugins	Compressors, EQ...
AIR Hybrid	Digital synthesis software	Several waveforms, saturation, EQ...
AIR xPand!	Virtual Instruments	
AIR creative collection	Various signal processing plugins	

In-situ demonstrations:

- 5.1 Surround sound editing/mixing



- Editing, mixing and mastering techniques for stereo surround 5.1
- Sound design techniques in multichannel audio
- Sound recording applied to 5.1 contexts
- Optimization of the Adobe Audition digital audio workstation for surround applications;

- Chladni's plate (cymatics) demonstration



- Observation of the effects of sound and vibration on matter (cymatics)
- Study of the geometries formed on physical objects
- Acoustic properties of materials

- Music production in industry-standard formats



- Multichannel audio recording facilities and equipment
- Editing suite
- Audio mixing and mastering in the most common formats in the industry
- Music sequencing with virtual instruments

CAM_LAB UTAD (Infrastructure reinforcement proposal) - Salviano F.S. Pinto Soares July, 2020

- MIDI protocol applied capabilities



- MIDI sequencing and automation
- Remote control of external physical units
- MIDI editing
- Remote control of software workstations and instruments

Collaborators and relations

- **Salviano Pinto Soares**, Audio Engineering Society Portugal and UTAD, (<https://aes.org.pt/>)&(<https://www.linkedin.com/in/salviano-soares-285b3423/>),
- **António J. de Oliveira** ((Audio and Analog Electronics Consultant, Lisboa, Portugal), <https://ajoliveira.com/>
- **Eduardo Moreira** ((producer and sound engineer at RubyDiscos studios, Chaves, Portugal),
<https://www.linkedin.com/in/eduardo-moreira-860653174/>)
- **Telmo Fernandes** , Anechoic chamber, IPL/IT Leiria, Portugal (, <https://www.linkedin.com/in/telmo-fernandes-bb63bb88/?originalSubdomain=pt>);
- Marco Conceição, **ESMA**, Porto, <https://www.linkedin.com/in/marco-concei%C3%A7%C3%A3o-1019469a/>
-

If done according to these guidelines and suggestions, the mentioned audio lab should be capable of the following tasks:

- Multitrack (simultaneous) audio recording in industry-standard specs (limited to available channels);
- Editing, mixing and introductory audio mastering;
- Mono, stereo and stereo-surround audio manipulation and reproduction;
- Sound-design, sound-foley, ADR and other audio-for-multimedia operations;
- Sound-based programming environments can be made operational (e.g. MAX/MSP, PureData, C-sound, Reaktor and others);
- Audio-for-videogames solutions can also be installed (Fmod);
- Music production;
- MIDI-sequencing and automation;
- Video editing and post-production can be done too (namely if there is a capable GPU with dedicated memory);

CAM_LAB (Calibration, Audio and MIDI LAB) members&connections:

- **Salviano Filipe Silva Pinto Soares**
 - (<https://aes.org.pt/>)&(<https://www.linkedin.com/in/salviano-soares-285b3423/>)
 - Presidente da Direção Nacional da *Associação Portuguesa de Engenharia de Áudio* desde 18 de janeiro de 2020..
 - Membro da *Audio Engineering Society* n.º61582, New-York – USA, November 2009 (<http://www.aes.org/>)
 - Professor Conselheiro para a organização *Grupo de Áudio da Audio Engineering Society da UTAD* com o Núcleo de Estudantes de Engenharia Electrotécnica da UTAD - NEUTAD, desde abril de 2007.



António J. de Oliveira

<https://ajoliveira.com/>

- Audio and Analog Electronics Consultant, Lisboa, Portugal)
- <https://www.linkedin.com/in/ajoliveira/>



- Eduardo Moreira (producer and sound engineer at RubyDiscos studios, Chaves, Portugal)
 - <https://www.linkedin.com/in/eduardo-moreira-860653174/>
- Sound engineer and producer at RubyDiscos studios (Chaves, Portugal)
- Former assistant guest lecturer at UTAD (Audio Production)
- Masters in Music Technology at CIT/CSM (Cork, Ireland)
- BA in Music Production and Technologies at ESMAE (Porto, Portugal)
- Audio technician at Solverde Hotel Casino Chaves and Centro Cultural de Chaves
- Music producer with international discography



Telmo Fernandes,

IPL/IT Anechoic chamber, Leiria, Portugal

<https://www.linkedin.com/in/telmo-fernandes-bb63bb88/?originalSubdomain=pt>

PhD in Radiocommunications by the Univ. of South Wales, UK.

- Professor at IPL – Electrical Engineering Department
- Senior Member of the IEEE
- Researcher at the Institute of Telecommunication – Antennas and Propagation Group Leiria
- Founder of Digiwest, Wireless and Embedded Solutions, Lda., an electronics development and prduction company.



- Anechoic Chamber Characteristics:
- Antenna Radiation pattern measurement Fully Anechoic Chamber.
- RF Frequencies ranging from 400MHz to 60GHz
- Dimensions: 6x5x3 (w x d x h) meters
- Fully customizable measurement setup.



Spatial Recording

Acoustic Musical Laboratory - <https://www.esmae.ipp.pt/investigacao/projetos>

<https://www.cienciavita.pt/EE19-63EA-4A72>

<https://www.linkedin.com/in/marco-concei%C3%A7%C3%A3o-1019469a/>

Links de interesse - <https://www.tsf.pt/vida/ciencia-e-tecnologia/microscopio-eletronico-mostra-o-que-faz-a-agulha-no-vinil-4639926.html>:

Trabalhos de mentoria

Trabalhos de PL II / Eletrónica à Físico QUímica / Projetos de Arduino passados

Anexo_E

Artigos Publicados

(sucédâneos a projetos finais de *Processamento Digital de Sinal*)
UTAD

Signal Processing Interpolation and Problem Conditioning Educational Workbench

Abstract

This work presents a newer and more complete version of an educational tool to be used in signal processing interpolation-related subjects. Besides the consolidation of acquired theoretical knowledge, the tool allows now its users to apply three error geometry patterns, test minimum or maximum dimension signal reconstruction algorithms and problem conditioning through the analysis of the matrix spectral radius or the condition number: these new features gives the possibility to alter the problem definitions to the desired goal before the reconstruction begins. The time unit that measures the algorithms performance is $\mathcal{O}(n \log n)$ thus independent from the machine's architecture. A video of the developed tool can be seen in: https://www.dropbox.com/s/t6yituy31ramxse/FILME_1.avi.

Keywords: signal reconstruction, educational tool, interpolation, oversampling, fast Fourier transform (FFT), condition number, problem conditioning.

1. Introduction

Nowadays, namely in engineering areas, there are a great amount of subjects that require a large and deeper knowledge of theoretical aspects related to a specific scientific area. Sometimes students don't really understand the need to have skills on important concepts and lack the perception to understand its utility or even if they really work. During the last decades, with the quick spread of information technologies, some new tools arose whose purpose is to ensure and consolidate the theoretical concepts taught in regular teaching classes. The main idea is to give students a way to experiment concepts taught in classes, as well as giving them the opportunity to test them in different scenarios. With these tools students and other kind of users can practice and enhance skills by themselves as self-taught persons.

In this paper a new version of the "Signal Processing Interpolation Educational Tool" (SPIEW) is presented (Costa M., *et al.*, 2012). In the first version we propose a new and easy-to-use tool which will help students consolidate concepts of signal processing and complement their knowledge by enabling graphical results after using complex signal reconstruction algorithms. The tool permits to achieve calculation in just one hit or step by step through the iterative reconstruction refinement process and the assessment of the reconstructed signals.

Now we present an improved version of SPIEW. Its main new features are: the possibility to apply three error geometry patterns (interleaved, random and burst); control the redundancy of the signals by applying a proper oversampling factor, r , in the band limiting operation; test two different reconstruction algorithms and the usage of the iterative and direct calculation methods; the previously set up the problem condition through the analysis of the spectral radius of the iteration matrix (in the case of iterative methods are used) and the computation of condition number in the case of the direct method is used. In this way, its user can employ these parameters to know in advance if the problem will converge to an accurate solution, even before running one of the three reconstruction methods available. A previous analysis of key parameters permit to chose the most suitable method to be used in the signal reconstruction.

Since all the chosen methods use the FFT computation, $\mathcal{O}(n \log n)$, the time necessary to compute a FFT is used as reference. So, the performance is measured through a relative Time Unit (TU) that is the time to compute a FFT in a particular machine.

2. Reconstruction Algorithms

The signal reconstruction algorithm used in SPIEW was the discrete version of the Papoulis algorithm (Papoulis A., 1975): the Papoulis Gerchberg algorithm (Ferreira, P. J. S. G., 1-1994) (Jones, M. C., 1986). A thorough description of this maximum dimension algorithm can be found in (Costa M., *et al.*, 2012). The Papoulis algorithm is still available to run but a new minimum dimension reconstruction algorithm was included like the one used in (Neves F., *et al.*, 2008). Two different methods can be used to implement this algorithm: iterative and non-iterative calculation. We also call the non-iterative method the direct calculation method. This way students can test and compare maximum and minimum dimension algorithms with direct and iterative formulations. In the next subsections the minimum dimension algorithm is presented.

2.1 Minimum Dimention Algorithm

This subsection describes a minimum dimension algorithm which also requires band-limited signals of finite-dimension similarly to the Papoulis Gerchberg algorithm.

Thus the signal

$$x = Bx \quad (1)$$

where B is a band-limiter operator matrix (Ferreira, P. J. S. G., 2-1994).

The minimum dimension algorithm is a system of only l equations corresponding to the l unknown samples when the total number of samples is n and $n > l$.

To establish the basic concepts, the specific case of an original signal $x[n]$ with length $n=5$ is used, i.e., $x[n] = \{x_1, x_2, x_3, x_4, x_5\}$.

For this signal, Eq. (1) becomes

$$\begin{aligned} x_1 &= B_{11}x_1 + B_{12}x_2 + B_{13}x_3 + B_{14}x_4 + B_{15}x_5 \\ &\dots \\ x_5 &= B_{51}x_1 + B_{52}x_2 + B_{53}x_3 + B_{54}x_4 + B_{55}x_5 \end{aligned} \quad (2)$$

For reconstruction purposes let us assume that the 2nd and 4th samples of $x[n]$ are lost. Then, because only the unknown samples are to recover, the set of equations, Eq. (2), will be limited to those including the unknown samples. In each of these equations, we are interested in separating the right side terms of Eq. (2) containing the unknown samples (x_2, x_4) from those containing the known ones. These yields,

$$\begin{aligned} x_2 &= B_{21}x_1 + B_{22}x_2 + B_{23}x_3 + B_{24}x_4 + B_{25}x_5 \\ x_4 &= B_{41}x_1 + B_{42}x_2 + B_{43}x_3 + B_{44}x_4 + B_{45}x_5, \end{aligned} \quad (3)$$

which is equivalent to

$$\begin{bmatrix} x_2 & x_4 \end{bmatrix} = \begin{bmatrix} B_{22} & B_{24} \\ B_{42} & B_{44} \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \end{bmatrix} + \begin{bmatrix} B_{21} & B_{23} & B_{25} \\ B_{41} & B_{43} & B_{45} \end{bmatrix} \begin{bmatrix} x_1 \\ x_3 \\ x_5 \end{bmatrix}. \quad (4)$$

Let us denote by u the subset of the original signal $x[n]$ which contains the unknown values. In this case, $u = \{x_2, x_4\}$ is of cardinality $l=2$. Also, let us define $U = \{i_1, \dots, i_l\}$ as the set of subscripts of l unknown samples in $x[n]$. In the present case, $U = \{2, 4\}$. Therefore, Eq. (4) can be written as

$$x_i = \sum_{j \in U} B_{ij}x_j + \sum_{j \notin U} B_{ij}x_j; \quad i \in U \quad (5)$$

or, in matrix form

$$u = Su + h, \quad (6)$$

where S is a $l \times l$ principal submatrix of B , as defined in Eq. (4), and h is the $(n-l)$ dimensional vector in the second sum of Eq. (5), which is a linear combination of the known samples of $x[n]$. S is the system matrix (Ferreira, P. J. S. G., 2-1994). Theoretically, Eq. (6) has a unique solution regardless the number and distribution of the lost samples.

On the one hand, by successive operations, Eq. (6), becomes equivalent to

$$\begin{aligned} u &= Su + h \\ u &= (I - S)^{-1}h, \end{aligned} \quad (7)$$

and a basis to implement the non-iterative method is so established. On the other hand Eq. (6) suggests the needed formula to the iteration process

$$u^{(i)} = Su^{(i-1)} + h, \quad (8)$$

for the case where a non-relaxation method is used. Then $u^{(i)}$ is obtained at iteration i and the solution is given by the limit

$$u = \lim_{k \rightarrow \infty} u^{(k)}, \quad (9)$$

regardless of $u^{(0)}$.

Direct calculation of u as given in Eq. (7) has the advantage of being done in one single step and gives an exact solution, providing that $(I-S)^{-1}$ exists. The drawback is that in case $(I-S)^{-1}$ does not exist or the calculation is not possible and no reliable approximation is possible. In practice, there are several factors which may lead to serious difficulties in calculating the inverse of $I-S$. For example, if one of the eigenvalues of S , $\lambda_i(S)$, is close enough to the unity, then the computation of $(I-S)^{-1}$ may become very difficult, even impossible, which leads to an ill conditioned problem. In such cases an iterative method is advantageous because it may be used to circumvent this difficulty and to find a relatively accurate approximation for the solution, u . Despite the fact of having an ill conditioned problem, in direct calculation method such problem is impossible to solve whereas in iterative methods an approximation is always possible to be found, though its accuracy may not be very high. The conditions under which these equations provide a solution for u can be found in (Ferreira, P. J. S. G., 1-1994) and will be discussed in the next subsection.

3. Convergence Analysis

It is necessary that $(I-S)^{-1}$ exists or the iterative processes converge to find a problem solution. It is demonstrated in [3] that for low pass signals of dimension n with s known samples and q nonzero harmonics, reconstruction iterative algorithm converges if:

$$(n-q) \geq (n-s) \quad (10)$$

which is equal to

$$q \leq s. \quad (11)$$

It is important to refer that $q=2m+1$ where m is the sample bandwidth of the Low Pass Filter (LPF) [1]. So Eq. (11) can be rewritten as follows:

$$s \geq 2m+1. \quad (12)$$

which establishes a sufficient condition on the density of the known samples. More precisely, the number of known samples s must exceed twice the number of nonzero harmonics. For low pass signals, the condition is consistent with the well-known classical sampling theorem.

The sufficient condition expressed by (12) is obviously necessary too. After all, low-pass signals with $q=2m+1$ nonzero harmonics are described by $2m+1$ parameters, and therefore one cannot expect to reconstruct them from less than $2m+1$ samples.

In order to use the algorithms available in the new tool it is necessary that all signals are band limited. This means that Eq. (12) establishes a necessary condition in our tool. Because only low pass signals are used, the necessary condition is met.

For the specific case of the Papoulis-Gerchberg algorithm, a relaxation constant, μ , is introduced to improve the convergence rate, as stated in the previous tool version. In (Ferreira, P. J. S. G., 1-1994) it is demonstrated that the algorithm converges if both conditions are satisfied: $0 < \mu < 2$ and $s \geq 2m+1$ Eq. (12). Because we have considered $\mu=1$ and used low pass signals the iterative process converges.

Regarding minimum dimension method, the characterization of the system matrix, S , is fundamental to infer about the existence of solution to the problem formulated in Eq. (2). Namely its eigenvalues, $\lambda_i(S)$, and in particular its spectral radius, $\rho(S)$, condition the problem. $\rho(S)$ values close to 1 (the unity) make the computation of $(I-S)^{-1}$ very difficult, even impossible. Theoretically, the condition $\rho(S) < 1$ would always guarantee the solution possible. However, due to the some computational representation constraints, small variations on input values can cause big variations on the output and cause the problem to be ill conditioned. In these cases a new implemented parameter must to be taken into account: the condition number of $(I-S)^{-1}$, presented latter below.

According to these conditions, it is possible to put a reconstruction problem into a point such that it is well conditioned. It is known that the eigenvalues of the system's matrix S depend on the distribution of the missing samples. In particular, its spectral radius is more likely to be unitary for burst distributions rather than for equidistant missing samples (Ferreira, P. J. S. G., 3-1994). In the case of signal reconstruction it is interesting to note that, if the distribution of the missing samples $U = \{i_0, i_1, \dots, i_{L-1}\}$ is equidistant by some fixed integer $p \geq 1$,

that is, $U = \{i_0p, i_1p, \dots, i_{l-1}p\}$, then the eigenvalues λ_i of S are given by $\lfloor rp \rfloor / p$, i.e.,

$$\frac{\lfloor rp \rfloor}{p} \leq \lambda_i(S) \leq \frac{\lceil rp \rceil}{p} \leq 1 \quad (13)$$

where $\lfloor rp \rfloor$ denotes the greatest integer less than or equal to rp and $\lceil rp \rceil$ denotes the smallest integer equal or greater than rp . In the particular case of $r = \lfloor rp \rfloor / p$, the eigenvalues of S are all the same $\lambda_i(S) = r, \forall i$.

Given the above analysis, it is possible to put the problem into a well-conditioning point by properly selecting the gap between the missing samples. Then it is possible to put $\lambda_i(S)$ close to either r or its multiples, regardless of the number of missing samples. By using an appropriate choice of the oversampling and interleaving factors, r and p , respectively, it is possible to put $\lambda_i(S)$ less enough than the unity in order to control the reconstruction accuracy and processing speed (i.e., such that $m \times r$ is an integer).

Fig. 1 shows the maximum and the minimum eigenvalues of S as a function of the interleaving factor, p , for a given bandwidth, defined by $r=0.6$. As the figure shows, greater values of p lead to better-conditioning problems because $\lambda_{\max}$ decreases as p increases. Also, when the product $m \times r$ is an integer, all eigenvalues are equal since they are $\lambda_i(S) = r$, as stated before. In Fig. 1, this occurs for $p=5$ and $p=10$.

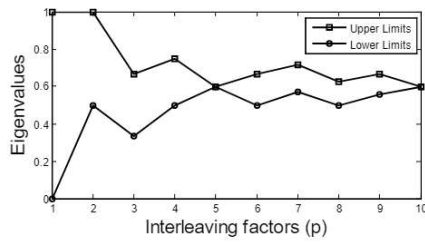


Figure 1. Spectral radius vs. interleaving factor, $r=0.6$

To better understand and control the problem conditioning, in the presented tool, some new parameters are calculated: the spectral radius of the S matrix, delimited in Eq. (13) and the Condition Number of the $(I-S)$ matrix (Eq. (14)) (Golub, G., Van Loan, C., 1996) (Kincaid, D., Cheney, W., 2002). Through the analysis of both values, and also the convergence condition, Eq. (12), the user can know in advance if the reconstruction algorithm will converge to an accurate solution or not. Therefore through the properly chosen condition point the signal reconstruction is guaranteed.

In practice, when $\rho(S)$ tends to be very near or even equal to the unity, the problem may become ill-conditioned, even if it is theoretical well-conditioned due to the fact that $\rho(S) < 1$. Condition numbers near or above 10^{15} do not generally lead to well-posed problems, even with double precision arithmetic (64-bit floating point, in many machines). The condition number of matrix A , $k(A)$, is calculated according the equation presented in (Kincaid, D., Cheney, W., 2002):

$$k(A) = \|A\| \|A\|^{-1} \geq \frac{|\lambda_{\max}(A)|}{|\lambda_{\min}(A)|} \quad (14)$$

4. Developed Tool

This tool was developed using Matlab (Matlab, 2012) and is to be used with band limited signals. Signals to be treated in this version are: frame sequences or still images. In the first version of SPIEW there was only one

signal corruption method, the interleaved decimation and only a maximum dimension algorithm, the Papoulis Gerchberg. An improvement was the addition of two geometry patterns error and a new reconstruction algorithm. This new version allows the signal corruption both in a random and contiguous or burst way: the Papoulis Gerchberg is also available. The next subsections describe each of the stages that constitute a complete cycle of using the tool.

3.1 Extract Signals Operation

To start using the SPIEW application the user must push the “Open File” button to choose a type of signal: as previously refers it can be the first frame of a sequence or a still image, here we have used Multiview Video plus Depth (MVD) 3D images (Rusanovsky, D., *et al.*, 2013). This is the only active button when the user starts the tool. The rest of the buttons will be activated sequentially to force the user to perform all the correct actions in the correct order. In Fig. 2 we can observe the first interface of the application after doing the opening file operation.

After the chosen signal is open and displayed, some information regarding it is showned. If the user has chosen the sequence frames to work with the “Play” button is enabled. In the case of a still image this button will not be activated. To perform the signals extraction from each type of signal the user must click the “Extract Signals” button. Each signal is made by: all the values corresponding to the same pixel along the frame sequence or each line of the still image. When this button is pressed the signals extraction is done and a progress bar shows the user the remaining time to complete the task (Fig. 3).



Figure 2. SPIEW initial interface, still image file

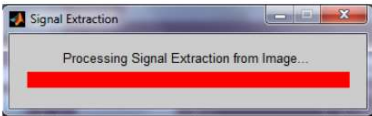


Figure 3. Signal Extraction Progress Bar

It is now available to the user the number of signals extracted from the entire sequence, as well as the total number of samples of each extracted signal.

