Universidade do Minho
Escola de Engenharia

Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores

---

PDS - Trabalho Prático nº2:

Deteção e Remoção de Silêncios na Fala

João Faria A85632

Junho 2020

Conteúdo

1 Introdução

Na atualidade é cada fez frequente a gravação de excertos de áudio, como a fala. Contudo, quando a mesma é gravada ocorrem sempre segmentos de silêncio que, além de não conterem informação relevante a ser trasmitida, aumentam desnecessariamente o tempo de processamento e a memória a ser alocada da mensagem enviada a ser recebida por um sistema de processamento de voz.

Deste modo, pretendeu-se elaborar um algoritmo que conseguisse distinguir e retornar, de forma eficiente, fala (limpa) de pausas e outros ruídos, aplicando diferentes limites (threshold) às amostras do sinal gravado, consoante as condições de ruído.

Para tal, foram elaborados testes com diferentes relações sinal-ruído (SNR). Variando o SNR de 0 dB a 40 dB com passos de 5 dB e modificando o valor de threshold de 0.1 a 12 conseguiu-se extrair a melhor relação threshold-SNR.

Por conseguinte, o processo foi automatizado de maneira a detetar o SNR do sinal de entrada, atribuindo-lhe o threshold mais adequado para detetar e remover os silêncios da voz.

No entanto, tal algoritmo estaria incompleto caso não fosse robusto a sons esporádicos, isto é, sons estridentes de alta amplitude, mas de duração inferior à duração da pronunciação de uma palavra, como o abrir da porta ou o estalar de dedos. Ora o mesmo consegue detetar e remover esses ruídos da fala fazendo a análise de segmentos consecutivos com amostras maioritariamente "outliers"e verificando se esse número de segmentos ultrapassa um determinado valor, sendo traduzido como fala caso esse limite, definido experimentalmente, seja ultrapassado.

Note-se também que foram adotadas outras alternativas, como usar filtros passa-baixo ou passa-alto, numa tentativa de remover esses sons dos da fala, contudo o método acima brevemente mencionado trouxe os melhores resultados.

2 Fundamentos teóricos do modelo de ruído

2.1 Ruído branco

O ruído encontra-se presente no dia-a-dia em maior ou em menor escala, dependendo da localização, e possui diversas origens, assim como também existem diversas categorias de ruído (ruído branco "white noise, ruído rosa,...).

No contexto do trabalho realizado, tentou-se emular diversas condições de interferência, aplicando ao sinal a transmitir, um ruído branco a ser somado a este.

O ruído branco é um processo de natureza estocástica onde os valores amostrados são aleatórios e estatisticamente independentes, isto é, a probabilidade de ocorrência de um valor de amostragem (ou evento) não altera a probabilidade de ocorrência de um outro valor de amostragem.

No que toca às estatisticas entre diferentes variáveis aleatórias que geram esses elementos do vetor de ruído, as mesmas são auto-correladas pelo que a sua covariância é nula.

Assim sendo, utilizou-se um modelo aditivo de ruído branco gaussiano:

- aditivo: o mesmo soma-se ao sinal original, em contraposição com o ruído multiplicativo;
- branco: analogia à cor branca, o ruído possui uma densidade espetral constante, ou seja, a potência é uniformemente distribuída pelo seu espetro de frequências;
- gaussiano: caraterizado pelas estatísticas de uma distribuição normal, média, que neste ruído é nula, e variância (desvio-padrão ao quadrado) finita e constante ao longo do tempo (processo estacionário), seguindo a dita distribuição. Assim, consegue-se extrair as ditas estatísticas deste processo a partir de um excerto de uma instância do próprio (processo ergódico).

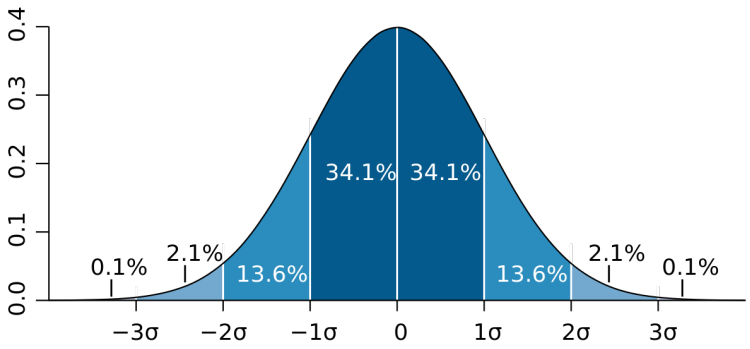

Figura 1: Curva Gaussiana: os dados concentram-se em torno da média, neste caso, 0, sendo estes representados por um tom de azul mais escuro

2.2 SNR

Partindo de um sinal gravado, s[n], o mesmo é composto por uma componente correspondente ao sinal sem ruído, p[n], (sinal puro, limpo) somada ao ruído, r[n], de fundo com que foi captado, ou propositadamente sintetizado e adicionado:

$$s[n] = p[n] + r[n] \tag{1}$$

Para estabelecer diferentes ambientes de poluição sonora, é necessário comparar a potência do sinal com a potência do ruído associado.

Tal comparação é feita na equação (2) onde se define o SNR como o quociente entre essas potências, ou ainda, o quociente do quadrado das suas amplitudes, de sinal e de ruído:

$$SNR = \frac{P_{sinal}}{P_{ruido}} = (\frac{A_{sinal}}{A_{ruido}})^2 \tag{2}$$

Expresso em décibeis, dB, vem:

$$SNR_{dB} = 10 * log_{10}[(\frac{A_{sinal}}{A_{ruido}})^2] \tag{3}$$

3 Threshold

3.1 Cálculos associados

Ao efetuar a gravação de um sinal de som, este é amostrado a uma frequência de amostragem definida a 10kHz e armazenada num vetor de dados.

Os dados, ao serem distribuídos pela curva gaussiana, concentram-se em torno de um intervalo de centro a média, μ, e vizinhança a média mais ou menos uma constante multiplicativa aplicada ao desvio-padrão, σ.

Essa constante (threshold, t) será definida de tal forma que separe, da maneira mais eficiente, dados provenientes da fala dos restantes (silêncio, barulhos esporádicos, barulhos de fundo).

Deste modo, se a distância normalizada de uma dada amostra, x_i, for superior ao threshold, então é considerado fala (ou ruído esporádico, a ser tratado posteriormente).

$$\frac{|x_i - \mu|}{\sigma} > t \tag{4}$$

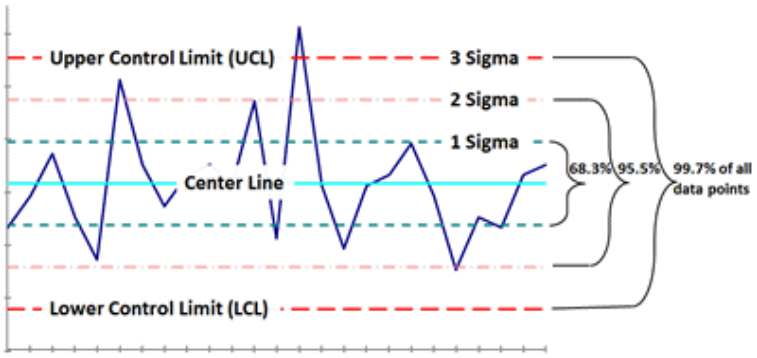

Figura 2: Dados cujos pontos seguem uma distribuição normal. Neste caso, os limites estabelecidos (UCL e LCL) são para um threshold de 3 desvios-padrões

3.2 Sensibilidade

Por um lado, conjugando o SNR com o threshold seria de esperar que quanto menor fosse o primeiro, menor o último já que, à medida que a amplitude do ruído se torna cada vez mais próxima da da fala, a margem de confiança definida pelo threshold terá de ser menor.

Por outro lado, se se diminuir em demasia essa margem, para um dado SNR, acontecerá que haverá muitas amostras de ruído fora dessa gama, tal como seria de esperar para o caso oposto, em que se se alargasse esse intervalo, muita fala de não tão alta amplitude "caíria"dentro do vetor de ruído.

A sensibilidade reside em executar sucessivas experiências, variando threshold em função de diferentes SNR, de forma a obter a combinação de SNR-threshold que fornece a melhor separação fala-ruído, para fabricação de um algoritmo automático de obtenção do melhor threshold com base na interpolação realizada na tabela descritiva dos melhores pares SNR-threshold.

4 Módulos Matlab

4.1 Casos excecionais: Métodos adicionais adotados para aumentar a eficiência do algoritmo

Também foi levado em consideração a existência de ruídos de curta duração, mas estridentes, que devem ser removidos bem como que nem sempre a potência do sinal a transmitir é superior à do ruído a ele aplicado. Ora, foram utilizadas técnicas adicionais para tratamento destes casos excecionais.