3.2 Band Limiting Operation

After the signals extraction the user can now perform the band limiting operation for each of the extracted signals. This is a necessary condition to all the reconstruction algorithms available in this tool. For that the user must chose the value of the oversampling factor, r . It indicates the percentage of redundancy that the signals will have after the filtering operation and also limits the number of samples that can be lost afterwards. To initiate this

process the user must press the “*Filter Signals*” button. Subsequently a progress bar appears showing the estimated time to finish this operation. Eventually a message box shows the Low Pass Filter Bandwidth that corresponds to the r chosen value (Fig. 4). This is the value of m to verify Eq. (12).

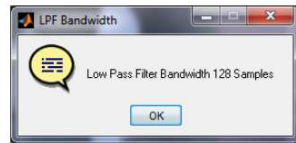


Figure 4. Band Limiting Operation – Low Pass Filter Bandwidth

3.3 Corruption Operation

The user is now ready to perform the corruption of the signals. As mentioned before, in the previous version of the tool, only one error pattern geometry was available to apply: the interleaved decimation. This new version has three possible error geometry patterns: interleaved, random and contiguous decimation (Fig. 5). This will allow the user to compare the reconstruction algorithms performance in different corruption scenarios.



Figure 5. Decimation process, available methods

In the case of the interleaved decimation corruption the user must choose a value for the jump, as shown in Fig. 6. He must choose values among 2, 4 or 8. After choosing the desired value to start the decimation process, the user must press the “*Apply*” button.

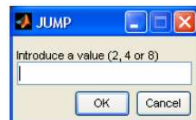


Figure 6. Choosing the jump value for the interleaved decimation method

In this particular case (interleaved decimation) the number of lost samples is automatically calculated and filled in when the decimation process is terminated. In the other two cases (random and contiguous) the user must choose how many samples he wants to lose in the decimation process (Fig. 7). If the user wants to visualize the results of both the band limiting and the decimation operations, he must choose a specific pixel by its number and press the “*View*” button. A figure with the original, band limited signal (observed signal) and a decimated version of that signal appears (Fig. 8).

It is important to refer that both band limiting and decimation operations are performed for all signals extracted from the video sequence or still image.



Figure 7. Contiguous decimation, 30 lost samples

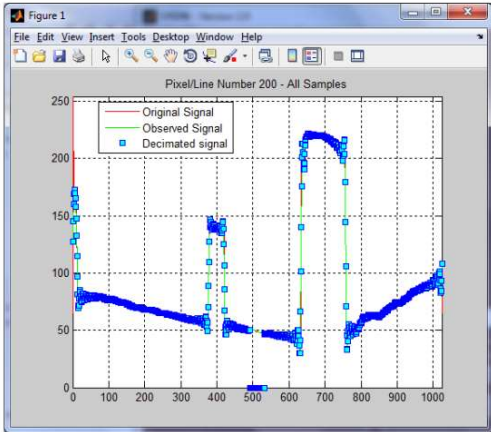


Figure 8. Original, Observed and Decimated versions of the signal corresponding to line number 200.

3.4 Convergence Analysis and Reconstruction

We are now ready to move into the next interface of SPIEW. To do so the user must press the “Next” button in the current interface. The reconstruction window appears. Here the user must choose what reconstruction algorithm is to run through a pop-up menu with three options: Papoulis-Gerchberg, Minimum Dimension, Direct method or Minimum Dimension, Iterative method (Fig. 9).



Figure 9. SPIEW, Choose the Reconstruction Algorithm.

Before starting the reconstruction of the chosen signal, the user is now able to see the problem conditioning data (spectral radius and condition number). It is through the analysis of these two values that the user will know in advance if the chosen algorithm and method will converge. To visualize these parameters, the user must press the Problem Conditioning button (Fig.10). Here the spectral radius and condition number concerning the problem under study are exhibited.

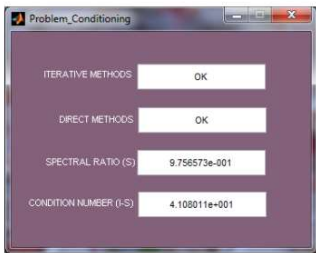


Figure 10. Problem Conditioning.

After analysing these values the user knows if the problem will converge to an accurate solution or not. He will also know which algorithm to choose to obtain an accurate solution. As mentioned in the previous section for iterative methods, a spectral radius smaller enough than the unity guarantees in practice the iterative algorithms' convergence. For direct methods a condition number smaller than 10^{15} guarantees a solution to the problem. More details about this issue will be presented in the Test and Results section.

After choosing the reconstruction algorithm the user must choose which signal we want to reconstruct and in the case of the iterative methods, he must also indicate the maximum number of iterations. To start reconstruction the user must press the "Run" button. After the signal reconstruction, besides the reconstruction results, including the reconstructed signal view, the user can also see the signal's energy and power. This will allow the comparison of the reconstruction of signals, or even signal segments, with more or less energy.

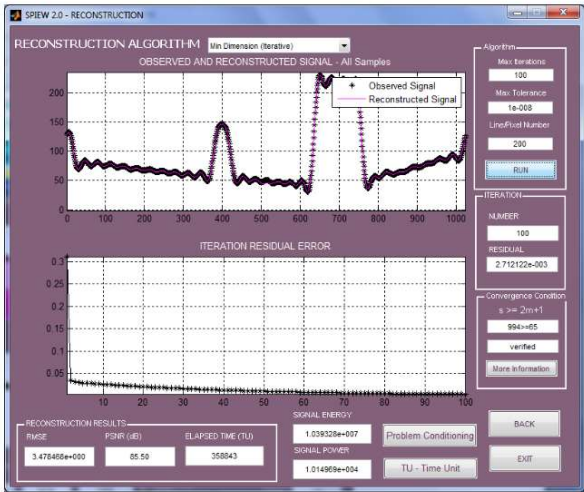


Figure 11. Reconstruction results using the Minimum Dimension, Iterative method (Line 200).

After the signal reconstruction has started, the window shown in Fig. 11 allows the user to visualize the iteration number, its residual error and if the convergence condition is verified. There is also an explanation button to help the user to understand the expression used to verify the convergence. It is the "More Information" button in the convergence condition area (Fig. 12).

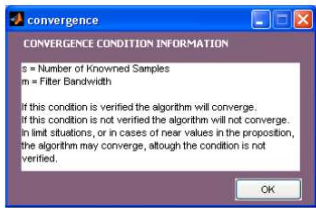


Figure 12. Convergence condition information.

At the end of the reconstruction several parameters to help the user to evaluate the reconstruction and compare the available algorithms are calculated. Those parameters are: the Root Mean Squared Error (RMSE), the Peak Signal to Noise Ratio (PSNR) and the Elapsed Time measured in TU. By pressing the “TU – Time Unit” button the user can access this information (Fig. 13). The tool also shows a graphic of the Residual Error calculated in each iteration (herein called the Iteration Residual Error). In it the user can see if the chosen algorithm tends faster to the maximum tolerance.



Figure 13. TU, time unit used to compare the reconstruction algorithms results.

When the user chooses the Minimum Dimension algorithm, Direct method, there are parameters that are not needed or calculated, such as the maximum number of iterations, the iteration number and its residual error, and instead of the Iteration Residual Error graphic, only observed and reconstructed signals are showed (Fig. 14).



Figure 14. Reconstruction results using the Minimum Dimension algorithm, direct method, to reconstruct line 200.

4. Tests and Results

SPIEW was tested several times in order to better explain the diverse contents that it addresses to its users. Still during its development some opinions of colleagues and students were taken into consideration to make this tool more effective and user-friendly. For example, its GUI was altered quite a few times in order to comply with the obtained feedback. Concerning the SPIEW features, first the three error geometry patterns were applied and tested. Clearly the most unfavourable case to the reconstruction efficiency is the contiguous (burst) lost of samples.

Through those tests we concluded that the convergence condition must be verified to guarantee the algorithms' convergence. Furthermore a spectral radius less than 1, preferably around 0.5, will ensure the well functioning of the iterative methods (convergence rate, accuracy, etc.). But in the specific case of the direct method the condition number is a key parameter to infer about how acceptable a solution using this method is. Values lower than 10^{15} lead to accurate solutions even when the iterative methods no longer converge.

In the following figures we present the results obtained from the tests achieved with $r = 12.5\%$ and a burst error geometry pattern for the lost of samples. It is important to refer that the burst, in this time, is located in the middle of the signal. The purpose is to compare different cases resulting from different number of lost samples by increasing the burst size. Fig. 15 shows the problem condition values for the case of 10 contiguous lost samples.

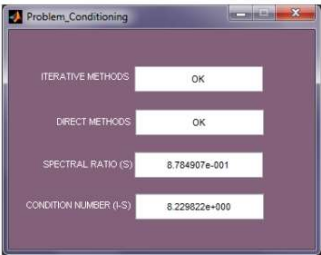


Figure 15. Problem conditioning values for 10 contiguous lost samples.

Through the analysis of both spectral radius and condition number we conclude that both iterative and direct methods will converge. In Fig. 16a and Fig. 16b, the results obtained using both methods are shown.

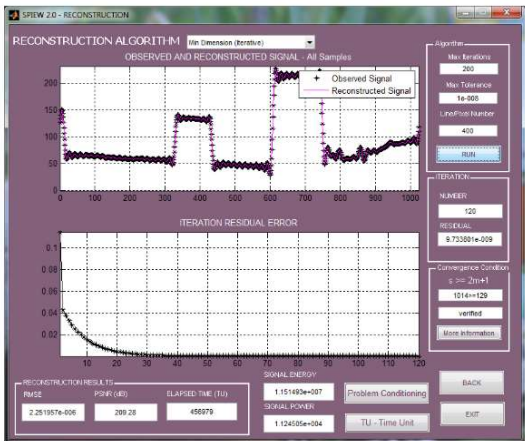


Figure 16a. Reconstruction results using the Minimum Dimension algorithm, Iterative method (Line 450).

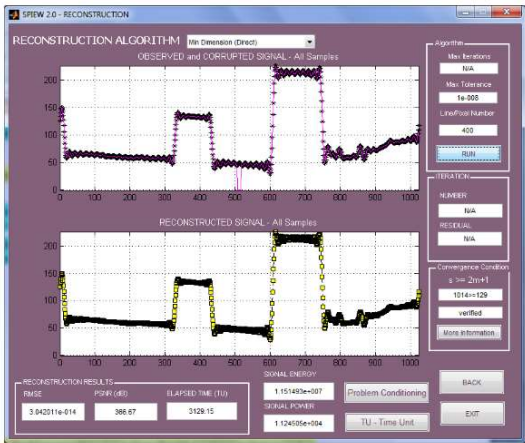


Figure 16b. Reconstruction results using the Minimum Dimension, Direct method (Line 450).

The next case results from increasing the number of the lost samples to 60. Fig. 17 shows the problem condition values that permit to achieve the respective convergence analysis.

Clearly the iterative methods will no longer converge to an accurate solution, since spectral radius, $\rho = 1$. However, it is possible to see that by using the direct method we will be able to obtain a solution, as shown in Fig. 18a and Fig. 18b.

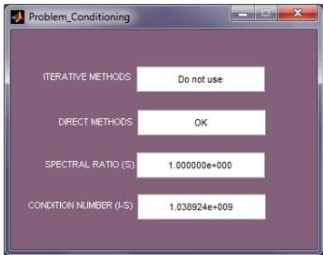


Figure 17. Problem conditioning values for 60 contiguous lost samples.

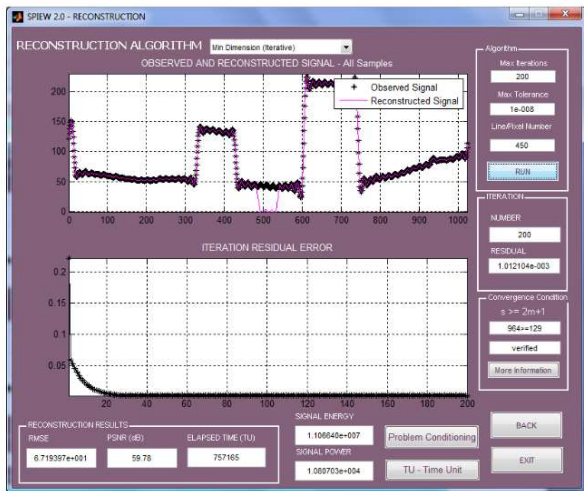


Figure 18a. Reconstruction results using the Minimum Dimension algorithm, Iterative method (Line 450).

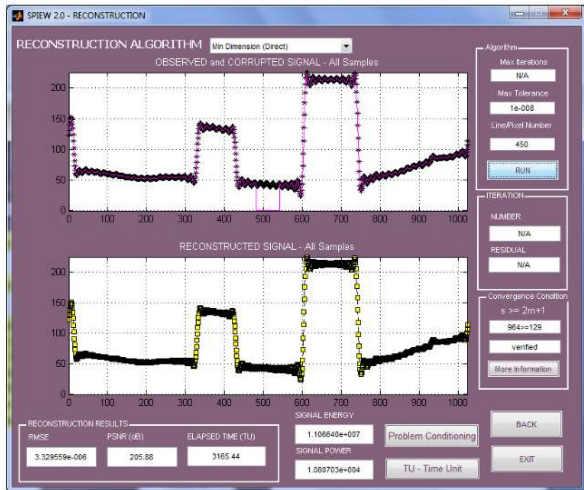


Figure 18b. Reconstruction results using the Minimum Dimension algorithm, Iterative method (Line 450).

In the next case the number of lost samples was increased up to 100. The problem conditioning values are shown in Fig.19. As it can be seen in Fig. 19, the condition number reached the value of 10^{15} . Since we used double precision (32 bits precision), it is not expected that direct mode can calculate an accurate solution. The computed solution would not be to trust on. Obviously, if we increase the burst size beyond 100, signal remains unrecoverable. Concerning the direct method, as we expected, since we use double precision, the condition number has reached a value of 10^{15} , and if we increase even more the burst window, the signals cannot be reconstructed and the direct method won't converge to a solution (see Fig. 20).

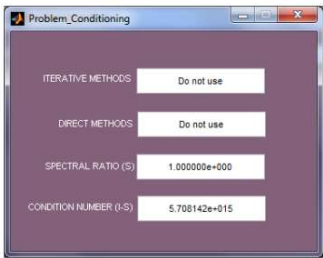


Figure 19. Problem conditioning values for 100 contiguous lost samples.



Figure 20. Reconstruction results using the Minimum Dimension Algorithm, Direct method (Line 450).

5. Survey Results

After obtaining the limit values to the conditioning of the problem, the tool was presented to a group of test subjects. The tool was properly explained through an oral explanation, and then a written tutorial was given to all of them. Some examples were presented so that they could see the application performance under several different situations.

Thereafter they were asked to explore the tool freely. Afterwards they were given a quiz test to answer. In some issues the test group has shown more difficulty, namely concerning the band limiting operation, as for example its direct relation to the signals redundancy. At the end, trough experimentation, it became clear to them that the decrease of redundancy in the signals clearly limited the number of samples that can be lost.

To validate SPIEW a careful elaboration of the referred quiz test was achieved before it was given to the subjects. In order to be effective, important educational and psychological issues were taken into account on designing it (Minium, E. W., et al, 1993), (Cohen, L., et al, 2000). Seven questions were found as being representatives of the main aspects the tool must provide. Questions are:

- 1. "Which oversampling factor would you choose to allow you to lose a larger number of samples in the signals?"
- 2. "A high oversampling factor corresponds to a high or low value in the Low Pass Filter Bandwidth?"
- 3. "Which is, in your opinion, the worst error geometry pattern to reconstruct the signals?"
- 4. "In a good problem conditioning which reconstruction method would you choose? Why?"

5. "If the spectral radius of matrix S is smaller than 1, which of the algorithms can you choose to reconstruct the signals?"
6. "If the condition number has a value of 10^{15} can you reconstruct the signals using the available methods?"
7. "Do you consider the tool useful? Did your knowledge in the area improved?"

The results obtained in quiz done to the subjects are showed in Fig. 21.

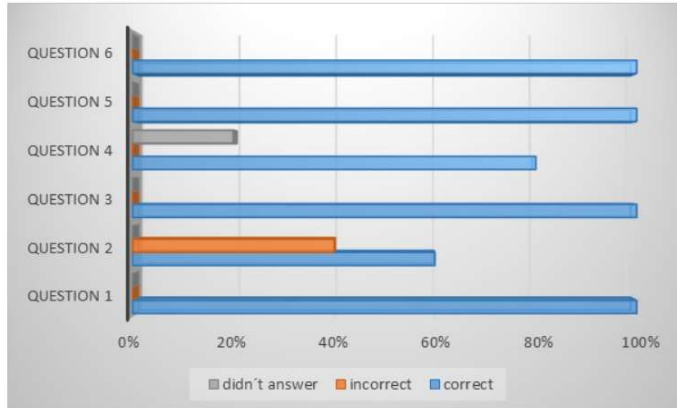


Figure 21. Test results.

As it can be seen, 100% of the subjects answered correctly to questions 1, 3, 5 and 6. 20% didn't answer to question 4. Concerning question 2, 60% answered correctly and 40% incorrectly. Finally, concerning question 7, they all consider the tool helpful, and say that their knowledge has improved with it.

6. Conclusions and Future Work

Similar to its previous version, SPIEW allows indubitably its users to deal with signal reconstruction algorithms in a much friendlier and accessible way. By approaching the problem in a more graphical and interactive manner, the users were able to retain and consolidate concepts such as filtering, band-limiting, problem conditioning, reconstruction and algorithm dimensions. Other concepts like signal redundancy and how it affects the number of samples that can be lost are also addressed in this version of the tool, as well as the three different error geometry patterns available and the identification of the worst case scenario to the reconstruction can also be tested.

Depending on the problem condition analysis the user can know in advance if the signal reconstruction is possible, or which is the best algorithm to reconstruct it and confront its previous hints or results with precise predictions. It permits user to enhance its knowledge and get the feeling about this kind of problems in an interactive way. This analysis is done through the convergence condition, spectral radius and condition number parameters. This allows the user to alter its problem conditioning to the desired goal before the reconstruction begins.

To compare the results obtained using the three different algorithms available in the tool the RMSE and PSNR of the reconstructed signals are computed. In the previous version of the tool only one reconstruction method was available, the Papoulis-Gerchberg algorithm. The reconstruction time, measured in TU (Time Units) can also be used to compare the performance of the three algorithms. Besides this, in the specific case of the iterative methods, the maximum number of iterations necessary to reconstruct the signal and the graphic evolution of the Iteration Residual Error can also act as comparative values between the iterative methods performance. In the direct method, instead of the iteration error evolution graphic, both the reconstruct and corrupted signal versions are shown.

This application is to be used in Digital Signal and Image Processing related subjects. So far, the users' receptivity has been encouraging. Currently we have used in this tool high frame rate sequences and still images

obtained with the MVD method (Rusanovskyy, D., *et al*, 2013), still we believe it can be used using imagens from different areas, such as medical areas for example.

As future work we aim to add new features to the developed tool and conduct more tests, namely test and add new algorithms to the developed tool (Karthik, M.; Prabhu, K., 2010). The aim is to choose the appropriate method according to a previous analysis of the problem conditioning.

References

- Cohen, L., Manion, L., Morrison, K., (2000). *Research Methods in Education*. Londres: RoutledgeFalmer.
- Costa, M., Barroso, J., Soares, S., (2012). Signal Processing Interpolation Educational Workbench. *Computer Applications in Engineering Education*, Wiley InterScience, 20(2):356-363. (DOI: 10.1002/cae.20402).
- Ferreira, P. J. S. G., (1-1994). Interpolation and the Discrete Papoulis-Gerchberg Algorithm. *IEEE Transactions on Signal Processing*, 42(10):2596-2606. (DOI: 10.1109/78.324726).
- Ferreira, P. J. S. G., (2-1994). Noniterative and fast iterative methods for interpolation and extrapolation, *IEEE Transactions on Signal Processing*, 42(11):3278-3282, 1053-587X. (DOI: 10.1109/78.330398).
- Ferreira, P. J. S. G., (3-1994). The Stability of a Procedure for the Recovery of Lost Samples in Band-Limited Signals. *IEEE Transactions on Signal Processing*, 42(11):3278-3282 1053-587X. (DOI: 10.1016/0165-1684(94)90067-1).
- Golub, G., Van Loan, C., (1996). *Matrix computations*. The Johns Hopkins Univ Press, 3rd. ed.
- Jones, M. C., (1986). The Discrete Gerchberg Algorithm. *IEEE Transactions on Acoustics Speech and Signal Processing*, 3(3):624-626. (DOI: 10.1109/TASSP.1986.1164861).
- Karthik, M.; Prabhu, K. (2010). On the Eigenvalues of Matrices for the Reconstruction of Missing Uniform Samples. *IEEE Transactions on Signal Processing* 58(5):2896-2900. (DOI: 10.1109/TSP.2010.2041277).
- Kincaid, D., Cheney, W., (2002). *Numerical Analysis: Mathematics of Scientific Computing, Third edition*. The Brooks/Cole - Thompson Learning, 3rd edition ed.
- Matlab, (2012) is a product of The MathWorks, Inc. URL: <http://www.mathworks.com/>
- Minium, E. W., King, B. M., Bear, G., (1993). *Statistical Reasoning in Psychology and Education*. New York: John Wiley & Sons, Inc.
- Neves, F., Soares, S., Reis, M. C., Tavares, F., Assunção, P., (2008). VoIP reconstruction under a minimum interpolation algorithm. *IEEE International Symposium on Consumer Electronics, ISCE 2008*. (DOI:10.1109/ISCE.2008.4559434).
- Papoulis, A., (1975). A New Algorithm in Spectral Analysis and Band-Limited Extrapolation. *IEEE Transactions on Circuits and Systems*, 22(9):735-742. (DOI:10.1109/TCS.1975.1084118)
- Rusanovskyy, D., Hannuksela, M., Su, W., (2013). Depth-based Coding of MVD Data for 3D Video Extension of H.264/AVC, 3D Res. 04. (DOI: 10.1007/3DRes.02(2013)6).