4.1.1 Ruídos breves e estridentes

A solução adotada à remoção destas interferências indesejáveis proveio da análise e comparação da duração entre segmentos de palavras e barulhos como abrir e fechar de porta, ou estalido de dedos.

Após vários testes, e tendo em conta que o excerto auditivo de 5 segundos é dividido em frames (segmentos) de 100 amostras, havendo 500 amostras ao todo, verificou-se que impondo uma condição de que, se pelo menos houver 25 frames consecutivos possuírem mais amostras de outliers que valores dentro do intervalo de confiança, esses segmentos serão considerados fala, caso contrário, irão para o buffer de ruídos, com bastante exatidão.

4.1.2 SNR's negativos: quando a amplitude da fala é inferior ao do ruído de fundo

Quanto a excertos de som com SNR negativo, até -10dBs, threshold a 0.55, utilizou-se o filtro wiener do professor Lima para extrair o sinal puro de um ruído que encobria totalmente o sinal a transmitir.
Apesar de haver segmentos auditivos de ruído que passaram para o vetor da fala, conseguiu-se reconhecer a fala sem pausas no meio, nem ainda sons como o bater de portas quer foram filtrados pelo método acima supracitado.

4.2 Código Principal

O código principal segue a seguinte linha de pensamento:

1. Aqui corresponde ao ponto onde o programa começa e ou invoca um som já gravado (1ºcaso) antes de possuir a tabela de threshold automático, ou grava um som captado pelo microfone para testar a tabela criada (2º caso).

2. Assim, no primeiro caso, contamina-se o som com a função contamina para o SNR pretendido ou, no segundo caso, invoca-se já a função de deteção de threshold automático.

3. Para SNRs muito baixos, próximos de zero ou negativos,pode-se invocar a função do filtro wiener que ajuda a eliminar o ruído que encobre o sinal.

4. Aplica-se a função de retirar ruído quer estejemos no primeiro quer antes no segundo caso.

5. Após guardar a fala e ruído, esboça-se os correspondentes gráficos e reproduz-se os sons obtidos para aferir a eficiência do algoritmo.

Consulte o fluxograma no repositório online, link na bibliografia, para melhor clarificação.

4.3 Função Contamina

```
%Contaminação do sinal original com ruído branco

function [sinal] = contaminar_v1(recorded, SNR)

S=mean(recorded.^2);                % Potência do sinal
L=S-(var(recorded(1:500)));
N=L/(10^(SNR/10));                  % Fórmula SNR

n = randn(1,length(recorded));      % Teorema do limite central
n = n - mean(n);                    %ruído branco com media 0
ruido = n * sqrt(N);
sinal=recorded+ruido';
```

Figura 3: Função Contamina

Obtém-se a potência S, do sinal original, e a potência L do sinal limpo, subtraindo S às primeiras 500 amostras onde não se fala.

Pela fórmula descrita na equação 3, calcula-se a potência N do ruído necessária para obter o SNR desejado.

Gera-se um vetor de números aletórios com distribuição normal, n, de média 0, cuja potência é obtida multiplicando pela raíz de N.

O sinal contaminado é o sinal gravado limpo mais o ruído obtido.

4.4 Função Retira ruído

```
%Separação de ruído branco esporádico da fala (sem pausas)

function [s_p, s_ruido] = retira_ruido_v1(sinal, threshold_limit, Fs)

noise_samples = floor(Fs/2);        %o primeiro meio segundo de amostras :

%calculo da media e do desvio padrao do ruido
background_noise_array=[];

for i=1:1:noise_samples
    background_noise_array = [background_noise_array sinal(i)];
end

media_ruido=mean(background_noise_array);
desviop_ruido=std(background_noise_array);
```

Figura 4: Função Retira Ruído 1/3

Calcula-se a média e desvio padrão do ruído branco do sinal passado (contaminado), analisando o primeiro meio segundo de amostras, dos 5 segundos de gravação. A gravação é dividida em amostras de 500 segmentos cada uma com 100 amostras, Fs = 10kHz.

Assinala-se como amostra vozeada caso verifique a inequação 4, da distância normalizada.

Conta-se quantas amostras vozeadas possui cada frame, sendo que caso as mesmas tenham menos que metade das amostras de uma frame (mais uma unidadde) são consideradas segmentos não vozeados.

Fez-se um esforço para minimizar o uso de ifs apenas com o intuito de ser minimizar a carga para o processador.
Claro que neste ambiente de trabalho não estamos num microcontralor de recursos reduzidos como processador e memória, mas é boa prática.