Digital Sound Processing using Arduino and MATLAB

Sérgio Silva^{1,3}

¹School of Sciences and Technology
Engineering Department
UTAD, Vila Real
Email: sergio.silva@inescporto.pt

Salviano Soares^{1,2}

²IEETA
UA, Aveiro, Portugal

António Valente^{1,3}

³INESC TEC
INESC Technology and Science
(formerly INESC Porto, UTAD pole)
Porto, Portugal

Abstract

Over the last decade, impelled by the huge open source software community support, the low cost Arduino platform presents itself as an alternative for digital sound processing. Although Arduino is generally used for small applications for the artistic and maker community, its built-in Analog to digital converter can be used for sound capturing, processing and reproduction. Equipped with a powerful AVR 8 bit RISC microcontroller, the Arduino, can achieve up to 200kHz with a 10 bit resolution according to the Atmel ATmega328P datasheet that is the AVR core that we are going to focus on this article.

Realizing the hardware potential, software suppliers like Matworks or National instruments, have included the Arduino packages on the software accessories of MATLAB and LABView.

This work presents some of the sound capabilities and specific limitations of the Arduino platform, enfacing its connection and installation with MATLAB software. A series of examples of the Arduino interface with MATLAB are detail and shown in order to facilitate users initiation of MATLAB and Arduino Digital Sound Processing enhancing education fostering.

Key words: Digital Sound Processing; MATLAB; Arduino; ADC/DAC; Sampling; Vocoder; FM Synthesis of Instrument Sounds

I. INTRODUCTION

The Atmel ATmega328P is the core of our Arduino and, as all the AVR cores, is equipped with a 10 bits analog to digital converter (ADC). An analog signal, as often call, is a continue steam of analog values and can be read or sampled a certain number of times per second, which is referred to as the sampling frequency (f_s). Figure 1 shows an analog signal with a D.C. component around 2.5Volts.

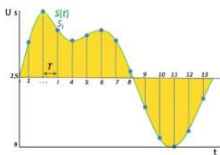


Fig. 1 - Analog Signal

To deal with sound signals center around 0 Volts (V) ranging [-5;5] (V) and to be able to read then with Arduino we need to do some level adjustment with a circuit like the one showed on figure 2.

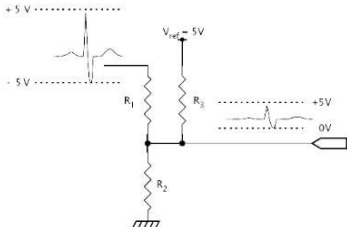


Fig. 2 - Simple Level Shifting circuit

To get the right input, usually R_3 and R_2 have the same value around 10 k Ω and R_1 is often replaced by a decoupling capacitor of 10 μ F (Coupling capacitors are used to block D.C. and pass A.C. that represents the music signal e.g).

There are different types of ADC architectures but most 8 bits AVR use successive approximation ADC, while 32 bits AVR uses comparative ADC. Figure 3 is a simplified diagram of the 8 bit AVR ADC.

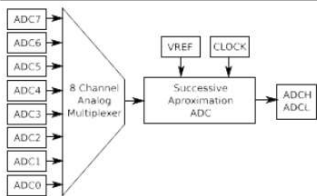


Fig. 3 - ADC Block diagram [1]

The AVR only has a 10 bit ADC and uses an 8 channel analog multiplexer to sample each of the 8 analog.

The ADC circuit takes samples between [13; 250] μ s and AVR has a dedicated clock that ensures the independency of conversion from other microcontroller parts. The conversion mechanism can be triggered either on demand or automatically.

The ADC conversion result is stored in two registers the ADC high and low. The result is obtained by the formula:

$$\left(\frac{V_{IN}}{V_{REF}}\right) * 1023.$$

If V_{IN} is 2.5 Volts the converted value will be 512 and if $V_{IN} = V_{REF}$ then the result is 1023.

The Arduino uses a simple command to start a single conversion the "*analogRead(pin number)*". If nothing else is performed the conversion is complete. Figure 4 shows both ADC Data Register, in this particular case the ADLAR bit is set and the result is left adjust (the ADLAR bit is bit number 5 of the ADMUX – ADC Multiplexer Selection Register).

Bit	15	14	13	12	11	10	9	8	
(0-7)	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCH
(0-7)	ADC1	ADC0	-	-	-	-	-	-	ADCL
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

Fig. 4 - ADC Data Register, ADCH and ADCL[1]

Remember that the ADC stores 10 bits the ADC0 ... ADC9 so if we read only ADCH we ignore the first 2 bits.

It is also possible to use ADCW which is not found in Atmel datasheet in order to get the ADC result. To use it just assign it to an unsigned integer:

```
unsigned int adc_value = ADCW;
```

Clock frequencies between 50 kHz and 200 kHz should be used to get the higher resolution. Higher frequencies above the 200 kHz will produce less bits resolution. The ADC prescaler, is set by the ADC Control and Status Register A (ADCSRA), as shown in Figure 5.

Bit	7	6	5	4	3	2	1	0	
(0-7)	ADEN	ADSC	ADIF	ADIF	ADIF	ADIF	ADIF	ADIF	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fig. 5 - ADCSRA - ADC Control and Status Register A[1]

Combining the 3 ADPS bits sets the relation from AVR clock frequency and the ADC clock is display in Table 1.

Table 1 – ADCSRA division factors

Division Factors	ADCSRA		
	ADPS2	ADPS1	ADPS0
2	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

Suppose we have system clock with frequency 16 MHz (16000000 Hz) and set division factor to 128 (Arduino default), then ADC clock frequency is:

$$\frac{16000000}{128} = 125000\text{Hz} = 125\text{kHz} \quad (2)$$

So in order to set the ADC Prescaler to 64 we should write the following setup instructions:

```
bitSet(ADCSRA,ADPS2);
```

```
bitSet(ADCSRA,ADPS1);
```

```
bitClear(ADCSRA,ADPS0);
```

The total conversion takes 14.5 clock cycles. AVR clock frequency is 16 MHz if we set the ADC prescaler to 2, the conversion period can be obtain using the formula:

$$T_{conv} = 14.5 * \frac{2}{16} = 1.8125\mu\text{s} \quad (3)$$

Table 2 presents the ADC conversion period times and frequencies for all possible combinations of the prescaler the conversion frequency has calculated using the formula $f_{conv}=1/T_{conv}$.

Table 2 - ADC conversion times and frequencies

ADC prescaler	T_{conv} (μs)	T_{nconv} (μs)	f_{conv} (kHz)
2	1,8125	5,6000	178,571
4	3,6250	7,1900	139,082
8	7,2500	10,5800	94,518
16	14,5000	17,0600	58,617
32	29,0000	30,0600	33,266
64	58,0000	56,1200	17,819
128	116,0000	112,0700	8,923

The differences between measure and calculated times can be explain by the use of the *micros()* function that has a resolution of about 4 μs [2]. As stated by André Bianchi is not entirely true because on our implementation the results show that this difference only applies to the first 3 measurements and all other presents smaller differences.

With the ADC prescaler set to 16 and a clock speed of 1 MHz we achieve a total of 58,617 samples per second ($f_s \sim 59\text{kHz}$) without compromising ADC resolution.

In fact the ADC accuracy also depends on the ADC clock. The recommended maximum ADC clock frequency is limited by the internal DAC in the circuitry conversion. For optimum performance, the ADC clock should not exceed 200 kHz, however, frequencies up to 1 MHz do not reduce the ADC resolution [3].

II. ADC IMPLEMENTATION

In order to perform the ADC measurements the following code was used:

```
unsigned long start;
unsigned long stop;
unsigned long ADC_value[100];
unsigned int j;
void setup() {
  Serial.begin(9600); // Begin Serial port
  pinMode(1, INPUT);
  ADCSRA &= ~(1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0); // ADC settings
  ADCSRA |= 1 << ADPS2; // set 1 MHz frequency
}
void loop() {
  start = micros(); // starts measurements
  for(j=0;j<100;j++) {
    ADC_value[j] = analogRead(0);
```



```

}
stop= micros();
Serial.println((stop - start));
for(j=0;j<100;j++) {
  Serial.println(ADC_value [j]);
}
delay(5000);
}

```

The operation $ADCSRA |= (1 \ll ADPS2)$; sets bit ADPS2 from the ADC Control and Status Register A and is equivalent to the instruction `bitSet(ADCSRA,ADPS2);`.

After processing we can use Pulse-Width Modulation (PWM) available in pins 3, 5, 6, 9, 10 and 11 of the AVR to convert it back to analog using an analog filtering stage to filter and smooth the sound wave.

III. DIGITAL SIGNAL PROCESSING

One of the main concerns in terms of sound processing is, of course, the amount of time available in computation of output samples because they must be ready to be consumed by the playback hardware avoiding glitches and other unwanted artifacts [2]. Nevertheless there are ways that involve delay (buffering techniques) to compute all the samples in time.

For our work, we study two different approaches for sound processing. The first one involves a more compact and simple processing where all computation is done by the Arduino and on a second approach the samples are sent to the laptop where some MATLAB programs do all the computation before send it back to playback on the Arduino.

So let's first analyze how can the Arduino playback the sounds and what kind of hardware do we need.

To generate high quality sound, from the output signal, one needs to add a few but important components: for example if we need a 8 bit resolution (8 bit video game music e.g)[4], we just need to use one PWM pin attach to a resistor and a capacitor in a low pass filter configuration as shown on figure 6 (Cut-off frequency of 22.104kHz when $R=1.8k\Omega$ and a $C=4nF$ capacitor).

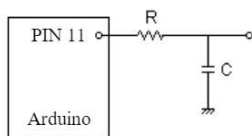


Fig. 6 - Low Pass Filter

In order to calculate the PWM frequency we should explain what PWM is and how it does works on AVR.

PWM depends on the microcontroller timer. This is a simple internal clock that counts up to some number, and then goes back down to zero. PWM is generated by these timers, by having an external pin go *high* when the timer hits zero, and then go *low* at some other number, which we

can vary. In this manner, it's possible to have an external pin stay high for a specific amount of time, without having to manually toggle it with the code.

A PWM waveform is generated from a counter by counting clock ticks, a register and a comparator [5]. The counter's purpose is to create the duty cycle resolution. One complete cycle of the counter is one period of the PWM. The counter's size (and thus the duty cycle resolution) is 8 bits, which equals to 256 values. So the 16MHz clock gets divided by 256 —because a cycle of the counter takes 256 ticks— and may give a maximum frequency of 62.5kHz. In order to get lower frequencies, we can divide your clock by another factor, known as the first clock division. This is also called pre-scaling, because it precedes the counter. Table 3 summarizes the overflow interrupt frequency for all possible values of prescaler.

Table 3 - Overflow interrupt frequency for prescaler values

PWM prescaler	f_{incr} (KHz)	$f_{overflow}$ (Hz)
1	16000	62500
8	2000	7812
32	500	1953
64	250	976
128	125	488
256	62.5	244
1024	15.625	60

The register and the comparator are used to set and create the duty cycle output. The register sets for how long during each period the output is high. For example for a 25% duty cycle, we would set the register to $256 \cdot 0.25 = 63$. The comparator compares the register's value with the current counter value and gives *high* in the output if the value of the former is equal or lower from the value of the latter and *low* otherwise.

For each counter we actually get 2 registers and 2 comparators, so we can get 2 PWM waveforms with different duty cycles but with the same frequency. Whenever is possible loosing precision, a faster PWM frequencies than 62,500 kHz on AVR is available.

Let's take some time to analyze the trade of between PWM frequency and noise floor.

The smallest sound that can be heard is correlated to the noise floor. This is the low level "hiss" heard in the background of most signals. For sound applications, there must be undetectable by the ear but, this is often not achievable with a single PWM generator. To get a lower noise floor, is necessary to use more bits, and the exact amount is set by the equation:

$$SNR_{(dB)} = (Bit\ Depth) \cdot 6.02_{(dB)} + 1.76_{dB}$$

Two options are available to get more bits: lower the PWM frequency, or increase the number of PWM in use. Due to Nyquist theory, the PWM frequency must be at least twice the highest frequency of interest [6].

Furthermore, if the PWM frequency is in the audible range (less than 22 kHz) we will need to filter it heavily to not hear a high pitched squeal behind the sounds. This sets a hard floor for how many bits you can achieve with your Arduino without having to add a ton of extra circuitry.

Also AVR have two different kinds of PWM: the Fast PWM (Single slope) versus Phase Correct PWM (Dual Slope). With Fast PWM, the counter will increase to TOP, and then reset to zero, whereas Phase Correct PWM will reach TOP, and then count backwards to zero, where it will count up again. Phase Correct takes twice as long to complete a cycle, so it will only go half as fast for any given bit depth but is much higher fidelity [4].

So let's build some code to summarize all of what has been said until now.

```
const int PWM1 = 11; //pin for the PWM output
const int PWM2 = 3; //pin for the PWM output
int val=0; //Variable used to store the value

void setup()
{
  cli(); //disable interrupts while registers are configured
  // Setup ADC to work with division factor of 16 and a clock speed of 1 MHz
  // we achieve a total of 58,617 samples per second 58 kHz
  bitSet(ADCSRA,ADPS2);
  bitClear(ADCSRA,ADPS1);
  bitClear(ADCSRA,ADPS0);
  DIDR0 = 0x01;
  // Analog output configuration
  bitSet(TCCR2A, WGM20); //Puts bit 0 of the TCCR2A register (Timer/Counter
                        //Control Register A), named WGM20 to 1
  bitClear(TCCR2A, WGM21); //Puts bit 1 of the TCCR2A register (Timer/Counter
                        //Control Register A), named WGM21 to 0
  bitClear(TCCR2A, WGM22); //Puts bit 2 of TCCR2A register (Timer/Counter
                        //Control Register A), named WGM22 to 1
  /*This sets Timer2 to PWM, Phase Correct mode
  Table of Timer/Counter Mode of Operation
  WGM22 WGM21 WGM20 Timer/Counter Mode of Operation
  0 0 0 Normal
  0 0 1 PWM, Phase Correct
  0 1 0 CTC - Clear Timer on Compare Match (CTC) Mode
  0 1 1 Fast PWM
  1 0 0 Reserved
  1 0 1 PWM, Phase Correct
  1 1 0 Reserved
  1 1 1 Fast PWM
  */
  bitSet(TCCR2B, CS20); //Sets to 1 bit CS20 (bit 0) of TCCR2B register
                        // (Timer/Counter Control Register B)
  bitClear(TCCR2B, CS21); //Sets to 0 bit CS21 (bit 1) of TCCR2B register
                        // (Timer/Counter Control Register B)
  bitClear(TCCR2B, CS22); //Sets to 0 bit CS22 (bit 2) of TCCR2B register
                        // (Timer/Counter Control Register B)
  sei(); //enable interrupts now that registers have been set
  pinMode (PWM1,OUTPUT);
  pinMode (PWM2,OUTPUT);
}
```

```
Serial.begin(57600);
}
void loop()
{
  val=analogRead(0); //read value of sound in at analogRead on pin 0
  Serial.println(val);
  analogWrite(PWM1,map(val,400,600,0,255));
  analogWrite(PWM2,map(val,400,600,0,255));
}
```

In the code we set the ADC to work with division factor of 16 and a clock speed of 1 MHz that gives 58.617 samples per second and we set the PWM for phase correct at 31.250 kHz with 2 pins with 2 resistors for hardware mixing of an upper and lower value: Figure 5 shows the circuitry hardware setup.

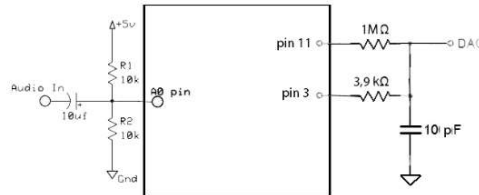


Fig. 7 - Circuitry hardware Setup [4]

IV. MATLAB INTERFACE WITH ARDUINO

For installing the Arduino software package in MATLAB, just type the command *supportPackageInstaller* this starts the Support Package Installer and select the web installation. Finally choose the Arduino package and it should have installed your Arduino package support.

In order to check if everything it's ok we are going first to develop a data logger. In MATLAB, there are several functions that are related to Serial communication. First, we need to target a Serial COM Port. Replace COM31' with whatever port your Arduino is connected to and set the communication speed.

```
s = serial('COM31');
```

```
s.BaudRate=115200;
```

Next, we need to open the Serial port.

```
fopen(s);
```

Then, to read from it and store to var data, we write:

```
dat = fscanf(s);
dado = str2double(dat);
```

The first reads the data and the second converts it from char to double.

After finishing with the Serial COM port, it is very important remember to close it else other applications cannot access it.

```
fclose(s);
```

So let's put all together and check if our ADC is sampling well.

```
%**CLOSE PLOT TO END SESSION
clear
clc
%User Defined Properties
plotTitle = 'Serial Data Log'; % plot title
xlabel = 'Elapsed Time (s)'; % x-axis label
ylabel = 'Data'; % y-axis label
plotGrid = 'on'; % 'off' to turn off
grid
min = 0; % set y-min
max = 1100; % set y-max
scrollWidth = 10; % display period in
plot, plot entire data log if <= 0
delay = .0001; % make sure
sample faster than resolution
%Define Function Variables
time = 0;
data = 0;
count = 0;
%Set up Plot
plotGraph = plot(time,data,'-mo',...
    'LineWidth',1,...
    'MarkerEdgeColor','k',...
    'MarkerFaceColor',[.49 1 .63],...
    'MarkerSize',2);
title(plotTitle,'FontSize',25);
xlabel(xLabel,'FontSize',15);
ylabel(yLabel,'FontSize',15);
axis([0 10 min max]);
grid(plotGrid);
%Open Serial COM Port
s = serial('COM31');
s.BaudRate=115200;
%define baud rate
disp('Close Plot to End Session');
fopen(s);
tic
while ishandle(plotGraph) %Loop when Plot is Active
    dat = fscanf(s); %Read Data from Serial as
    Float
    dado = str2double(dat);
    if(~isempty(dado) && isfloat(dado)) %Make sure
    Data Type is Correct
        count = count + 1;
        time(count) = toc; %Extract Elapsed Time
        data(count) = dado(1); %Extract 1st Data
        Element
        %Set Axis according to Scroll Width
        if(scrollWidth > 0)
            set(plotGraph,'XData',time(time >
            time(count)-scrollWidth),'YData',data(time >
            time(count)-scrollWidth));
            axis([time(count)-scrollWidth time(count)
            min max]);
        else
            set(plotGraph,'XData',time,'YData',data);
            axis([0 time(count) min max]);
        end
        %Allow MATLAB to Update Plot
        pause(delay);
    end
end
%Close Serial COM Port and Delete useless Variables
```

```
fclose(s);
clear count dat delay max min plotGraph plotGrid
plotTitle s ...
scrollWidth serialPort xlabel ylabel;
disp('Session Terminated...');
```

If there are now news you should see on MATLAB a graph appearing like the one on Figure 8 (on second 72 we disconnect A0 from 3.3 volts and connected it to GND).

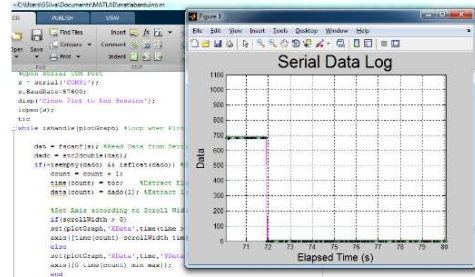


Fig. 8 - Serial Data Log In MATLAB

We will notice that if everything it's okay then when connecting A0 port from the Arduino to 5 or 3.3 or even 0 Volts the values will appear on logger and should have precise values. If we change the *scrollWidth* parameter from 10 to 0 the graph will display the entire log history instead of just a section. After closing the plot, the data log is available by accessing the data variable in the workspace.

V. OTHER MATLAB SOUND APPLICATIONS

A. Vocoder Application

A Vocoder is by definition a sound effect that can make a human voice sound synthetic [7,8] so, one of the purposes is to replace the carrier sound with another carrier from a different source. A main goal is obtained: it changes how the original sound sounds but keeps the original message.

The next example shows a small sample of our Vocoder application.

To create the vocoder a Matlab function Start Vocoder App is call as can be seen on the block diagram that illustrates de operation of the vocoder. After executing the start function a wave file is call and transformed into several parameters namely x-sampled data, f-sampling rate and b-number of bits. The first two are sent to the vocoder function where we simulate a transmission channel, with transmission and reception, after the signal is recover, from the parameters sent and, after reconstruction, the signal is playback.

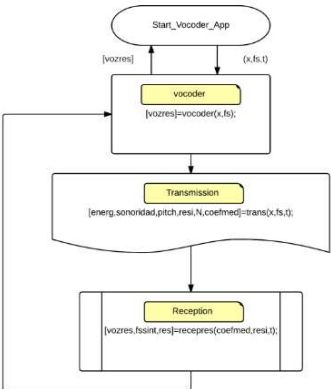


Fig. 9 – Block diagram of Vocoder Application

After playing the reconstruct signal is possible to execute some measurements to evaluate the quality of the reconstruct signal. Figure 10 shows the original signal versus the synthesized and the error.

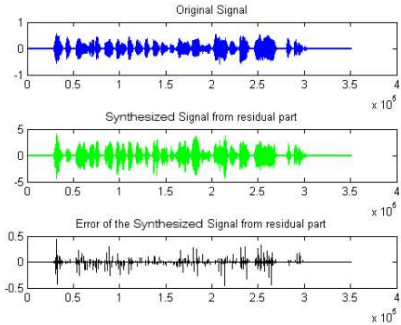


Fig. 10 - Original versus synthesized and error signal

From the figure analyses it's clear that the error from the synthesized signal from the residual part is very small, figure 11 show the overlap of the 3 signals showing this fact.

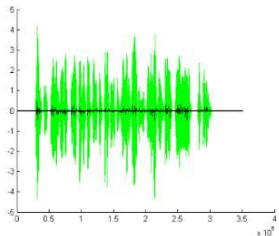


Fig. 11 - Overlapped of original, synthesized and error signals

The Matlab the code for signal reconstruction is:

```
Function [vozres,fssint,res]=recepres(coefmed,resi,t);
N=length(coefmed);
l=length(resi);
vozres=[];
c=[];
residu=[];
res=[];

for i=1:N
    %%ASSIGNED EACH LINE OF COEFFICIENTS(H) TO A VECTOR
    for j=1:l
        c(j)=coefmed(i,j);
    end
    %%ASSIGNED EACH LINE OF RESIDUAL(H) TO A VECTOR
    for s=1:l
        residu(s)=resi(i,s);
    end
    %%REVERSE FILTER FROM CODIFICATION
    res=[res residu];
    v=filter(1,c,residu);
    vozres=[vozres v];
end
fssint=(1*1000)/t;
wavwrite(vozres,fssint,'vozres.wav');
```

At the end of the code the play frequency is calculated and the residual reconstructed signal wav is created.

B. FM Synthesis of Instrument Sounds

Frequency modulation can be used to make interesting sounds that mimic musical instruments, such as bells, woodwinds or drums.

Our Bell application allows the user to control the sound of bells by touching different parts from the control system. To show the potential of the Arduino-Matlab connection the control system is attached to the Arduino and consists into 12 wires connected to the Arduino allowing music performance in one octave.



Fig. 12 – Piano Paper Octave with wires to arduino

The implementation code from the matlab size is:

```
dur=6;
tau=2;
Io=10;
fc=110;
fm=220;
Fa=8000;
```



```

samples=[0:1/Fa:dur];
tempoexecuciao=300; % 5 minutes 300 seconds
%Open Serial COM Port
s = serial('COM36');
s.BaudRate=9600;
disp('0 to end session or wait 5 minutes');
fopen(s);
s.ReadAsyncMode = 'continuous';
t=0;
tic
dat=4;
while t<tempoexecuciao
    if (s.BytesAvailable>0)
        dat = fscanf(s); %Read Data from Serial as Float
        temp = str2double(dat);
        if(temp>0&&temp<8)
            tau=temp;
        end
        if(temp>=8 && temp<20)
            Io=temp+2;
        end
        t=toc;
        if(temp==0)
            t=301;
            break
        end
    end
    for i=1:size(samples)
        A=exp(0-samples(i)/tau);
        I=Io*exp(-samples(i)/tau);
        onda(i)=A*cos(2*pi*f_c*samples(i)+I*cos(2*pi*f_m*samples(i)-
            pi/2)-pi/2);
    end
    sound(onda,Fa);
end
end

```

It's also possible a different operation mode where by touching the different wires the user controls different parameters creating different sounds. The general equation for an FM sound synthesizer is [9].

$$x(t) = A(t) \cos(2\pi f_c t + I(t) \cos(2\pi f_m t + \varphi_m) + \varphi_c)$$

Where $A(t)$ is a function of time called envelope, f_c is the carrier frequency, f_m is the modulation frequency, and technically $I(t)$ is called the modulation index envelope.

Table 4 – Different case values for the bell sound[9]

Case	f_c Hz	f_m Hz	$I(0)$	T sec	T_{dur} sec	F_s
1	110	220	10	2	6	11,025
2	220	440	5	2	6	11,025
3	110	220	10	12	3	11,025
4	110	220	10	0.3	3	11,025
5	250	350	5	2	5	11,025
6	250	350	3	1	5	11,025

We will demonstrate this application and others during the conference.

VI. CONCLUSION

This paper present an approach to Digital Sound Processing using the Arduino platform and MATLAB, with

some examples we show that the Arduino can be use together with Matlab to achieve the connection between physical and computation worlds. The AVR's registry analysis gave us the insight to how we should control our Arduino in order to achieve the best performance. An example of MATLAB and Arduino shows how we can proceed with better sound analysis since the choose Arduino limitations. One should here remember that there is also the Arduino Due that is around 15 times faster than the Arduino Uno used and has 2 DAC ports besides the normal PWM ones. Furthermore it has a 12 bits ADC instead of the current 10 bits of the Uno, unfortunately it doubles the Arduino Uno platform price. Nevertheless this price is still much less than the traditional DSP platforms used in Sound processing.

VII. FUTURE WORK

The Arduino platform can be use together with Matlab for real time sound processing, but some cares should be taking into account when using it, because the sampling process introduces some noise.

The use of Interrupt routines for the ADC sampling/PWM synthesis process and compare the performance improvements relatives to our approach is also a viable path.

ACKNOWLEDGMENT

This work is financed by the ERDF – European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project ref. FCOMP – 01 – 0124 – FEDER – 022701, and in the context of the projects PEst-OE/EEI/UI0127/2014 and Incentivo/EEI/UI0127/2014.

REFERENCES

- [1] "Atmel ATmega48A/48PA/88A/88PA/168A/328/328P datasheet," <http://www.atmel.com/devices/ATMEGA328P.aspx?tab=documents>, [Online; accessed 1-Oct-2014].
- [2] A. André Jucovsky Bianchi - Real time digital audio processing using Arduino <http://www.ime.usp.br/~ajb/artigos/article-smc2013-ajb-mqz.pdf> [Online; accessed 1-Oct-2014]
- [3] "AVR120: Characterization and Calibration of the ADC on an AVR – Application Note, 2006." <http://www.atmel.com/images/doc2559.pdf>, [Online; accessed 12-Oct-2014]
- [4] Dual PWM Circuits: Online Article from Open Music Labs <http://www.openmusiclabs.com/learning/digital-pwm-dac/dual-pwm-circuits/>, [Online; accessed 1-Oct-2014].
- [5] M. Nawrath, "Arduino realtime sound processing," <http://interface.khm.de/index.php/lab/experiments/arduino-realtime-sound-processing/>, [Online; accessed 12-Jun-2014]
- [6] Alan. V. Oppenheim, Ronald. W. Schaffer, "Digital Signal Processing", Prentice Hall, 1975.
- [7] L. R. Rabiner, Ronald. W. Schaffer, "Digital Processing of Speech Signals", Prentice Hall, 1978.
- [8] Achim Settlemier – Online Article - What is a Vocoder, http://www.srlab.de/linux/descr_vocoder.html [Online; accessed 5-Oct-2014]
- [9] James. H. McClellan, Ronald. W. Schaffer, Mark A. Yoder, "DSP FIRST A Multimedia Approach", Prentice Hall, 1998.

A dynamic programming algorithm to select optimal high-priority voice segments using Arduino

Sérgio Silva^{1,3}, Salviano Soares^{1,2}, Manuel Cabral^{1,2}

¹School of Sciences and Technology, UTAD, Vila-Real

²IEETA UA Campus, Aveiro; ³INESC-TEC, Porto
Portugal

sergio.s.silva@inescporto.pt; salblues@utad.pt;
mcabral@utad.pt

Filipe Neves^{4,5}, Pedro A. Assuncao^{4,5}

⁴Instituto de Telecomunicações

⁵Instituto Politécnico de Leiria, ESTG
Leiria, Portugal

fneves@ipleiria.pt; paassunc@ieee.org

Abstract—Prioritization of voice packets is useful to cope with packet losses in VoIP services by providing a mechanism to priority-enabled networks for dropping first the least important content, in order to reconstruct the best quality signal for user perception. A contribution to an efficient solution for voice packet prioritization is proposed in this paper, using an Arduino platform to implement a dynamic programming algorithm that is capable of finding the most important voice segments to convey in the payload of high-priority VoIP packets. Despite the fairly high complexity of the algorithm versus the limited computational resources of Arduino, the efficient implementation developed in this work demonstrates that the Mean Opinion Score (MOS) of voice signals is consistently improved when losses are primarily affecting the least important segments. The paper also discusses technological aspects relate to the choice of the Arduino DUE platform rather than UNO, MEGA, ZERO or 101 boards.

Keywords—voice communications; packet prioritisation; VoIP, MOS LQO; PESQ; Arduino.

I. INTRODUCTION

The prioritisation of voice packets according to their importance for the perceptual quality permits to increase the probability of successful reception for the most important voice segments over transmission channels that are prone to packet losses [1]. For example, it has been shown that by protecting 10% of the most important packets by a simple form of forward error correction, e.g. packet repetition on noisy channels, yields the same performance as that provided by randomly protecting twice as many packets [2]. Another approach is to give higher priority to packets that are more delayed than others, which also results in improved performance [3]. In a communication system where it is possible to distinguish between high and low priority packets, the low priority ones are always the first to be discarded (in case of network congestion, for example). As a result, better voice quality is expected to be achieved in comparison with the case where no packet differentiation is used. Therefore an efficient solution for the problem of selecting the best voice segments to transport in the payload of high priority packets is crucial for improving the signal quality and intelligibility when errors or erasures (packet losses in known positions) occur across the transmission network.

Different approaches have been used to differentiate voice packets per category or flow and to give them more priority in networks when contention with other packet types occur. To implement priority-based quality of service (QoS), some solutions distinguish between classes of flows as a per-queue

and as a per-user basis, while others are based on packet lengths [4] for instance. However none of them takes into account the specific contribution that each packet provides to the intelligibility in order to distinguish packets according to their actual importance.

Therefore an alternative approach is to prioritise voice packets according to the relative importance of each voice segment (*i.e.*, packet payload) in the overall quality of experience (QoE) when considering the sequence with the previous ones. This allows implementation of a selective packet dropping mechanism whenever network constraints do not guarantee reliable packet delivery. It is accomplished through packet prioritisation in order to allow priority-enabled networks to discard first the least important packets, *i.e.*, those packets labeled as low-priority. As a consequence, the lost packets have smaller impact on the speech intelligibility and QoE is improved [5], because the signal reconstruction is more effective by using the most important packets.

This paper follows this last approach by proposing a dynamic programming (DP) algorithm to find the optimal voice segments to be encapsulated in high-priority packets. Such high-priority packets comprise the best set of voice segments, which provide the highest quality in the worst case where all low priority packets are lost. A novel implementation, on the Arduino platform, and performance evaluation is presented for the DP algorithm. Despite the limited computational resources available in Arduino platforms and the fairly high computational complexity required by the DP algorithm to find the best set of high priority packets, this work demonstrates how a modest microprocessor board can be used to significantly enhance the voice quality transmitted over error prone networks.

The paper is structured as follows. Next section presents the DP problem to find the optimal set of voice packets and then Section III describes the Arduino platform focusing the hardware requirements. Then Section IV describes the algorithm implementation and Section V presents the performance evaluation and results. Finally Section VI concludes the paper.

II. PROBLEM FORMULATION

The problem of selecting the most important voice segments according to their relevance for the quality perceived by users is defined as a binary classification problem, where voice segments defined as packet payloads are assigned either low or high priority. For the problem formulation, the following definitions are useful:

- *Utterance*: a speech sample sequence of vocal sounds preceded and followed by silence.
- *Segment*: chunk of voice signal samples belonging to an utterance, i.e., one utterance is comprised of several segments.
- *Packet*: network layer data unit encapsulating one segment as payload.
- *Priority rate*: ratio between the number of high priority packets and the total number of packets comprising an utterance.

We denote by n the total number of segments in an utterance U and by m the number of high priority segments in a subset M of U . The problem of optimal segment classification is formulated as follows: *Given an utterance of n segments, corresponding to n segments (packets), and a maximum priority rate, $R(M)$, the problem is to find the best $m < n$ segments that minimize some distortion function, $D(m, n)$, between the original utterance and the one reconstructed from the reduced subset M , of m segments.*

The need to determine the best segments that minimise the distortion at the lowest computational expenses, on the one hand, and the possibility to formulate this problem as a rate-distortion problem, on the other hand, permits to use a DP algorithm to solve it [6]. To solve the problem, the algorithm first parses the entire set of segments, p_i of U ($i=0, \dots, n-1$) to identify a limited set of candidate segments to include in M . Then it makes the reverse pathway to choose the definite set of optimal segments. In the forward pathway, the first element of M is defined as the first segment in N , i.e., p_0 , and then other segments p_k of U are added as candidates for the same i^{th} position of M ($i=\{1, 2, \dots, m\}$; $(i-1) \leq k \leq (n-1-m+i)$). Each t defines a stage of the DP algorithm and the cardinality of the intermediate subset, herein designated by a t -subset. At each stage t , k defines a segment p_k to be added to a $(t-1)$ -subset, thus the last one of the t -subset, herein represented by p_t^k . Generally, for each p_t^k , several $(t-1)$ -subsets are possible to identify, to which correspond a different combination of segment indices $\{l_0, l_1, \dots, l_{t-2}\}$ in U , and a distortion value for the whole utterance (size n) reconstructed from each reduced of segments (size $t < n$). The minimum distortion achieved by each t -subset ending at segment k in the stage t , is defined as a distortion state D_t^k , defined as follows

$$D_t^k = \min_{l_1, l_2, \dots, l_{t-2}} \left\{ \sum_{j=0}^{k-1} d(p_j, p_{l_j}) + \sum_{j=k}^{n-1} d(p_j, p_k) \right\} : i = \max\{0, l_1, l_{t-2}\} \quad (1)$$

Since p_t^k is the last packet in a t -subset, the distortion caused by packets beyond p_t^k has the constant form $\sum_{j=t}^{n-1} d(p_j, p_t)$ and Eq. 1 may be rewritten as:

$$D_t^k = \min_{l_1, l_2, \dots, l_{t-2}} \left\{ \sum_{j=0}^{k-1} d(p_j, p_{l_j}) + \sum_{j=k}^{n-1} d(p_j, p_k) \right\} \quad (2)$$

which is equivalent to

$$D_t^k = \min_{l_1, l_2, \dots, l_{t-2}} \left\{ \sum_{j=0}^{k-1} d(p_j, p_{l_j}) + \sum_{j=k}^{n-1} d(p_j, p_{l_j}) - \sum_{j=k}^{n-1} d(p_j, p_{l_{t-2}}) + \sum_{j=k}^{n-1} d(p_j, p_k) \right\}$$

by adding and subtracting the same value $\sum_{j=k}^{n-1} d(p_j, p_i)$. Since in $\sum_{j=k}^{n-1} d(p_j, p_i)$ all i are $i < k$, then $\sum_{j=k}^{n-1} d(p_j, p_i) = \sum_{j=k}^{n-1} d(p_j, p_{l_{t-2}})$; and Eq. 3 can be

rewritten as

$$D_t^k = \min_{l_1, l_2, \dots, l_{t-2}} \left\{ \sum_{j=0}^{n-1} d(p_j, p_i) - \underbrace{\sum_{j=k}^{n-1} [d(p_j, p_{l_{t-2}}) - d(p_j, p_k)]}_{e^{l_{t-2}k}} \right\} : i = \max\{0, l_1, l_{t-2}\} \wedge i \leq j \quad (4)$$

Therefore, if we take into account that the first term of the second sum of Eq. 4 corresponds to subtract the distortions caused by missing segments above or equal to p_k when $p_{l_{t-2}}$ is the last one in the subset and the second term of this sum corresponds to add the distortions caused by missing segments above or equal to p_k , when p_k is the last segment in the subset, Eq. 4 can be rewritten as:

$$D_t^k = \min_{l_{t-2}} \left\{ \min_{l_1, l_2, \dots, l_{t-3}} \left\{ \sum_{j=0}^{n-1} d(p_j, p_i) \right\} - e^{l_{t-2}k} \right\} : i = \max\{0, l_1, l_{t-2}\} \wedge i \leq j \quad (5)$$

$$= \min_{l_{t-2}} \left\{ D_{t-1}^{l_{t-2}} - e^{l_{t-2}k} \right\},$$

which expresses the recursion formula of the DP algorithm. Then, it is possible to derive the expression that should be used to derive the indices $\{l_0, l_1, \dots, l_{t-2}\}$ of the optimal segments to include in U , as follows,

$$l_t = \begin{cases} \arg \min_k \{D_t^k\}, & t = m-1 \\ \arg \min_l \{D_{t+1}^l - e^{l, l_{t+1}}\}, & t \in [1, m-2] \\ 0, & t = 0 \end{cases} \quad (6)$$

Once the indices $\{l_0, l_1, \dots, l_{t-2}\}$ are found, the optimal subset M of segments to be classified as the payload of high priority segments is fully determined.

III. THE ARDUINO PLATFORM

To implement the DP algorithm described above in an Arduino platform, it is necessary to ensure the required amount of memory and processing power to guarantee successful operation. From previous work [7] based on the Arduino UNO, using an Atmel ATmega328P (16MHz 8-bit

RISC microcontroller) [12] the limitation imposed by its 32kB Flash memory prevents its use for this implementation because the program and data memory is not sufficient. The following table provides a general view of the Arduino platforms technical specifications:

TABLE I. ARDUINO TECHNICAL SPECIFICATIONS

Platform	Microcontroller	Clock	Flash Memory	SRAM
Uno	Atmega 328	16 MHz	32 kB	2 kB
Mega	Atmega 2560	16 MHz	256 kB	8 kB
101	Intel Curie	32 MHz	196 kB	24 kB
MKR1000	SAMD21 Cortex-M0+	48 MHz	256 kB	32 kB
Zero	ARM ATSAM21G18	48 MHz	256 kB	32 kB
Due	ARM AT91SAM3X8E	84 MHz	512 kB	96 kB

Comparing the specifications, one can observe that UNO is the one with less processing speed and memory resources, respectively clock speed, Flash Memory and Static Random Access Memory (SRAM). Flash memory is used to store the program image and initialization data. It is possible to execute program code from the Flash, but its stored data cannot be modified by the executing code. To modify the data, it must be first copied into SRAM, which can be read and written by the executing program. Usually a running program uses the SRAM for three purposes:

1. **The Static Data** - all the global and static variables have a block of reserved space in SRAM. For variables with initial values, the runtime system copies the initial value from the Flash whenever the program starts.
2. **The Heap** - The heap is used for dynamically allocated data items. The heap grows from the top of the static data area upwards as data items are allocated.
3. **The Stack** - The stack is used for local variables and for maintaining a record of interrupts and function calls. The stack grows from the top of memory down towards the heap. Every interrupt, function call and/or local variable allocation causes the stack to grow. Returning from an interrupt or function call will reclaim all stack space used by that interrupt or function. The following image shows the SRAM allocation directions.

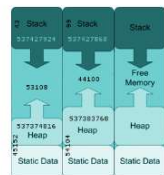


Fig. 1. Dynamic allocation of SRAM

The Static Data, located at the bottom of the SRAM (Fig. 1) is allocated during the program startup and remains stable throughout the program execution. The remaining SRAM is disputed between the Stack and the Heap, both growing and shrinking during program execution. If they collide they can cause either immediate crash or just data corruption: either way the results are unpredictable.

To choose one particular Arduino platform among the various available options it is necessary to analyse the requirements of Flash and SRAM. For Static Data and Heap the algorithm uses over 54kB in the worst case and 45kB in the best case. For the Stack the system uses 99kB in the worst case and 43kB in the best case. From these numbers, we can easily see that the only platform that can support the algorithm is the Arduino DUE, because even in the best case we need 45 kB of SRAM and the Arduino Zero for example only has 32 kB. To calculate the amount of memory allocated we have used the following Pseudocode:

```

PSEUDOCODE 1: FreeRams()
uint32_t stackTopPosition;
uint32_t heapTopPosition;
// Current position of the stack.
// Current position of heap.
Set heapTopPosition to hTop as uint32_t
Call method free with hTop
Call method SerialUSB.print with stackTopPosition
Call method SerialUSB.print with ","
Call method SerialUSB.print with heapTopPosition
Call method SerialUSB.print with ","
The difference is the free, available sram.

```

The program gets the memory address from the top position for the Heap and for the Stack and the difference from these two addresses gives the available memory. On the other hand, from the ATME1 32 bit Cortex M3 SAM3X Microcontroller datasheet [8] we can see that the memory addresses end at 0x20087FFF (537427967) and start at 0x20070000 (537329664) so by subtracting the heapTopPosition to the start address we obtain the total amount of memory used by the Heap plus the Static Data. The same procedure can be done to calculate amount of memory used for the Stack, but in this case the end address must be used. In terms of Flash memory, the algorithm requires 53 kB, therefore making impossible to run in the Arduino UNO platform even without limitations of SRAM.

In terms of the algorithm speed, there are three major functions to consider: the signal loading/power calculation; the DP classifier function that is responsible for the optimal classification and packetization overhead. The next graphic shows the processing timing required for each of the three functions, in milliseconds (mSec).