Coloca nos arrays de fala s_p e ruído s_{ruido} respetivamente a fala e ruidos de fundo e esporádico.

```
for i=1:1:length(sinal)
    if(((abs(sinal(i)- media_ruido))/desviop_ruido) > threshold_limit)
        voice(i) = 1;%#voiced
    else
        voice(i) = 0;%#unvoiced
    end
end

% 10 ms per frame sample with Fs defined
samples_per_frame = floor(Fs/100);
disp('Amostras por frame:');         %100 amostras por frame
disp(samples_per_frame);
number_samples = length(sinal) - mod(length(sinal), samples_per_frame);
n_frames = number_samples/samples_per_frame;
frame_voice_count = zeros(1, n_frames);
for i = 1:1:n_frames
    c_voiced = 0;
    for j = i*samples_per_frame-samples_per_frame+1:1:i*samples_per_frame
        c_voiced = c_voiced + voice(j);
    end
    frame_voice_count(i) =  floor(c_voiced/((samples_per_frame/2)+1)); %Given
end
% sinal_puro = [sinal_puro sinal(j)];
```

Figura 5: Função Retira Ruído 2/3

```
s_p =[];
s_ruido =[];
consecutive_frame_voice_count = 0;
holding_samples = 0;
for i = 1:1:n_frames
    if(frame_voice_count(i) == 1)
        consecutive_frame_voice_count = consecutive_frame_voice_count+1;
        if(consecutive_frame_voice_count >= 25)
            for j = (i-24+24*holding_samples)*samples_per_frame-samples_per_frame+1:1:i*samples_per_frame
                s_p = [s_p sinal(j)];
            end
            holding_samples  = 1;
        end
    else
        if(consecutive_frame_voice_count < 25)
            for j = (i-consecutive_frame_voice_count)*samples_per_frame-samples_per_frame+1:1:i*samples_per_frame
                s_ruido = [s_ruido sinal(j)];
            end
        end
        consecutive_frame_voice_count = 0;
        holding_samples = 0;
    end
end
```

Figura 6: Função Retira Ruído 3/3

A deteção de ruído esporádico é feita analizando o número consecutivo de frames vozeadas ultrapassar 24, passando a serem consideradas falas até receber o frame não vozeado.

Se for menos que 24, vai para o buffer de ruído, e o contador de frames sucessivas é reiniciado.

4.5 Função Filtro Wiener

```
function filtered=my_wiener(signal)

varn=var(signal(1:500));
for i=501:length(signal)
    vars=var(signal(i-500:i));
    means=mean(signal(i-500:i));
    filtered (i)=means+((vars-varn)/vars)*(signal(i)-means);
end
```

Figura 7: Função Filtro Wiener

Primeiramente é extraído a variância das primeiras 500 amostras, que são apenas ruído, σ_n^2.
Depois, são obtidas a variância, σ_s^2, e a média, m_x, de todo o sinal para filtrar o mesmo depois das primeiras 500 amostras, aplicando a equação descrita na última linha do ciclo for:

$$\widehat{s}(n) = m_x + \frac{\sigma_s^2 - \sigma_n^2}{\sigma_s^2}(x(n) - m_x) \tag{5}$$

Em que x(n) corresponde a uma dada amostra do sinal de entrada e $\widehat{s}(n)$ a essa amostra depois de passar pelo filtro wiener. Note-se que caso σ_s for muito maior que σ_n, $\widehat{s}(n) \simeq$ x(n). Se, em vez disso, σ_n for muito maior que σ_s, o filtro entra em ação, sendo que o sinal de saída é principalmente afetado por m_x. Ora, a média do ruído branco é zero, logo, o que resta é caraterística do sinal a ser transmitido.

5 Testes

Note que os testes efetuados foram com o mesmo sinal para ser coerente na criação da tabela. O sinal original que é mostrado é antes de contaminar (cima), contudo o mesmo foi contaminado e separado em fala (centro) ou ruído (baixo). Os próximos testes foram executados ao excerto sonoro: ("João", abre porta, "Pedro", fecha porta).