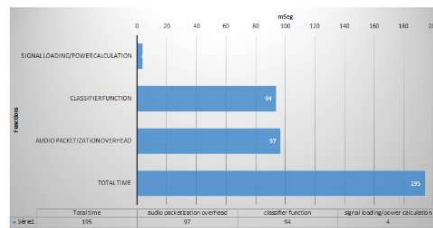


Fig. 2. Processing time

From the graphic analyses, we can see that between loading, analyzing, processing and playing the sound there is a gap of 195 mSec making the system capable of real-time processing the audio samples.

Besides the implementation of the DP algorithm, our system also includes an erasure packets loss simulator and subsequent signal reconstruction for testing and performance evaluation. A priority loss model where all low-priority packets are lost, a random burst loss model and an interleave model where every other packet is lost. The user can choose any of these network loss models and the type of signal reconstruction may also be specified as zero-order or first-order interpolation.

IV. ALGORITHM IMPLEMENTATION

A. The Arduino code implementation

The Arduino code implementing the DP algorithm can be divided in four parts. The first part is the function 'LoadSignal(int n_original_segments, int n_samples)', used to recursively break the utterance into segments and at the same time creating a list with the RMSE for each segment in order to use it later in the segmentation process.

PSEUDOCODE 2: Load Signal (N° Original Segments, N° Samples)

```
Initialise a to 0
for i is 0, i is less than n_original_segments, i increments by 1
  for j is 0, j is less than n_samples_segment, j increments by 1
    Set i of frames_v to
    Set i of RMSE to position i in RMSE plus position i in
    frames_v
    a++;
  endfor
  Set i of RMSE to sqrt with position i in RMSE divided by
  n_samples_segment
  Call method RMSEList.add with position i in RMSE
endfor
```

Next the voice segments samples are classified using function 'segmentstoloseose(int n_samples_loose)'. As an input argument, this function accepts the number of segments to classify as low-priority in each utterance.

PSEUDOCODE 3: Segments to lose (n° samples lose)

```
Initialise listSize to RMSEList.size
for h is 0, h is less than listSize, h increments by 1
  Initialise val to RMSEList.get with h
  Initialise count to 0
  for j is 0, j is less than listSize, j increments by 1
    if (val < RMSEList.get(j))
      Count++;
    endif
  endfor
  If (count >= (120 - n_samples_loose))
    Call method RMSEListToLose.add with h
    Initialise aux to position h in RMSEListLostPlaces
    Set h of RMSEListLostPlaces to aux minus 1
  endif
endfor
for j is 0, j is less than RMSEListToLose.size, j increments by 1
  Call method Serial.print with RMSEListToLose.get/j
  Call method Serial.print with ", "
endfor
```

This function goes through the list of the RMSE values for the segments and compares them with each other, creating a list with the segments that are less important to the utterance, i.e., the low-priority segments.

The third part is the function that simulates the network erasure losses and reconstructs the degraded signal, after dropping all low-priority packets. In our code, we implement a state machine that allows choosing between the three network loss models: prioritized, burst or interleave. The user just

needs to choose the variable 'estado' to be 1, 2 or 3 respectively.

PSEUDOCODE 4: Degraded Signal (Orig. Segments, N° Samp, N° S Lose)

```
Call method randomSeed with analogRead 0
if estado is equal to 2
  Set randomNumber to random with 4, +1
endif
if estado is equal to 1
  Call method segmentstolose with n_samples_loose
endif
for i is 0, i is less than n_samples_loose, i increments by 1
  for j is 0, j is less than n_segment_samples, j increments by 1
    if estado is equal to 1
      Set j plus (RMSEListToLose.get with i
      multiplied by n_segment_samples) of buffer to 0
    endif
    if estado is equal to 2
      Set j plus (position i in SegmentosToLoseb
      multiplied by n_segment_samples) multiplied by randomNumber of buffer to 0
    endif
    if estado is equal to 3
      Set j plus (position i in SegmentosToLose
      multiplied by n_segment_samples) of buffer to 0
    endif
  endfor
endfor
```

For each state the function applies the respectively segments to lose list. This function just passes all the cells of the original maze and copies them to the output maze.

The last part is the for-loop function that is responsible for writing the output to a file.

PSEUDOCODE 5: Output function ()

```
for i is 0, i is less than numBytes i increments by 1
  Write position i of buffer to output
endfor
```

The rest of the code allows us to clear the list in order to start the process again.

V. PERFORMANCE EVALUATION

In order to evaluate the performance of the system, we defined a method to compare an original voice signal with its degraded version reconstructed from the smaller set of segments that result from packet loss. We have followed the Telecommunication Standardization Sector, ITU-T guidelines P.862.3, which define methods for objective quality measurement based on the Recommendations P.862, P.862.1 and P.862.2 for terminals and subjective and objective assessment methods [9].

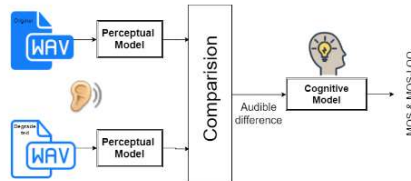


Fig. 3. The PESQ Model

The ITU-T provides the PESQ algorithm and C files necessities to carry the implementation of the Perceptual Evaluation of Speech Quality (PESQ) as a reference implementation for ITU-T Recommendations P.862, P.862.1

and P.862.2. Version 2.0 October 2005. The following command line is an example of how to use this tool:

PESQ +8000 or105.wav dg105.wav

The +8000 parameter indicates that the '.wav' files are sampled at 8000Hz. The **or105** is the *original* file and the **dg105** is the *degraded* '.wav' file. The Mean Opinion Score (MOS) and Mean Opinion Score Listening Quality Subjective (MOS-LQO) results are presented with a scale of 1 to 5 where:

TABLE II. ABSOLUTE CATEGORY RATING (ACR) TEST

Quality of the speech Score	Quality of the speech Score
Bad	1
Poor	2
Fair	3
Good	4
Excellent	5

The PESQ method produces objective MOS scores that are designed to reflect, as accurately as possible, subjective speech quality as perceived by human subjects participating in subjective testing, and the MOS is the average of all individual opinion scores resulting from the tests. The average MOS for each audio file is then calculated from the group of individual scores.

The chart in Figure 4, shows the raw scores from an actual Absolute Category Rating (ACR) Test, that resulted in MOS=2.4. The high number of votes for opinion scores '2' and '3' is consistent with the score of 2.4. However, a significant number of listeners did vote '1' and '4', therefore when we analyze these tests it is important to take into account their viability.

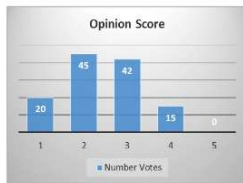


Fig. 4. Opinion Score for an ACR Test

Test labs typically use high quality audio recordings of phonetically balanced source text, such as the Harvard Sentences: these sentences comprise a set of English phrases chosen to ensure that the spoken text contains the range of sounds typically found in speech.

The recordings are obtained in quiet conditions using high resolution (16-bit) digital recording systems, adjusted to standardized signal levels and spectral characteristics. The International Telecommunications Union (ITU) and the Open Speech Repository provide sources of phonetically balanced speech material [10]. The following sentences represent the List 1 of the Harvard Sentences:

1. *The birch canoe slid on the smooth planks.*
2. *Glue the sheet to the dark blue background.*
3. *It's easy to tell the depth of a well.*
4. *These days a chicken leg is a rare dish.*
5. *Rice is often served in round bowls.*

6. *The juice of lemons makes fine punch.*
7. *The box was thrown beside the parked truck.*
8. *The hogs were fed chopped corn and garbage.*
9. *Four hours of steady work faced us.*
10. *Large size in stockings is hard to sell.*

To differentiate between listening and conversational scores, the ITU introduced the terms MOS-Listening Quality (MOS-LQ) and MOS-Conversational Quality (MOS-CQ) with the additional suffixes (S)ubjective, (O)bjective and (E)stimated [9]. Hence, a listening quality score from an ACR test is called a MOS-LQS.

P.862.x (PESQ) is a 'full reference' model designed to measure objective listening quality (MOS-LQO) that compares an undistorted reference file with a test file that may be distorted through encoding or network induced artifacts such as packet drops. P.862.x compares the input audio signals in the psychoacoustic domain, which requires the signals to be transformed using Fast Fourier Transform, proper level scaling and temporal alignment. Because the process is fairly computationally intensive and requires access to the undistorted reference signal, full reference model is mainly employed in a laboratory environment.

There are other algorithms like the VQmon [11] that is an advanced VOIP perceptual quality estimation algorithm that incorporates support for key international standards including ITU-T P.564, ITU-T G.107, ITU-T G.1020, ETSI TS 101 329-5 Annex E and IETF RFC 3611. Contrary to the PESQ, VQmon does not use the original reference signal being less computer demanding than the PESQ algorithm. Typically, VQmon has proven to be more reliable than other algorithms such as G.107 E-Model, which was originally developed within the European Telecommunications Standardization Institute (ETSI) as a transmission planning tool for telecommunication networks. The objective of the E-Model is to determine a transmission quality rating, *i.e.*, the R factor, that incorporates the 'mouth-to-ear' characteristics of a speech path. The E-Model is based on the incorrect premise that the effects of impairments are linear. Non-linear approaches have been found to be better suited to describe the relationship between key impairment factors. In particular, VQmon's non-linear impairment factor combination model has been shown to improve the accuracy of estimated MOS scores when there is a high level of several dissimilar impairments (for example, packet loss and echo).

Impairments caused by network congestion tend to be highly time-varying, with high packet loss "bursts" occurring intermittently between 'gaps' of low or no packet loss. VQmon outperforms the E-Model by incorporating on one hand the effects of time varying IP network impairments, providing a more accurate estimate of user opinion and on the other hand extensions to support wideband and super-wideband/fullband codecs.

The R factor is mapped to a subjective voice quality measure MOS using the following equations:

$$\begin{aligned} \text{MOS} &= 1 + 0.035R + R(R - 60)(100 - R)(7 \times 10^{-6}) \text{ for } R = 1 \text{ to } 100 \\ \text{MOS} &= 4.5 \text{ for } R \geq 100 \\ \text{MOS} &= 1 \text{ for } R \leq 0 \end{aligned}$$

A. Performance of packet classification and prioritization

To evaluate the performance of the DP classification algorithm we have used all Sentences from the List 1 of Harvard Sentences with a 16-bit resolution at 8 KHz. Figure 5 shows the PESQ results (MOS PESQ) obtained from the reconstructed signals after simulated erasures losses either using different packet priorities obtained by the DP classification algorithm, burst and interleaved.



Fig. 5. Voice quality of observed signals using the original as reference

The results show that prioritized signals lead to better reconstruction with greater MOS than those obtained from utterances randomly corrupted either from burst or interleaved. As expected the MOS values monotonically decrease with the percentage of lost packets. The results show that using prioritization in general achieves over 20% better performance than non-prioritized random loss either in bursts or interleaved.

The following expression was used to convert raw MOS into MOS-LQO:

$$y = 0.999 + \frac{4.999 - 0.999}{1 + e^{-1.4945x + 4.6607}}$$

Figure 6 shows the converted raw values of PESQ onto the MOS-LQO (Mean Opinion Score - Listening Quality Objective) where again the prioritized strategy outperforms all others methods.



Fig. 6. Voice quality of observed signals using the original as reference

VI. CONCLUSIONS AND FUTURE WORK

In this work, we have developed a DP algorithm to find the best set of voice segments that should be transported in high-priority packets over transmission networks. We have also discussed its implementation in Arduino platforms in terms of memory requirements and processing speed. The results demonstrate the effectiveness of the DP algorithm, which consistently leads to higher MOS by assigning packet priorities to packets according to the relevance of each one. In comparison with non-prioritized transmission and random losses the DP classification algorithm is capable of selecting the most important packets as high-priority and consequently delivering better quality to end-users. The Arduino implementation presented in this paper demonstrates that good performance can be achieved with modest computational resources.

ACKNOWLEDGMENT

This work is partially financed by the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia, Portugal.

REFERENCES

- [1] Jae-Yul Yoon, Hoonchong Park (2011) Improving the Speech Quality of VoIP by Packet Prioritization. *IEEE Signal Processing Letters* Vol. 18, No 12, pp. 725–728, Dec. 2011.
- [2] M. Petracca, A. Servetti and J. C. De Martin, "Voice transmission over 802.11 wireless networks using analysis-by-synthesis packet classification" *First International Symposium on Control, Communications and Signal Processing*, pp. 587-590, Hammamet, Tunisia, March 2004.
- [3] C. Olariu, J. Fitzpatrick, Y. Ghamri-Doudane and L. Murphy, "A delay-aware packet prioritisation mechanism for voice over IP in Wireless Mesh Networks," *IEEE Wireless Communications and Networking Conference*, pp. 1-7, Doha, Qatar 2016.
- [4] Stylianos Dimitriou and Vassilis Tsaoussidis, "Promoting effective service differentiation with Size-oriented Queue Management," *Computer Networks*, vol. 54, no. 18, pp. 3360–3372, Dec. 2010.
- [5] Jammeh E, Mkwawa I, Khan A, Goudarzi M, Sun L, Ifeakor E (2012) Quality of experience (QoE) driven adaptation scheme for voice/video over ip. *Telecommunication Systems*, Vol. 49, No. 1, pp. 99–111, Jan. 2012
- [6] T. Cormen, C. Leiserson, and R. E. M. P. . M.-H. B. C. Rivest, *Introduction to Algorithms*, ch. 16: Dynamic Programming, pp. 301–328. The MIT Press, ISBN: 0-262-03141-8, Cambridge, Massachusetts, 1990.
- [7] S. Silva, S. Soares, A. Valente and S. T. Marcelino, "Digital sound processing using arduino and MATLAB", *Science and Information Conference (SAI)*, pp. 1184-1191, London, July 2015.
- [8] ATMEL 32 bit Cortex M3 SAM3X Microcontroller datasheet, March 2015, http://www.atmel.com/Images/Atmel-11057-32-bit-Cortex-M3-Microcontroller-SAM3X-SAM3A_Datasheet.pdf, (accessed in Jan 2017).
- [9] ITU-T Recommendation P 862, "Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrowband telephone networks and speech codecs," February 2001
- [10] <http://www.cs.columbia.edu/~hgs/audio/harvard.html>, (accessed in Jan 2017).
- [11] ITU-T SG12 Contribution D105, VQmon Description, January 2003.
- [12] "Atmel ATmega48A/48P A/88A/88P A/168A/328/328P datasheet", [online] Available: <http://www.atmel.com/devices/ATMEGA328P.aspx?tab=documents>.

Morse code translator using the Arduino platform:

Crafting the future of microcontrollers

Sérgio Silva^(a,b,d), António Valente^(a,b), Salviano Soares^(a,c), M.J.C.S. Reis^(a,c), Jean Paiva^(a), Paulo Bartolomeu^(d)

a) School of Sciences and Technology, Engineering Department UTAD, Vila Real, Portugal

b) INESC TEC - INESC Technology and Science (formerly INESC Porto, UTAD pole), Porto, Portugal

c) IEETA - Institute of Electronics and Informatics Engineering of Aveiro, Aveiro, Portugal

d) Globaltronic - Electronics and Telecommunications - Águeda, Portugal

+351 932 343 343

sergio.s.silva@inescporto.pt

Abstract—The ability of the Arduino platform to enhance student interest and performance in science, technology, engineering, and mathematics (STEM) courses, while fostering skills that are important prerequisites for future IT careers, has been proven more than once in the past years. But can the future be crafted without the past? We believe that many past inventions crave the future, so their understanding is a bridge of knowledge that must be passed to students. According to Grand View Research web site the microcontroller market will rise from the 20 billion units in 2015 to an amazing 39 billion units in 2020. Therefore an increase on IT careers is also expected. The Morse code and the telegraph revolutionized long-distance communication in the past and laid the groundwork for the communications revolution. In fact, although developed in the 1830s and 1840s by Samuel Morse (1791-1872) and other inventors, only in 1844 the first telegraph message, from Washington, D.C., to Baltimore, Maryland, was sent. To provide the means to students to start learning this technology we have developed four experiences that introduce them to the fundamentals of communications, including the Li-Fi technology. This new technology is based on the Morse code, and can spark again the communications revolution by using tiny, imperceptibly flickering lights can provide a new way of sending data to computers and mobile devices. Therefore, we decided to revitalize the almost forgotten Morse code by implementing it with an Arduino in order to lay again the foundations to this new revolution that is coming. This paper presents the implementation model of two Morse code translators, how they work, their implementation, and some results. We also present a VLC (Visible Light Communication) system based on the same principles of the Morse code building this way the foundation for students to proceed with this course of investigation.

Keywords— Arduino; Morse Code; Visual Light Communication; VLC; STEM; Education; Communications; Microcontroller Applications

I. INTRODUCTION

“What hath God Wrought” (meaning “What God has Done”) was the first text message transmitted by the telegraph in 1844.

w h a t h a t h G o d W r o u g h t

... ..

Morse invention revolutionized long-distance communication. The telegraph messages were a series of dots and dashes, that Morse grouped to form letters and numbers as can be seen in Figure 1. For example, to form the letter S we should join 3 dots (...), follow by a brief silence.

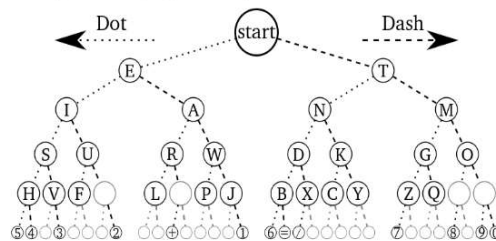


Fig.1 - Morse code tree

This revolutionary idea allows long distance communication between people connecting humanity and building bridges between nations.

By the end of the 19th century, however, new technologies began to emerge, many of them based on the same principles first developed for the telegraph system.

In time, these new technologies would overshadow the telegraph, which would fall out of regular widespread usage, but as in the past, today the same principles developed by the telegraph can craft again the future so researchers think with the rise of Li-Fi and VLC.

One of the most interesting things about Morse code is the resemblance to binary encoding, has both used two different symbols in their representation.

In fact Morse code offers a slow but reliable means of transmitting and receiving wireless text messages through conditions involving noise, fading, or interference. This is primarily because it is simple binary code (key down or key up) allowing for an extremely narrow bandwidth.

Other resemblances between binary and Morse code is the binary tree from figure 1 where each branch is always divided into two new branches. The efficiency of Morse encoding style led to data compression schemes like the one created by David Huffman in 1952, using the same algorithm of Morse trees to construct the base of all modern compression schemes. Figure 2 shows the Huffman tree.

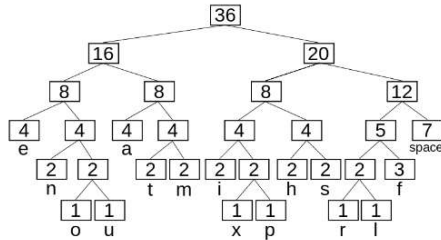


Fig.2 - Huffman tree [9]

The Huffman coding is indeed the best possible compression of the data (of course, one can do better compression in practice using additional properties of the data and careful hacking). The output from Huffman's algorithm can be viewed as a variable-length code table for encoding a source symbol (such as a character in a file) [10] like in the Morse code, but instead of using binary numbers Morse uses different lengths on the coding.

II. IMPLEMENTING THE MORSE CODE TRANSLATOR

There are many different ways to implement the dichotomic search table or Morse tree of figure 1. If you look closely you can see that there are really three possibilities at each node: go right, go left, or stop. So to decode a Morse transmission we need to remember that the Morse code has only two kinds of signals, dashes and dots. So if we start on the beginning of the Morse tree and the first signal is a dot we go left to letter 'E' if the next signal is a dot we go left again, if it is a dash we go right and if it is a pause we got the letter.

There is also some ground rules considering pauses and other rules that we need to take into account. A pause for a dot time is between different signals for the same letter, a pause for a dash worth time is between different letters in the same word, a dash is three times the length of a dot and between words we get a seventodot time pause (this represent the space character).

To store the Morse tree in our implementation we have used an array of chars. The array has 127 positions and the pointer starts pointing to its middle (position 64), that in our case is a space for convenience.

Figure 3 shows the middle 65 characters used to implement the Morse tree from figure 1.

E**L**R*+, ****A***P@**W***I* *6-B*=-D*/X***N***C;*(K*)(Y***T*

Fig.3 - Morse array

The arrow represents the pointer that will move to the left if we receive a dot or to the right if we receive a dash.

At every receive signal dot or dash the pointer will jump to the next level and as there are 5 levels, on the tree, the jump will be equivalent of $2^5 = 32$ positions in the array.

This also means that now there are only 4 levels remaining so the next jump will be of $2^4 = 16$ positions, the next jump will be of $2^3 = 8$ positions, and so on, as can be seen in figure 3, where we have received a dash and a dot followed by a space.

E**L**R*+, ****A***P@**W***I* *6-B*=-D*/X***N***C;*(K*)(Y***T*

E**L**R*+, ****A***P@**W***I* *6-B*=-D*/X***N***C;*(K*)(Y***T*

E**L**R*+, ****A***P@**W***I* *6-B*=-D*/X***N***C;*(K*)(Y***T*

Result= 'N'

Fig.1 - Received ' - . ', then 'N' was produced

Each letter is composed from 1 up to 5 levels, or from 1 to 5 symbols, meaning that if we receive 6 symbols there is an error in the received signal. All positions on the Morse array represented by '*' are also error positions.

At the end of each letter the pointer goes back to the middle of the array, the 64th position.

III. MORSE CODE SPEED AND LETTER TIMING

A. The decoding of the receive signals

To decode the received signal we need to take into account the speed of the transmitted signal, which in the case of the Morse code is usually stated in a nonstandard fashion, meaning that instead of defining the dot duration, it is usually stated the amount of words per minute. So if someone says an operator can transmit 20 words per minute (20 wpm), it means that the average dot equals 50 milliseconds.

The way this value is obtained is by stating that the typical word is a 5 letter word (like the word CODEX or PARIS) and that an operator that has a 20 wpm transmission rate is able to transmit the word CODEX 20 times in one minute.

To further explain this topic, we have constructed table 1 in order to show the timing calculations for each letter of the referred word considering 50 milliseconds duration time for the dot and following the Morse standard rules: a dash is 3 times the length of the dot, so 150 milliseconds; the pause between different signals of the same letter is of the same size of the dot time, and if the pause is for a dash time, it is between different letters in the same word, also pauses equivalent to 7 dots (350 milliseconds), separate different words. Table 1 shows the standard calculation times, showing also the meaning of 20 wpm.

TABLE I. STANDARD WORD TIME CALCULATIONS

Letters	Morse Code	Millisec.
Cc	-.-.	550
		150
o	---	550
		150
d	-..	350
		150
e	.	50
		150
x	-...-	550
		150
	Word time	2650
	Space words	350
	Total time	3000
	Words/minute	20 words

Generally speaking, that is a quite slow transmission rate, if for example, we consider the ASCII code (American Standard Code for Information Interchange), an 8-bit code with 256 (2^8) characters. For transmitting the word CODEX we will need 5x8 bits. Therefore, to transmit 20 times this word it will take $5 \times 8 \times 20 = 800$ bits plus a start and a stop bit. If we consider that serial communication between a PC and any other device can achieve, without compression, 115200 bits/sec then to transmit our 20 words, it will take about 6.944 milliseconds, i.e., 432 times faster than the human Morse operator. Even if we speed up transmission of the dot up to 1 millisecond, the total time to transmit the word will be 60 milliseconds, i.e., 8.94 times slower than the current standard transmission rate. Of course, no human could receive this transmission, but a computer can, in fact, even with low quality speakers and microphones a computer can write and receive Morse code up to 300 words per minute, that is equivalent to 17.7 milliseconds per word.

The main reason for this limit is that normal speakers and microphones are tuned to frequencies up to 15 kHz. It is possible to achieve much higher transmission rates by using ultrasound microphones and speakers.

However, one important thing to remember is that although the Morse code signal needs to be converted from electromagnetic waves to sound for human understanding. If

microcontrollers are used, this conversion is not necessary, making possible to transmit at an even higher frequency/rate, in the order of 10^6 , or even faster, depending on the microcontroller receiving ADC.

B. The Hardware implementation

The hardware implementation is based on the Arduino UNO platform equipped with an Atmel ATmega328P (a 16 MHz 8 bit RISC microcontroller), a Morse key, and some connection wires. The simplicity of the design is used to encourage students to build and program the system.

Figure 4 presents the hardware implementation of the encoder, also showing a metronome that serves as a timing basis to users that have no Morse code practice.



Fig.4 - Hardware implementation

The electric equivalent circuit is shown on Figure 5. There are two buttons, one for the dot signal and one for the dash. S1 represents the dot and S2 the dash. The LED will blink at the same frequency of the dash or dot, making Morse visible besides audible. The electric circuit was drawn using the IDE Fritzing.

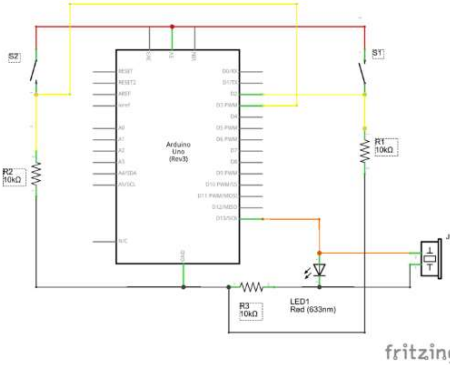


Fig.5 - Electric equivalent schematic

C. The Matlab implementation

After receiving the signal the Arduino sends it to Matlab using the USB serial port that converts the code into human voice allowing hearing what has written in Morse code.

The Matlab implementation is divided in three parts. The first part is the function 'plotgraph_SerialPort'. This function is responsible for the reception of the Morse code signal sent by the Arduino.

While receiving data, this function calls the decoder in order to decode the encrypted Morse code into letters and words. It also calls the function 'plot' to show, in real time, what the system is receiving, as can be seen on the graph in the left-upper corner of the screenshot presented in figure 6.



Fig.6 - Matlabbmorse decoder

When the end of the Morse message is reached the system opens a new window, like the one in figure 7, showing the received Morse signal and the signal spectrogram.

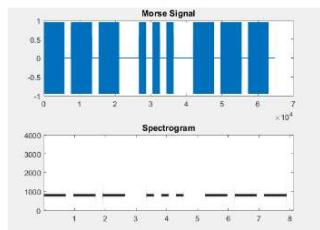


Fig.7 - Morse signal and spectrogram

The function responsible for this part of the program is the 'Sin' function. After displaying the graphs, the function calls the 'sound' function, over the Morse signal, so that we can hear how the signal sounds.

The process ends with a call to the function 'Speak' that converts the text to speech, allowing the user to listen to the message. This function uses the google API to convert messages from text to speech.

IV. MACHINE TO MACHINE COMMUNICATION

To test machine to machine communication we built a firmware version of the Matlab converter, which allows the Arduino to code text messages into Morse code. The message can then be transmitted either by sound, light or any other form

to another Arduino (or other device) in order to be decoded and display as text, or sent to the Matlab to be converted into speech sound.

Figure 8 shows the hardware simulation of the machine to machine prototype. In order to introduce the Visual Light Communication (VLC) the hardware implementation uses light.

There are two LED, used for signal emission, and two phototransistors, used for the signal reception. Although this is a very simple setup it represents the conceptual theory behind VLC.

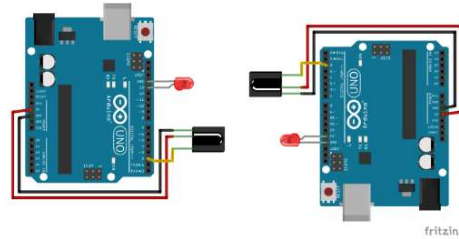


Fig.8 - Machine to machine

This simple hardware implementation allows VLC between both machines.

The use of white LEDs for illumination allows this type of communication to happen at the same time making this a wide area of application and a considerable interest is around building applications that make use of this technology. VLC provides, for example, equipment and staff communications with no EMI and RFI problems on Hospitals [11]. Multiple LED light bulbs can be used with relative location for more accurate indoor positioning and navigation [12]. Also in the malls VLC tags, can provide, for positioning and localization purpose [13][14].

For our tests we transmit 1000 English words at several different distances, using one 5 mm LED as emitter and one 3 mm NPN phototransistor as the receiver, and we have achieved a zero error rate as far as one meter long.

Better distances can be achieved by using transmission arrays [6]. There can also be array arrangements to have a certain transmission frequency, offering a data-rate of 40Mb/s for Non-Return to Zero (NRZ) On-Off Keying (OOK) [7].

On a second set of tests we have transmitted 1000 times the word "CODEX" and count the times each transmitted letter arrived to the receiver machine without errors.

A. Li-Fi implementation

The first experiment on VLC was conducted by Alexander Graham Bell in 1880 with a system called the photo-phone [1], depicted in figure 9.

We can see that a sunbeam, reflected on a small mirror, was used to transfer voice. It is also interesting to know that Bell did

believe that the photo-phone was “the greatest invention he has ever made, greater than the telephone”, as he told a reporter shortly before his death [2].

So like Morse, Optical Wireless Communication, can be seen as a rebirth of an ancient technology.

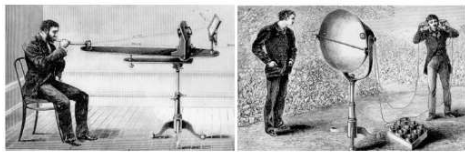


Fig.9 - Bell's Photo-phone U.S. Patent #235199.

In a broadband manner of speaking Optical Wireless Communication (OWC) refers to the transmission of data using light propagating without being guided by any waveguide and through free space or air. So OWC includes, besides VLC, the Infrared (IR), and Ultra Violet (UV), and although OWC chair a story as long as RF, the research contributions targeting the use of visible light to communications are quite limited and only a few papers, dated from 1999 [3] and 2000 [4], address low-bit-rate systems and applications in narrowband and visible light.

Later in 2001 Takana et al. [5] address the use of visible light communications using WLED for broadband domestic applications. By the end of 2003 the Visible Light Communications Consortium (VLCC) was established in Japan. Companies like Samsung, Toshiba, NEC, KDDI, Panasonic, Sony, Toyota, Sumitomo, Mitsubishi, NTT DoCoMo, Casio and Sharp were some of the first members.

The consortium was established to develop, test, investigate, plan, and standardize ubiquitous high-speed biologically friendly VLC LED systems. Later on, the Wireless World Research Form (WWRF) initiated some activity on this subject, but only in 2007, with the appearance of the subject in the Scientific American by Mohsen Kavehrad [5], grabbed the research community attention in Europe and the United States with the funding of the hOME Gigabit Access (OMEGA) project, seeking to develop global standards for home networking, including the use of optical wireless using infrared and VLC technology.

By the end of 2009 the IEEE issued a Call for Contributions on IEEE 802.15.7 VLC standard.

The Morse encoder that we have implemented can be used to transmit VLC and therefore we can speed up communications, in line of sight to the speed of light.

It is also very interesting to note that if we have applied the Morse code principles to VLC communications we can, at least in theory, achieve 3^n bps rate, where n is the number of bits transmitted, instead of the traditional 2^n .

V. RESULTS

The first developed prototype allows users to write simple Morse code messages. Without having prior experience with

Morse code, it introduces students to Morse code syntax and allows them to understand the transmission fundamentals.

It also allows users to listen to the written messages in Morse and in text to speech.

The second prototype allows machine to machine communication through VLC. Chart 1 summarizes the results obtain concerning wpm versus transmission errors.

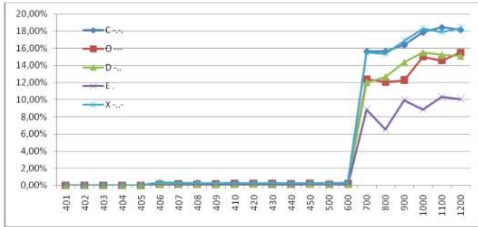


Chart 1 - Wpm versus Transmission errors

It uses Morse code in order to facilitate user understanding and allows transmission over the air without errors, to a maximum distance, in the simple prototype, of 80 cm.

Based on the error we decide to check the transmitted and received signal on the oscilloscope in order to understand what's happening. Figure 10 shows channel 1 and 2 of the oscilloscope as transmitter and receiver respectively.

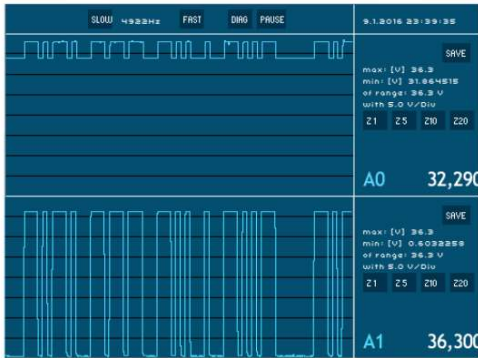


Fig. 10 - Oscilloscope Channels

The signal shows the word “CODEX” from left to right. It's also clear that we are receiving the correct signal although transmitting at 1200 wps. Therefore the errors obtain were introduced during the decoding stage, as result of some phase shift introduced by the transmitter microcontroller.

Table 2 represents the words per minute versus error rate per transmitted character. To obtain the results we have transmitted 1000 times the word “CODEX” and compared the transmitted Letters with the received ones calculating the error for each character.

Table 2 - Wpm versus error rate per transmitted character

wpm	C --	O --	D --	E .	X --
401	0,01%	0,00%	0,01%	0,00%	0,01%
402	0,02%	0,00%	0,01%	0,00%	0,01%
403	0,01%	0,00%	0,01%	0,00%	0,01%
404	0,00%	0,00%	0,00%	0,00%	0,01%
405	0,01%	0,00%	0,01%	0,00%	0,01%
406	0,30%	0,12%	0,14%	0,00%	0,33%
407	0,21%	0,09%	0,12%	0,00%	0,28%
408	0,25%	0,12%	0,15%	0,00%	0,26%
409	0,22%	0,08%	0,09%	-0,02%	0,23%
410	0,28%	0,15%	0,13%	0,00%	0,24%
420	0,21%	0,16%	0,15%	-0,02%	0,25%
430	0,28%	0,15%	0,13%	0,01%	0,26%
440	0,20%	0,07%	0,11%	0,00%	0,24%
450	0,23%	0,13%	0,14%	0,00%	0,26%
500	0,23%	0,12%	0,11%	0,00%	0,21%
600	0,28%	0,16%	0,13%	0,00%	0,28%
700	15,60%	12,40%	12,00%	8,80%	15,50%
800	15,60%	12,05%	12,65%	6,55%	15,35%
900	16,35%	12,25%	14,35%	9,95%	16,85%
1000	17,85%	15,00%	15,50%	8,85%	18,25%
1100	18,45%	14,55%	15,20%	10,30%	17,90%
1200	18,15%	15,50%	15,10%	10,05%	18,35%

From table 2 it is clear that the error rate increases as the character is deeper in the coding tree. Therefore character "E", which is on the top of the tree, has the lower error rate, and characters "X" and "C", which are in the bottom of the Morse tree, have the higher error rate and this makes sense considering the phase shift.

VI. CONCLUSIONS AND FUTURE WORK

The developed prototypes allow users to lay the bases of communication and introduce the concepts necessary to understand Visual Light Communication. Results show that even with a simple hardware configuration it is possible to achieve a high quality transmission with low error rates at speeds of 1200 words per minute. To achieve higher transmission rates it is necessary to change the firmware used by the prototypes for example using interrupts to measure the received signal allowing even higher transmission rates. In the transmitter side it is necessary to use array of LEDs and faster microcontrollers in order to process the data received in the serial port and convert it to Morse before sending it to the receiver. In the receiver side, an amplifying receiving stage will allow longer receiving distances. The use of the amplifying receiving stage will also allow the receiving firmware to use interrupts instead of time sampling, making measurements more precise.

It is clear that VLC technology introduces new challenges and this work allows users to give the first steps in this technology.

The developed prototypes present some of the limitations and gave students tools and clues to some of the possible paths to find the answers.

Future work will implement a prototype that uses the LED driver 6 channels, developed by Globaltronic, to allow customers to interact with end users throw a short message system, giving information about the location and general information of the products exhibit.

ACKNOWLEDGMENT

This work is partially financed by the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology).

REFERENCES

- [1] Mohsen Kavehrad, "Sustainable Energy-Efficient Wireless Applications Using Light", IEEE Communications Magazine, December 2010.
- [2] Bhalerao, M. V., S. S. Sonavane, and V. Kumar. "A survey of wireless communication using visible light." IJAET, Jan (2013).
- [3] G. Pang et al. "Visible Light Communication for Audio Systems," IEEE Trans. Consumer Elect., vol. 45, no. 4, Nov. 1999, pp. 1112–18.
- [4] Y. Tanaka, S. Haruyama, and M. Nakagawa, "Wireless Optical Transmission with the White Colored LED for the Wireless Home Links," Proc. IEEE PIMRC, London, UK, 2000, pp. 1325–29.
- [5] M. Kavehrad, "Broadband Room Service by Light," Sci. Amer., July 2007, pp. 82–87.
- [6] O'Brien, D.; Zeng, L.; Hoa Le-Minh; Faulkner, G.; Walewski, J.W.; Randel, S., "Visible light communications: Challenges and possibilities," in Personal, Indoor and Mobile Radio Communications, 2008. PIMRC 2008. IEEE 19th International Symposium on , vol. no., pp.1-5, 15-18 Sept. 2008 doi: 10.1109/PIMRC.2008.4699964
- [7] E-Minh, H., O'Brien-Dc, Faulkner, G., Zeng, L., and Lee, K.: 'High-Speed Visible Light Communications Using Multiple-Resonant Equalization', accepted for publication in IEEE photonics technology letters, 2008
- [8] M. Petre and B. Price, "Using robotics to motivate 'back door' learning," Education and Information Technologies, vol. 9, no. 2, pp. 147–158, 2004. [Online]. Available: <http://dx.doi.org/10.1023/B%3AEAIT.0000027927.78380.60>
- [9] HELD, Gilbert. Compressão de Dados. Tatuapé, SP: Erica, 1998. 390 p. ISBN 85-7194-110-6
- [10] Huffman, D.A., "A Method for the Construction of Minimum-Redundancy Codes," in Proceedings of the IRE , vol.40, no.9, pp.1098-1101, Sept. 1952 doi: 10.1109/JRPROC.1952.273898
- [11] Birendra Ghimire and Harald Haas, "Self-organising interference coordination in optical wireless networks", EURASIP Journal on Wireless Communications and Networking, 2012
- [12] D. O'Brien, H. L. Minh, L. Zeng, G. Faulkner, K. Lee, D. Jung, Y. Oh, and E. T. Won, "Indoor visible light communications: Challenges and prospects", Proceedings SPIE, vol. 7091, 2008
- [13] Soo-Yong Jung, Swook Hann, and Chang-Soo Park, "TDOA-Based Optical Wireless Indoor Localization Using LED Ceiling Lamps", IEEE Transactions on Consumer Electronics, Vol. 57, No. 4, pp: 1592-1597, 2011
- [14] IEEE 802.15 LED(Light Emitting Diode) Interest Group (IG-LED)

Anexo_F

Genomic Signal Processing

Digital Signal Processing
MSc in BioInformatics and Applications to Life Sciences
UTAD - 2020

Identificação de filtros FIR e IIR usando o método dos Mínimos Quadrados (LMS)

Digital Signal Processing
MSc in Electrical and Computer Engineering
UTAD

(waiting for publication)

Genomic Signal Processing

Fundamental key points of this work:

Brief introduction to understand Molecular Biology concepts in our presentation:

- ✓ The structure of DNA of eukaryotes contain genic and inter-genic regions;
- ✓ In the genic regions introns or exons are present;
- ✓ Exons are protein-coding regions of the genome;
- ✓ Comparing sequences of DNA enable us to infer evolutionary relationships between sequences;
- ✓ This requires multiple sequence alignment algorithms that are relatively computationally costly;

Signal Processing of DNA sequences:

- ✓ Sequences of DNA are inherently signals as they are functions of an independent variable, position of a sequence;
- ✓ DNA can be converted into numerical values from its bases (A,T,G,C) depending on different models discussed in this work. The one we use is based on their electron-ion interaction potential;

Signal Processing in Gene Prediction:

- ✓ Protein-coding regions exhibit a 3-base periodicity property due to *codon bias*;
- ✓ This can be detected by analyzing their spectra, produced by discrete fourier transform (DFT);
- ✓ Exons exhibit a peak at $2\pi/3$ in their fourier spectra;
- ✓ Digital filters are applied to sequence containing multiple regions of exons and introns in order to isolate peaks representing exons;
- ✓ IIR filters like inverse chebyshev is applied as a zero phase bandpass filter to isolate $2\pi/3$ regions;
- ✓ A smoothing filter Savitzky-Golay is applied to smooth the resulting noisy DNA signal;
- ✓ Final comparison between exon-identifying peaks and real value exons (verified through *Genscan*);

Signal Processing in alignment-free phylogenetic analyses:

- ✓ The fourier transform allows transition of the multiple sequences in time domain to the frequency domain;
- ✓ All spectra are evenly scaled in size;
- ✓ Euclidean distance between the spectra is calculated;
- ✓ Distance-based phylogenetic tree construction algorithm UPGMA is applied, resulting in a final tree;
- ✓ A variation is also studied using convolution of the DNA sequence numerical values and mean kernel of size 3, and the process is repeated, creating another tree;
- ✓ Result comparison between the originated trees from fourier spectra method described here and traditional multiple sequence alignment algorithm (we used *Clustal W*) and phylogenetic construction using different algorithms based on distances UPGMA, based on maximum likelihood RAxML and based on bayesian inference Mr Bayes.

**UTAD
CONFIDENTIAL****utad** UNIVERSIDADE
DE TRÁS-OS-MONTES
E ALTO DOURO

Genomic Signal Processing

Ana Rita Ferreira; João Eduardo Teixeira

University of Trás-os-Montes and Alto Douro
MSc in Bioinformatics and Applications to Life Sciences
February, 2020

Final work report for evaluation of the curricular unit of Digital Signal Processing
Professor: Salviano Soares

Abstract

Improvements in the production of data through high-throughput techniques have allowed for a fast accumulation of biological data. The application of computational techniques in order to organize, manipulate and extract information from biological data is known as Bioinformatics. Digital Signal Processing tools can provide support in both common and emergent bioinformatics tasks. The merge of Digital Signal Processing with Bioinformatics is known as Genomic Signal Processing. In this final report we explain the signal processing tools and its applications to this field of research. We develop a program to analyse these subjects and its utility alongside with a critical review of their biological meaning.