5.1 SNR = -10dBs

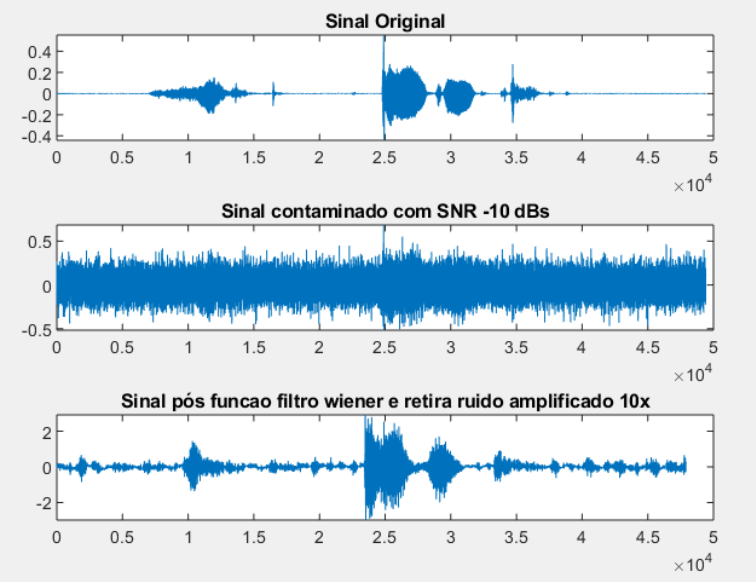

Figura 8: Teste a -10 dBs com threshold a 0.55

Conseguiu-se ouvir o sinal captado mesmo com um SNR negativo de -10 dBs, onde a amplitude do ruído supera a da fala. Teve de ser amplificado 10 vezes no fim de passar no filtro e função que lhe retirou os silêncios bem como o ruído esporádico do bater de uma porta... A ampliação consistiu em multiplicar por uma constante, neste caso 10, o sinal final.

5.2 SNR = 0dBs

Com 0 dBs mal se ouvia dado que só tinha aplicado o filtro Wiener para a situação de SNR negativo. Mesmo assim, apenas para um threshold de 0.55 é que se conseguia alguma eficiência na obtenção do sinal.
Abaixo desse threshold e o programa reportava erros de execução.

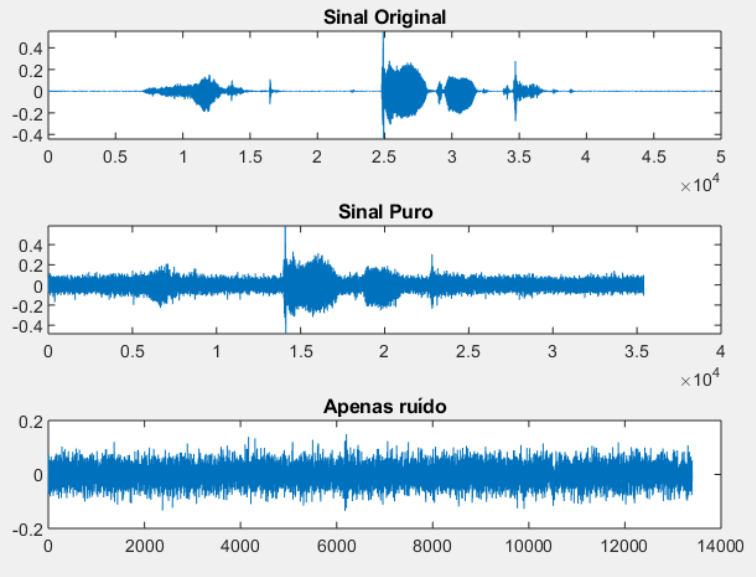

Figura 9: Teste a 0 dBs com melhor threshold, o da tabela

5.3 SNR = 5dBs

Com 5 dBs ouvia-se melhor que em 0, contudo, apenas se conseguia retirar os barulhos esporádicos para thresholds próximos de 0,6. Acima disso e só partes de palavras eram detetadas.