Keywords: *Digital Signal Processing, Bioinformatics, Fourier Transform, Protein Coding Regions Prediction, IIR Filters, Phylogenetic Analysis*

I. INTRODUCTION

Ongoing research within the "omics" science fields such as Genomics, Proteomics and Metabolomics has provided an enormous quantity of data and together aim to fully understand systems biology. Genomics is defined as field that studies the structure, function, evolution and mapping of DNA sequences, while Proteomics and Metabolomics are fields that are concerned with the study of proteins and metabolites respectively [1]. In this work, our focus will be on Genomics and Comparative Genomics. It is becoming progressively important to process the data from Genomics, in order to infer valuable information. Signal processing techniques displayed an important role in some of the tasks related to this data extraction in Bioinformatics. For instance, the most notorious is the bioinformatics tool for multiple sequence alignment MAFFT (Multiple Alignment Fast Fourier Transform) [2]. The use of Digi-

tal Signal Processing (DSP) techniques in Bioinformatics creates a entirely new field, Genomic Signal Processing (GSP). The prediction of genes is a common task and a common study topic among bioinformaticians, there are multiple techniques that predicts a gene in a sequence of DNA, one of the valid alternatives is the use of Digital Signal Processing [3].

It is important to note that research in Genomic Signal Processing is mainly done, in the past years, not to develop new algorithms necessarily, but to improve already existing ones. The main advantage with using Digital Signal Processing in this field is to ease the computation cost, as many of these techniques provide less computational cost [4]. Problems addressed in Bioinformatics, for the most part, come with a high computational cost. For instance, multiple sequence alignment is proven to be a NP-complete optimization problem while assembling a phylogenetic tree constitutes a NP-hard optimization prob-

lem [5]. We developed an algorithm constituting two parts of Gene prediction and Phylogenetic analysis, with implementation in MATLAB R2019b with the support of the Bioinformatics Toolbox and the Digital Signal Processing.

In this document, we provide an introduction to some concepts in molecular biology that are important to understand this work in Section 2. Section 3 explains the necessary letter (DNA base) to numeric conversion based on some properties, properly discussed. In Section 4 we discuss our first case study of identifying exonic regions based on signal processing techniques like Fourier transform and the use of digital filters. Another case study, inside the scope of phylogenetics, is presented also in section 4. In this work we decided not to include a section exclusive to explaining the tools in Digital Signal Processing and instead compromise with explaining the necessary methodologies during the work. In both our case studies we compare the performance of the algorithm with state of the art methodologies.

II. MOLECULAR BIOLOGY FUNDAMENTALS

Life on earth is organized in an enormous number of forms and features in almost all conceived environments and also characterized by adaptation. Despite all the existing diversity, all living organisms share a feature in common: they all have a similar genetic system.

The genome of any specie is its complete set of genetic instructions, and all genomes are encoded in nucleic acids (DNA and/or RNA). The way of encoding the genetic instructions between species is in the same format and, in rare exceptions, the code is identical.

These common features of heredity prove that all life on Earth arose from the same primordial ancestor billions of years ago. Like this, we can say that life's diversity and adaptation through the years are products of evolution. Some authors describe evolution as a two-step process: in the first one, the differences are inherited and arise in a randomly way; then, the individuals with particular differences increase or decrease. In other words, the genetic recombination of genes gives rise to different features; individuals who express these features may have advantage in some type of habitat (increasing its population) and individuals without those features will not survive (decreasing). Therefore, genetic variation is the foundation of all evolutionary change and is ultimately the basis of all life as we know it.

The gene is known to be the fundamental unit of heredity encoding the genetic information. This genetic information is transformed in proteins that will express the

encoded characteristics. Fig. 1 represents an example of a gene located in chromosome 1 of humans. Its precise location is given by the blue arrow. As we may see, one chromosome can contain several genes although they're not all of the same type. 1% express proteins (exons) and 99% express regulatory functions for genetic operations (introns).

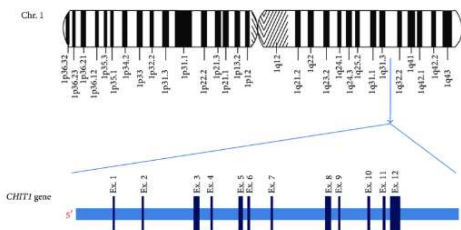


Figure 1: Schematic representation of human chromosome 1 and CHIT1 gene on chromosome 1q32 locus.

It is important to understand for our work, that there are genic sequences and intergenic sequences, and in the genes, there exists introns and exons. Exons are also called protein-coding regions because they give instructions in order to produce a functional element. In Bioinformatics, one of the most common tasks involve the prediction of these exonic regions for a wide variety of studies. The genetic information is carried in DNA, in chromosomes, that is transcribed to RNA to be, then, translated in proteins (Fig. 2).

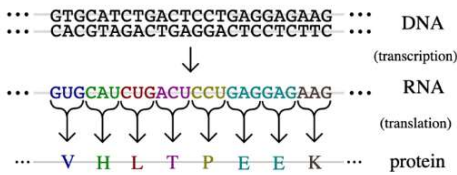


Figure 2: Central dogma of biology - from DNA to RNA to proteins.

DNA is composed by four nucleobases: A (Adenine), C (Cytosine), T (Thymine) and G (Guanine); while RNA, in other hand, is composed by A (Adenine), C (Cytosine), U (Uracil) and G (Guanine). The translation of RNA into proteins is done based on the genetic code.

In order to find similarity regions between two different sequences of DNA, RNA or proteins there is the

technique of sequence alignment. The alignment is made between a known sequence and an unknown sequence.

Pairwise Sequence Alignment is used to identify regions of similarity that may indicate functional, structural and/or evolutionary relationships between two biological sequences (protein or DNA or RNA).

By contrast, Multiple Sequence Alignment (MSA) is the alignment of three or more biological sequences of similar length. From the output of MSA applications, homology can be inferred and the evolutionary relationship between the sequences studied.

Molecular phylogenetics is the study of evolutionary relationships. It analyses the genetic, hereditary molecular differences, predominately in DNA sequences. This process helps us to gain information on an organism's evolutionary relationships. From these analyses, it is possible to determine the processes by which diversity among species has been achieved. The result of a molecular phylogenetic analysis is expressed in a phylogenetic tree. The first phylogenetic tree was scratched by Charles Darwin when he wrote his book about the evolution of species (Fig. 3).

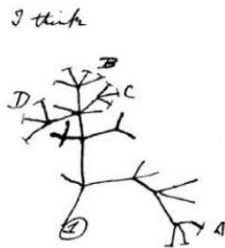


Figure 3: Charles Darwin first phylogenetic tree.

Phylogenetic trees show the evolutionary relationships among various biological species or other entities—their phylogeny. The tree in Fig. 4 is showing the three life domains: *bacteria*, *archaea*, and *eukaryota*. The black branch at the bottom of the phylogenetic tree connects the three branches of living organisms to the last universal common ancestor.

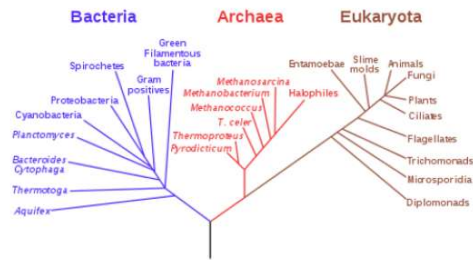


Figure 4: A rooted phylogenetic tree.

Computational phylogenetics is the application of computational algorithms, methods, and programs to do phylogenetic analyses. The goal is to assemble a phylogenetic tree. The main methods are:

- UPGMA (Unweighted Pair Group Method with Arithmetic mean);
- Maximum likelihood;
- Bayesian Inference

In order to computationally analyze DNA sequences, they need to be sequenced. This means to determine the sequence of nucleotides (As, Ts, Cs, and Gs) in a piece of DNA and then determine all the genome sequence. For instance, the human genome was completed in 2003, after a many-year, international effort. Nowadays, the technology of sequencing is much more advanced and less expensive, although it remains a complex task. It requires breaking the DNA of the genome into many smaller pieces, sequencing the pieces, and assembling the sequences into a single long "consensus."

The technology consists of marking nucleotides with dyes of fluorescence intensity, that will cross a "finish line" at the end of the tube and will be illuminated by a laser, allowing the attached dye to be detected. Thus, from the colors of dyes registered one after another on the detector, the sequence of the original piece of DNA can be built up one nucleotide at a time. The data recorded by the detector consist of a series of peaks in fluorescence intensity, as shown in the chromatogram above. The DNA sequence is read from the peaks in the chromatogram.

Next-generation sequencing, also known as high throughput sequencing enabled researchers in the past few years to gather large amount of genomic data at a relatively low cost. This led to a quick advancement of research tools to extract information from this data.

III. NUMERICAL REPRESENTATIONS OF BIOLOGICAL SEQUENCES

In order to compute signal processing techniques into biological sequences, we have to first convert the alphabetic DNA base code (A,T,G,C) into numerical values. There are three main groups of methods of numerical representations, fixed mapping methods or simple numerical representations, physico-chemical based property based mapping and at last, statistical based property mapping representations [6].

The first group the DNA sequences are converted into a series of arbitrary numerical sequences. This is the most simple type of representation as that a substitution is made in the four nucleotides, each with its own mapping of numbers. It constitutes the following main schemes:

- Voss representation;
- 2 bit binary.

The first group includes the transformation of a DNA sequence in which each of the nucleotides is represented by four numerical values of zeros and ones, where the position of the one in the four total defines that nucleotide. The 2 bit binary represent models of substitution based on replacing a nucleotide with a specific number. 2 bit binary, for instance, replaces each of the nucleotides A, T, G and C with the following numerical code: 00, 01, 10 and 11 respectively [7].

The second group, based on physico-chemical properties entices methods such as:

- Genetic Code Context (GCC);
- Electron-ion interaction potential (EIIP);
- Atomic number.

These models are based on physico-chemical properties of each of the nucleotides, each of the models replaces the DNA base letter by a corresponding biophysical or biochemical property, accounting for atomic number or the potential of interaction of electrons and ions in the molecule based on the nucleotide. The numbers were previously established via biological studies. In the most general terms, GCC takes the context of coding frames within DNA by transforming the indicator sequence according to the amino acids, corresponding to the triplets of DNA, properties of hydrophobicity, described by a unique complex number [6].

The final representation is the statistical based characteristics including as an example the frequency of nucleotide substitution (FNO). We will not describe this final group of methods, but it should be noted for anyone who aims to start a project in GSP. FNO approaches the DNA sequence based on the bases's frequency of occurrence in the different datasets [6].

In this work, we decided to go for the EIIP representation for two main reasons. The first reason is that most authors state the results to be more accurate, and the second based on personal opinion, the representation of the biological sequences via a biological property seems to us to be more correct.

Nucleotides	A	T	G	C
Voss Representation	1000	0001	0010	0100
2 bit binary	00	01	10	11
EIIP conversion	0.1260	0.1335	0.0806	0.1340
Atomic number conversion	70	66	78	58

Table 1: Most common numerical representations of DNA sequences used.

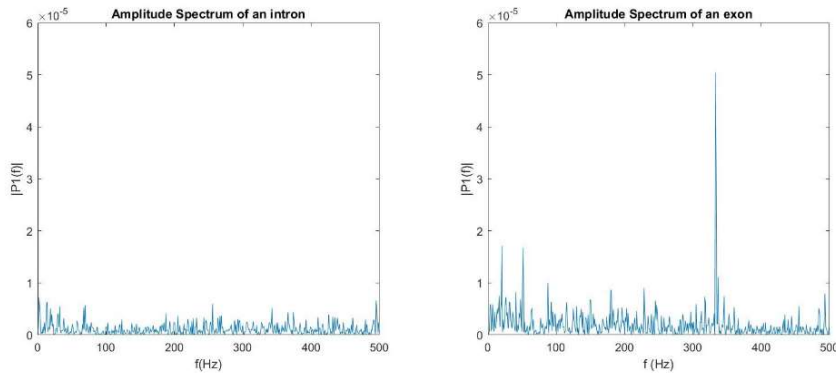


Figure 5: MK541389.1 spectrum (on the left) and GU571285 spectrum (on the right).

IV. PROTEIN CODING REGION PREDICTION

Base sequences that produce mRNA, that consequently translate into amino acids via ribosomes, are called protein-coding regions or exons. The signal conversion of the DNA sequences that are protein-coding typically exhibit the so called period-3 periodicity. This periodicity is present on the majority of exons of eukaryotes [8]. There is not to this day a detailed explanation to why the protein coding regions present this component, but some authors theorize that the cause is a nonuniform codon usage, named codon bias. A codon bias is a concept that explains the nonuniform probabilities that codons are used within a sequence [9].

Due to this periodicity, techniques in DSP can often be used to detect exons, by computing the discrete fourier transform (DFT) of the DNA signal. This transformation allows the signal in time domain to be converted into the frequency domain, causing a facilitated perception on the presence of the 3-base periodicity. Many authors confirm that it is the case with the spectrum of the protein-coding sequence exhibiting a peak in the region $2\pi/3$ [10].

In our implemented algorithm we performed the fast fourier transform (FFT) by the given MATLAB function `fft` available with the Digital Signal Processing Toolbox, on two sequences: an intronic (accession number: MK541389.1) and exonic sequence (accession number: GU571285), both retrieved via Genbank with the use of the MATLAB function `getgenbank`, provided with the Bioinformatics Toolbox.

It was expected that we see a peak in the $2\pi/3$ in the exonic region and absent in the intronic sequence. Our results displayed in Fig. 5 again confirm what many authors had previously confirmed.

In sequences that contains both introns and exons, a possible way to detect exons is to perform a windowed fourier transform. The window size is selected and we can obtain the fourier spectra from different portions of the DNA sequence, being able to isolate exonic regions this way [11]. This can be performed with use of the MATLAB function (DSP Toolbox) `stft` (short time fourier transform). However, we can also isolate exonic regions by applying digital filters. The main objective is to isolate the $2\pi/3$ region of the DNA spectra.

When exploring digital filters, we have to properly choose a filter adequate for a specific problem [9]. There exists two main groups of filters, the FIR (finite impulse response) filter and the IIR (infinite impulse response) filter. IIR filters are often used in these types of problems due to their high speed, but being less overall accurate [12]. The classification of IIR filters are based on the equivalent analog filter approximation into Butterworth, Bessel-Thomson, Elliptic, type I Chebyshev and type II Chebyshev or Inverse Chebyshev [6]. We decided to follow [6] filter design. The authors explain that their selected filter Inverse Chebyshev works well because of its high selectivity, enabling a low-order filter and because it does not display undulations in the passband amplitude response [6].

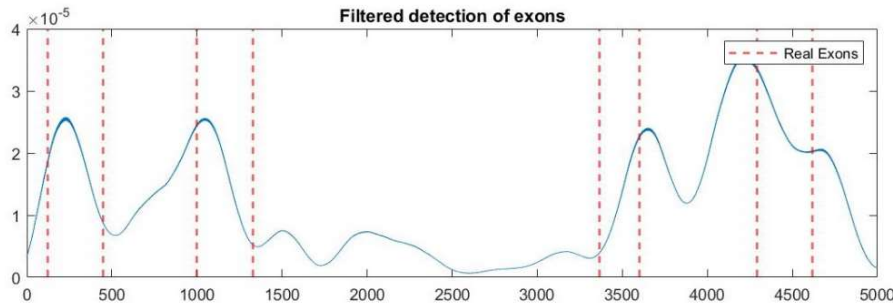


Figure 6: Use of digital filters to identify exonic regions from *Homo sapiens distal-less homeobox protein (DLX3)* gene (accession n^o: AF028233).

The filter is applied by using a technique known as zero-phase filtering. This is applied because IIR filters normally have a phase response delay. In order to correct this using the function *filtfilt* from the DSP toolbox, zero-phase filtering is applied. To measure the power levels of the signals, a power computation is also applied, squaring the signal amplitude [6].

At last, the Savitzky-Golay filter was used to remove the background noise, enabling a more precise and more smoothed signal [6].

Our results are shown in Fig. 6 where the peaks represent detected exons. For comparison, we used the bioinformatic tool *Genscan* to predict the exon positions and we plotted (striped red) those regions. The peaks do not precisely match but they are obviously correlated with the results. A more accurate detection was to be led by a more accurate choice of filters and a more optimized set of filter parameters (filter order, stopband attenuation, passband attenuation).

V. PHYLOGENETIC ANALYSIS

Molecular phylogenetic analyses allows to infer evolutionary relationships between sequences of DNA [13]. For this objective to be accomplished the sequences have to be compared based on their distance. There are two groups of methods to compare biological sequences, alignment methods and alignment-free methods. The first is a well established group of algorithms that align the sequences. The most common and well explored method is multiple sequence alignment (MSA)[14]. There are a lot of algo-

rithms used for this, each differing in the algorithm but all striving for the same goal, align more than two sequences. Most popular MSA algorithms include ClustalW, T-Coffee and MAFFT [15, 16].

This work will focus on alignment-free methods, one that uses DSP approaches. Our algorithm based on Changchan et al work in [17], computes the discrete fourier transform of the DNA signals and proceeds to calculate the euclidean distances between each point of frequency from each sequence to another. Our study case involved 25 sequences of different strains of the Influenza A virus retrieved from Genbank. Our goal is to, therefore construct a phylogenetic tree involving the 25 sequences based on their DNA spectra.

After computing the euclidean distances, the result is a distance matrix. With this as input we are already able to construct a tree based on distances. One of the phylogenetic tree construction methods is the known UPGMA. Using the *seqlinkage* from the bioinformatics toolbox we get the results depicted in Fig. 3 on the left. We also performed the same phylogenetic tree construction but with sequence comparison based on MSA. We used *ClustalW* to perform the alignments and then proceeded to construct the tree based on UPGMA Fig. 7 on the right. Both trees are similar proving that, generally, the methodology in GSP works well. The trees differ slightly which is normal, since different methods produce slight changes to the tree. We also wanted to compare the produced tree with statistical-based construction of trees such as maximum likelihood and bayesian inference. The programs RAxML and Mr. Bayes were both used for maximum likelihood and bayesian inference estimation, respectively.

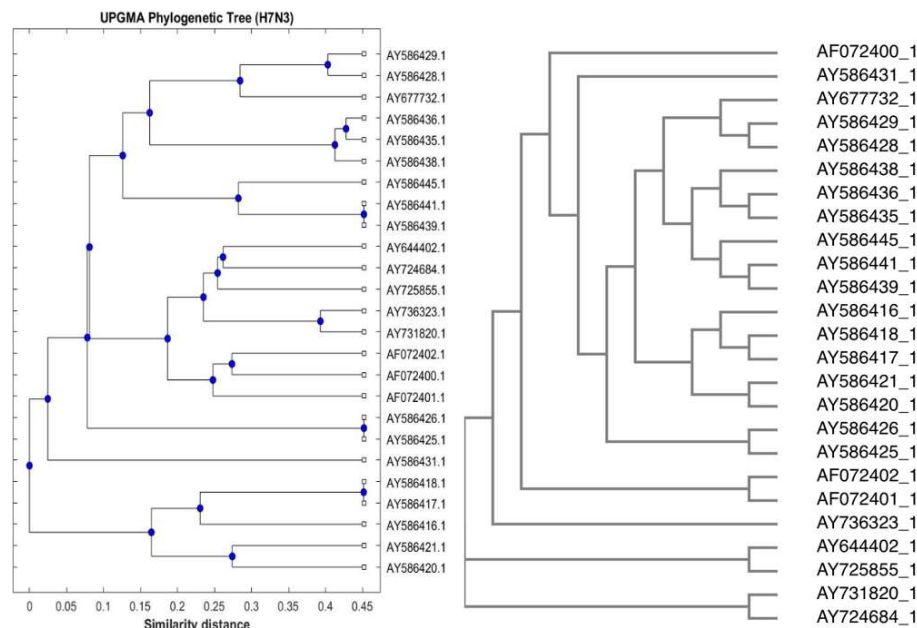
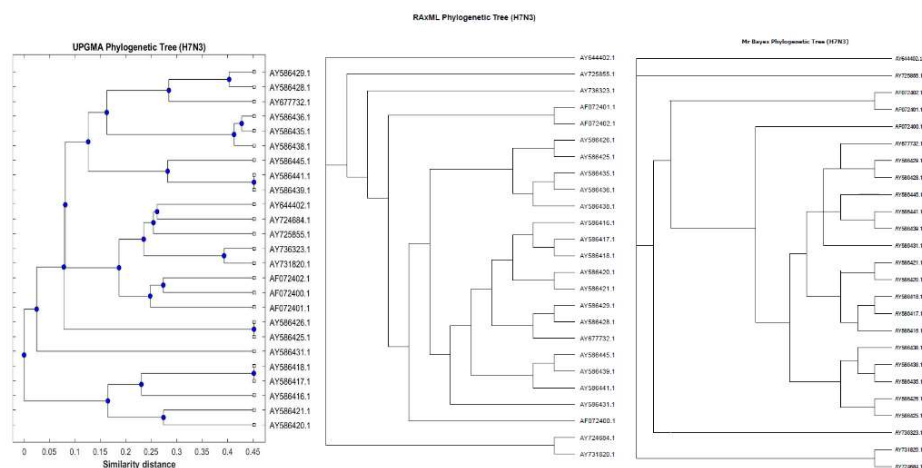


Figure 7: Comparison, side by side, of DFT based UPGMA (left) and multiple sequence alignment UPGMA (right).

Results show the tree also resembles the RAXML and Mr. Bayes produced trees, shown in Fig. 8. In our code we also implemented a different type of DNA numerical representation to evaluate the final tree. We still converted the DNA sequence based on EIIP model, however, we then proceeded to perform a convolution between the numerical sequence and a mean kernel of size 3. This is known as moving average filtering. We then evaluated the final tree and we got slightly different positions, but most accurately matching RAXML and Mr. Bayes produced trees. We cannot by this prove that this method is valid, since it would require more tests on different sequences, more sequences, different algorithms and different tree construction methods. Be that as it may, while in problems like gene prediction we want accurate data, to correctly predict exonic regions in the genome, in phylogenetic analysis the generalization in order to compare sequences may be advantageous when coupled with already-existing algorithms.

VI. FINAL REMARKS

In this work we explored the realization of DSP techniques into Bioinformatics topics like gene prediction and phylogenetic analyses. Other concepts we could have explored instead was motif discovery and DNA long-range auto-correlations. A motif is a pattern in a DNA sequence with a specific function, while correlation between sequences can reveal more hidden patterns. Though these techniques explore already existing and well defined algorithms, programs that work with signals being in time domain and specially in frequency domain tend to be much less computationally costly. Besides this, DNA signal spectra can reveal hidden features existing in DNA, like the 3-base periodicity. Techniques in GSP have the potential to complement bioinformatic algorithms by providing a less computational cost and provide support and information that can help in ultimately reaching the main goal. We hope revision works like this and the developments in both DSP and Bioinformatics will attract more research into more topics in the promising field GSP.



- [8] Soberkari, H., Shamsi, M., Sedaaghi, M., Golabi, F. (2012). Prediction of protein coding regions in DNA sequences using signal processing methods. 2012 *IEEE Symposium on Industrial Electronics and Applications*, 355-360.
- [9] Vaidyanathan, P., P., Yoon, B. (2004). The role of signal-processing concepts in genomics and proteomics. *Journal of The Franklin Institute*.
- [10] Akhtar, M., Epps, J., Ambikairajah, E. (2008). Signal Processing in Sequence Analysis: Advances in Eukaryotic Gene Prediction. *IEEE Journal of Selected Topics in Signal Processing*, 2(3), 310-321.
- [11] Yin, C., Yau, S., S., T. (2007). Prediction of protein coding regions by the 3-base periodicity analysis of a DNA sequence. *Journal of Theoretical Biology*, 247(4), 687-694.
- [12] Litwin, L. (2000). FIR and IIR digital filters. *IEEE Potentials*, 19(4), 28-31.
- [13] Yang, Z., Rannala, B. (2012). Molecular phylogenetics: principles and practise. *Nature reviews genetics*, 13(5), 303-314.
- [14] Kolekar, P., Kale, M., Kulkarni-Kale, U. (2012). Alignment-free distance measure based on return

- time distribution for sequence analysis: applications to clustering, molecular phylogeny and subtyping. *Molecular phylogenetics and evolution*, 65(2), 510-522.
- [15] Edgar, R., C., Batzoglou, S. (2006). Multiple sequence alignment. *Current opinion on structural biology*, 16(3), 368-373.
- [16] Zheng, W., Li, K., Li, K., So, H. C. (2017). A Modified Multiple Alignment Fast Fourier Transform with Higher Efficiency. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 14(3), 634-645.
- [17] Yin, C., Chen, Y., Yau, S. S. (2014). A measure of DNA sequence similarity by Fourier Transform with applications on hierarchical clustering. *Journal of Theoretical Biology*, 359, 18-28.

Identificação de filtros FIR e IIR usando o método dos Mínimos Quadrados (LMS)

Maria Manuela Silva Reis Soares

Universidade de Trás-os-Montes e Alto Douro
Quinta dos Prados
5000 Vila Real
PORTUGAL
email:msoares@utad.pt

1. INTRODUÇÃO

Modelar e identificar sistemas é uma tarefa muito abrangente de grande importância em várias áreas, tais como o controlo de sistemas, as comunicações ou o processamento de sinal. Mesmo fora das tradicionais disciplinas de engenharia, quando estudamos sistemas sociais económicos ou biológicos, a modelação continua a ser relevante.

2. SISTEMAS ADAPTATIVOS

Existem várias definições de sistemas adaptativos. Em seguida passamos a enumerar algumas delas em vários contextos:

O acto de adaptar; estado de estar adaptado; (Biologia): qualquer alteração na estrutura ou função de um organismo ou uma parte desse organismo que resulta de uma selecção natural a partir da qual esse organismo fica melhor adaptado a sobreviver ou a multiplicar-se no seu ambiente; (Oftalmologia): a regulação da pupila no sentido de controlar a quantidade de luz que entra no olho; (Sociologia): a lenta e normalmente inconsciente modificação de actividade social do indivíduo que está a ser aculturado; (Fisiologia): a adaptação dos órgãos responsáveis pelos sentidos humanos visão, tacto, olfacto e audição, às modificações constantes das condições ambientais.

Apesar da maior parte destas definições se referirem à adaptação em sistemas naturais, o conceito pode ser transportado de igual forma para sistemas adaptativos artificiais, ou seja, construídos pelo Homem.

Um autómato adaptativo é um sistema cuja estrutura é alterável ou ajustável no sentido de melhorar a seu comportamento e performance de acordo com um dado critério. Temos como exemplo um sistema de controlo automático do ganho usado nos receptores de rádio ou televisão onde a sua função é ajustar a sensibilidade do receptor inversamente à potência do sinal recebido, ou seja, se o sinal chegar com um nível de potência insuficiente o ganho deve ser de tal forma que na saída não se note esta característica de entrada.

Os sistemas adaptativos, normalmente têm algumas das seguintes características:

- Podem adaptar-se automaticamente face a mudanças ambientais ou mudanças nos requisitos do sistema;
- Podem ser treinados para realizar tarefas específicas, ou seja, os sistemas adaptativos podem ser programados por um processo qualquer de treino;
- Podem extrapolar um modelo de um comportamento de forma a reagir a novas situações depois de serem treinados para tal;
- No limite podem-se auto-reparar (corrigir), ou seja, podem adaptar-se a certos tipos de defeitos internos;

- Normalmente são descritos como sistemas não lineares cujos parâmetros variam com o tempo;
- Na maior parte dos casos são mais complexos e difíceis de analisar do que os sistemas não adaptativos mas oferecem reais potencialidades no aumento da performance quando as características do sinal de entrada são desconhecidas e variam no tempo.

2.1. UM MODELO LINEAR ADAPTATIVO

O diagrama da figura 1 apresenta a forma genérica de um modelo adaptativo linear.

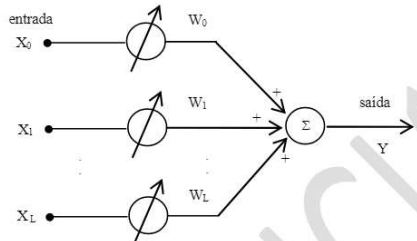


Fig.1: Forma genérica de um modelo linear adaptativo

Existe um sinal de entrada com elementos $X_0, X_1, \dots, X_L$, um conjunto correspondente de pesos $W_0, W_1, \dots, W_L$, uma unidade de soma e uma saída única Y . O acto de adaptar, consiste no ajuste dos pesos de forma a identificar um sistema.

Os elementos correspondentes ao sinal de entrada podem ser visto de duas maneiras:

- Cada entrada pode ser correspondente a diferentes sinais de entrada. (exemplo: sistema de detecção acústico onde cada linha de entrada é conectada a um sensor em separado).
- Os elementos $X_0, X_1, \dots, X_L$, podem ser as $L+1$ amostras de um sinal de entrada.

No nosso trabalho considerou-se o segundo caso (como na figura 1) em que o sinal de entrada é aplicado ao sistema desconhecido e ao sistema adaptativo, como no ponto 4 deste trabalho vamos verificar.

Se quisermos representar um sistema real com ruído através de um modelo temos que acrescentar o sinal d_k à figura 1.

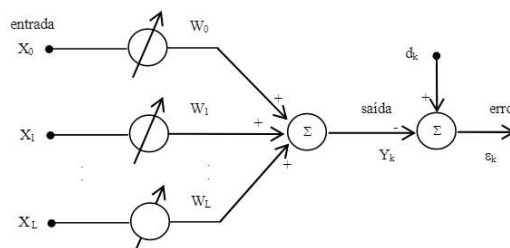


Fig.2: Forma genérica de um modelo linear adaptativo com ruído

Pela figura 2 temos que o sinal de erro é dado por,

$$\varepsilon_k = d_k - Y_k.$$

Substituindo Y_k e tomando como estimativa do gradiente o erro quadrático médio [2], temos

$$\text{MSE} = \xi = E[\varepsilon^2] = E[d_k^2] + W^T R W - 2P^T W.$$

Verificamos então, que o ξ é uma função quadrática das componentes dos pesos W quando as componentes de entrada e a saída desejada são variáveis estacionárias estocásticas.

Na figura 3, representa-se uma superfície quadrática bidimensional que pode servir de abstracção à pesquisa do valor óptimo para os pesos do filtro adaptativo. Por simplicidade perceptual, consideram-se apenas dois pesos.

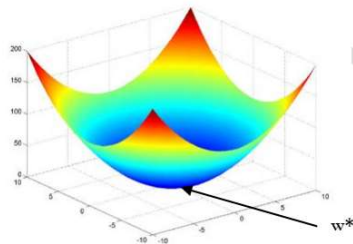


Fig.3: Parabolóide correspondente à pesquisa de solução

No eixo vertical representa-se o erro quadrático médio e no plano horizontal os valores dos pesos. Esta superfície designa-se parabolóide e se existissem mais pesos seria um hiper-parabolóide. A concavidade é voltada para cima pois de outra forma resultariam valores de pesos com erro quadrático médio negativo, que se traduz numa impossibilidade para sinais físicos reais. O ponto mínimo do parabolóide é projectado no plano dos pesos, e w^* é o ponto óptimo ou seja, o ponto onde o MSE é mínimo. Na figura 4 verificamos a evolução do gradiente ao longo da superfície do parabolóide a partir do ponto óptimo neste caso colocado no meio da figura 4.

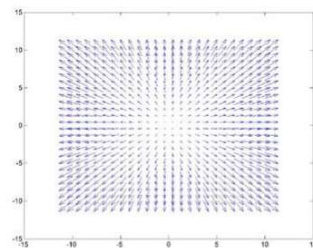


Fig.4: Evolução do gradiente ao longo da superfície do parabolóide

Nota-se claramente que no ponto óptimo (correspondente ao meio da figura) o gradiente é mínimo. De referir ainda que para esta superfície quadrática existe apenas um mínimo global, não existindo por conseguinte mínimos locais.

3. FILTROS DIGITAIS

Uma definição possível para um filtro digital é a seguinte:

- Um filtro digital é um sistema discreto projectado para deixar passar o conteúdo espectral de um sinal de entrada numa determinada banda de frequência.

As tarefas desempenhadas pelos filtros podem ser do tipo:

- ▲ **Filtragem** - Extração de informação no instante n usando dados medidos até esse instante.
- ▲ **Alisamento** - Utilização de informação passada ou futura para calcular a saída presente.
- ▲ **Predição** - Uso da informação passada para prever o valor que tomará no futuro.

3.1. Filtros digitais do tipo FIR e IIR

Os filtros digitais, devido às suas características de resposta impulsiva, podem ser classificados em filtros FIR (Finite Impulse Response) e filtros IIR (Infinite Impulse Response), ou seja, de resposta ao impulso finita e infinita respectivamente. Podem-se ainda subdividir em recursivos e não recursivos.

Os filtros FIR, correspondem a sistemas não recursivos, são sempre estáveis e causais, e normalmente apresentam características de fase linear. Os filtros IIR são recursivos, podendo ser instáveis e não causais.

Para as mesmas especificações, os filtros FIR originam sistemas de ordem superior aos filtros IIR, sendo portanto necessário um maior número de operações aritméticas na sua realização quando comparamos com os filtros IIR. Por outro lado, o efeito dos erros de quantificação é mais significativo nos filtros IIR.

O comportamento dum sistema contínuo, linear e invariante no tempo, pode ser aproximado pela equação diferencial no domínio do tempo:

$$y^{(N)}(t) + \sum_{k=0}^{N-1} a_k y^{(k)}(t) = \sum_{k=0}^M b_k x^{(k)}(t)$$

O seu comportamento no domínio da frequência faz-se para a sua função de transferência, calculando a transformada de Laplace, através da variável complexa $s=j\omega$, definida por:

$$H(s) = \frac{Y(s)}{X(s)} = \int_{t=0}^{+\infty} h(t) e^{-st} dt$$

A representação discreta dum sistema contínuo, linear e invariante no tempo, origina um sistema discreto linear e invariante, caracterizado pela equação às diferenças:

$$y[n] + \sum_{i=1}^N a_i y[n-i] = \sum_{i=0}^M b_i x[n-i]$$

Normalmente, os filtros são descritos pela sua equação às diferenças ou a versão discreta da

transformada de Laplace ou seja, a transformada Z. Assim, para os filtros IIR, temos:

$$y[n] = \sum_{k=0}^M a_k x[n-k] + \sum_{k=1}^N b_k y[n-k] \Leftrightarrow H(z) = \frac{\sum_{k=0}^M a_k z^{-k}}{1 - \sum_{k=1}^N b_k z^{-k}}$$

Do mesmo modo para FIR:

$$y[n] = \sum_{k=0}^M b_k x[n-k] \Leftrightarrow H(z) = \sum_{k=0}^M a_k z^{-k} \quad (*)$$

3.2. Síntese de filtros

A síntese de filtros digitais faz-se usando vários métodos, sendo estes diferentes tratando-se de filtros FIR e IIR.

Os coeficientes dos filtros IIR podem ser obtidos através das características dos filtros analógicos correspondentes segundo vários métodos, nomeadamente, a conservação da resposta impulsiva, a transformação bilinear e pela localização de pólos e zeros da função de transferência. Porém, embora simples, este último método não é indicado para filtros mais complexos, devido aos efeitos dos erros numéricos na localização de pólos e zeros. O método mais usual é o método da transformação bilinear.

Os filtros FIR não podem ser obtidos pelas aproximações utilizadas nos sistemas contínuos, dado que não existem sistemas analógicos com resposta impulsiva finita. A resposta impulsiva dos filtros FIR é definida em (*). Para síntese dos filtros FIR usam-se, geralmente, o método dos mínimos quadrados (LMS) ou o método das janelas, para truncatura da resposta impulsiva [3,4].

4. FILTROS ADAPTATIVOS

Os filtros adaptativos podem ser usados para modelar sistemas, ou seja, imitar o comportamento de um sistema físico dinâmico normalmente visto como uma caixa negra com uma ou várias entradas e também uma ou várias saídas.

Modelar um sistema dinâmico com um filtro adaptativo que tem uma entrada e uma saída pode ser traduzido pelo diagrama de blocos representado na figura 5.

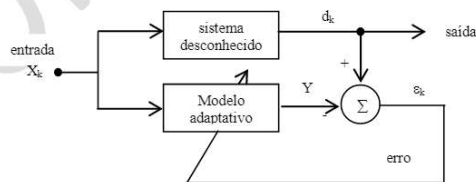


Fig.5: Modelo adaptativo

Por análise da figura 5, podemos verificar que o sinal de entrada é aplicado tanto ao sistema desconhecido como ao modelo adaptativo. Este ajusta-se por forma a que a sua saída Y_k se assemelhe à saída do sistema desconhecido d_k . Esta tarefa pode ser realizada tendo como critério o método dos mínimos quadrados [1,5].

Se o modelo adaptativo for um filtro, normalmente representa-se pelo diagrama de blocos da figura 6.

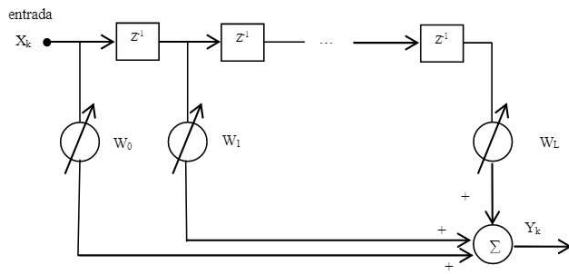


Fig.6: Diagrama dum filtro adaptativo

O sistema adaptativo deve ser flexível e ter os graus de liberdade necessários (ajuste de pesos) de forma a que a aproximação seja o mais perfeita possível. Quando isto acontece, o sistema adaptativo passa a ser o modelo do sistema desconhecido que se pretendia identificar.

O ruído faz inevitavelmente parte de qualquer sistema realizável. Assim a função desconhecida que queremos modelar tem internamente ou externamente forças aleatórias que perturbam a sua acção, ou seja, é afectada de ruído. Na figura 7 apresenta-se o diagrama de blocos que realiza a identificação de um sistema desconhecido afectado por ruído com um filtro adaptativo.

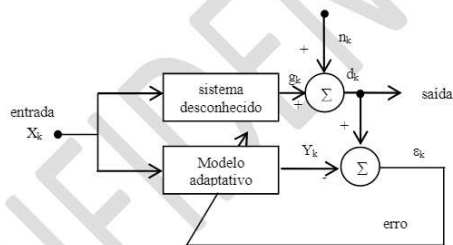


Fig.7: Modelo adaptativo com ruído

Quando um modelo adaptativo tem a flexibilidade suficiente, a sua saída assemelha-se à resposta dinâmica da saída do sistema desconhecido a menos do ruído que na figura 7 é representado por n_k . Este tipo de distúrbio é normalmente representado por ruído aditivo e descorrelacionado com o sinal de entrada. Se o modelo adaptativo for capaz de ajustar os pesos minimizando o erro quadrático médio, a solução final não vem afectada de ruído. Isto não quer dizer que a convergência do processo iterativo adaptativo, não sofra alteração, mas sim que o vector final dos pesos depois da convergência não está afectado pelo ruído introduzido.

5. O MÉTODO DOS MÍNIMOS QUADRADOS (LMS)

Existem vários métodos para encontrar o ponto mínimo da superfície da figura 3. Por exemplo os métodos de Newton's e Steepest-descent encontram o ponto óptimo para os pesos necessitando para isso de estimar o gradiente em cada iteração no sentido de encontrar o melhor caminho para o ponto mínimo [1,2]. A estimação do gradiente nestes métodos $\xi = E[\varepsilon_k^2]$, é normalmente feita

fazendo estimas de diferenças médias de ε_k^2 .

O método dos mínimos quadrados, usa uma estimativa especial para o gradiente que é válida para o modelo linear adaptativo que se apresentou na figura 2. Ou seja,

$$\varepsilon_k = d_k - \mathbf{X}_k^T \mathbf{W}_k.$$

Neste caso, toma-se o próprio erro ε_k^2 como um valor estimado para ξ_k facto este que simplifica em grande parte o algoritmo já que a tarefa de estimar o gradiente em cada iteração noutros métodos é normalmente complicada.

Temos então que em cada iteração do processo adaptativo, a estimação do gradiente é da forma

$$\hat{\nabla}_k = \begin{bmatrix} \frac{\partial \varepsilon_k^2}{\partial w_0} \\ \vdots \\ \frac{\partial \varepsilon_k^2}{\partial w_L} \end{bmatrix} = 2\varepsilon_k \begin{bmatrix} \frac{\partial \varepsilon_k}{\partial w_0} \\ \vdots \\ \frac{\partial \varepsilon_k}{\partial w_L} \end{bmatrix} = -2\varepsilon_k \mathbf{X}_k.$$

Assim, o algoritmo em cada iteração adapta os pesos $\mathbf{W}$ pela equação

$$\mathbf{W}_{k+1} = \mathbf{W}_k - \mu \hat{\nabla}_k.$$

A convergência do algoritmo encontra-se bem descrita em [2] e prova-se que o parâmetro μ deve pertencer ao intervalo

$$0 < \mu < \frac{1}{(L+1)p},$$

onde o parâmetro 'p' é a potência do sinal.

6. APLICAÇÃO

Desenvolveu-se uma aplicação em MatLab – versão 6.5 que identifica os dois tipos de filtros acima referidos. Existem quatro funções que se apresentam em anexo A que realizam esta tarefa.

Consideram-se alguns parâmetros fixos por simplicidade de análise da simulação. Este facto não implica perda de significância na qualidade dos dados obtidos.

7. CONCLUSÕES

As figuras obtidas pelas simulações apresentam-se em anexo B.

Verificamos que os sistemas FIR são mais fáceis de identificar quando comparados com os IIR, para as mesmas condições. Este facto deve-se à característica infinita da resposta impulsional dos IIR, e para além disso pode verificar-se que nos extremos do filtro a identificação é mais complicada do que para outros quaisquer pontos.

A existência de ruído faz com a identificação se torne mais complicada como seria de esperar. Apesar da convergência ser normalmente afectada, o valor final pode ser próximo do óptimo dependendo principalmente das características do ruído. Fizeram-se várias experiências e apresentam-se os resultados para dois valores de ruído.

Concluímos que a simplicidade do método LMS torna-o fácil de implementar num computador pois a estimativa do gradiente em cada iteração é simplificada. Se o sistema adaptativo é um sistema adaptativo linear, se a entrada $\mathbf{X}_k$ e a saída d_k forem conhecidas em cada iteração o algoritmo LMS é na maior parte das vezes a melhor escolha para as diferentes aplicações existentes no processamento de sinal adaptativo.

8. REFERÊNCIAS

- [1] Simon Haykin, *Adaptive Filter Theory*, 3ª edition, Prentice Hall.
- [2] Bernard Widrow, Samuel D. Stearns, *Adaptive Signal processing*, Prentice Hall,
- [3] Vijay K. Madisetti, Douglas B. Williams, *The Digital Signal processing handbook*, IEEE Press,
- [4] A. V. Oppenheim, R. W. Schaffer, *Discrete-Time Signal processing*, Prentice Hall.
- [5] Steven M Kay, *Modern Spectral Estimation – Theory and Application*, Prentice Hall.

CONFIDENCIAL