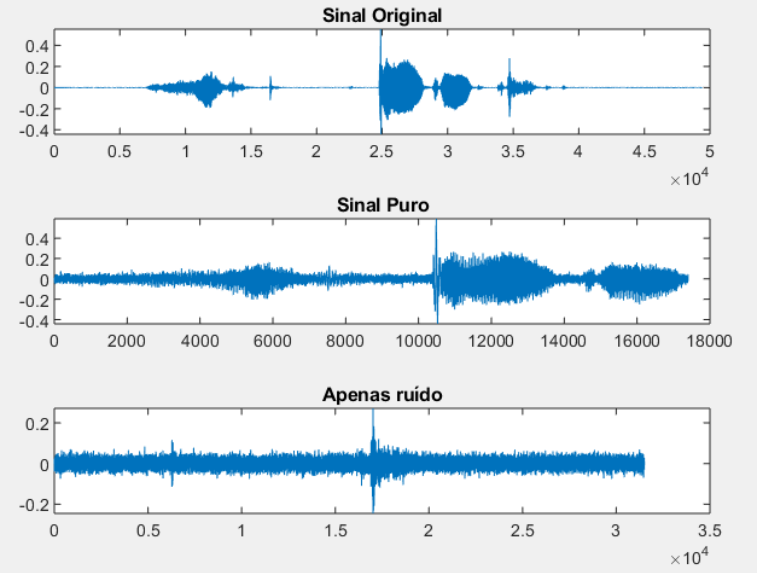

Figura 10: Teste a 5 dBs com melhor threshold, o da tabela

5.4 SNR = 10dBs

Com 10 dBs ouvia-se melhor que em 5, contudo, apenas se conseguia retirar os barulhos esporádicos para thresholds próximos de 0,8. Acima disso e só partes de palavras eram detetadas.

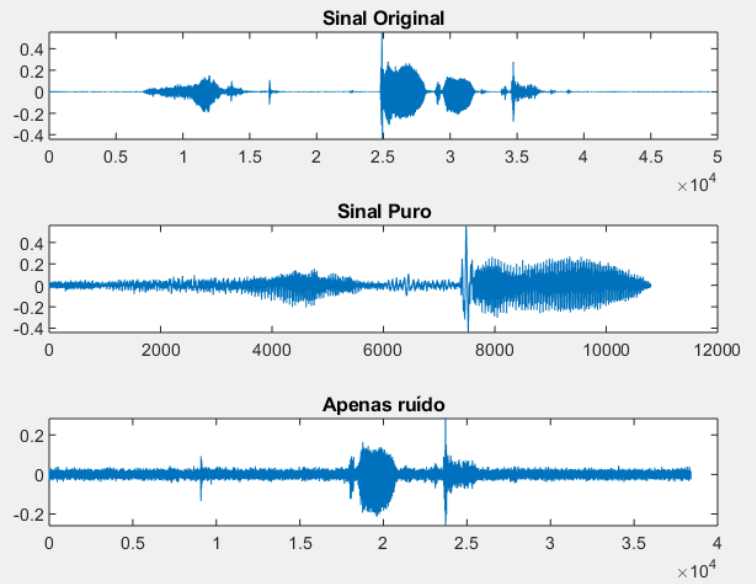

Figura 11: Teste a 10 dBs com melhor threshold, o da tabela

5.5 SNR = 15dBs

Com 15 dBs ouvia-se melhor que em 10, contudo, apenas se conseguia retirar os barulhos esporádicos para thresholds próximos de 1. Acima disso e só partes de palavras eram detetadas.

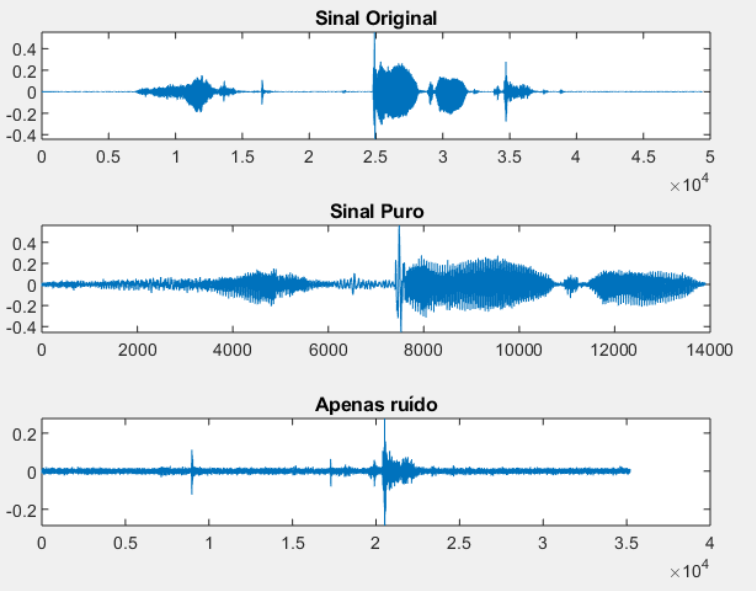

Figura 12: Teste a 15 dBs com melhor threshold, o da tabela

5.6 SNR = 20dBs

Com 20 dBs ouvia-se melhor que em 15, contudo, apenas se conseguia retirar os barulhos esporádicos para thresholds próximos de 1,5. Acima disso e só partes de palavras eram detetadas. Abaixo e captava-se muito ruído no vetor de fala.

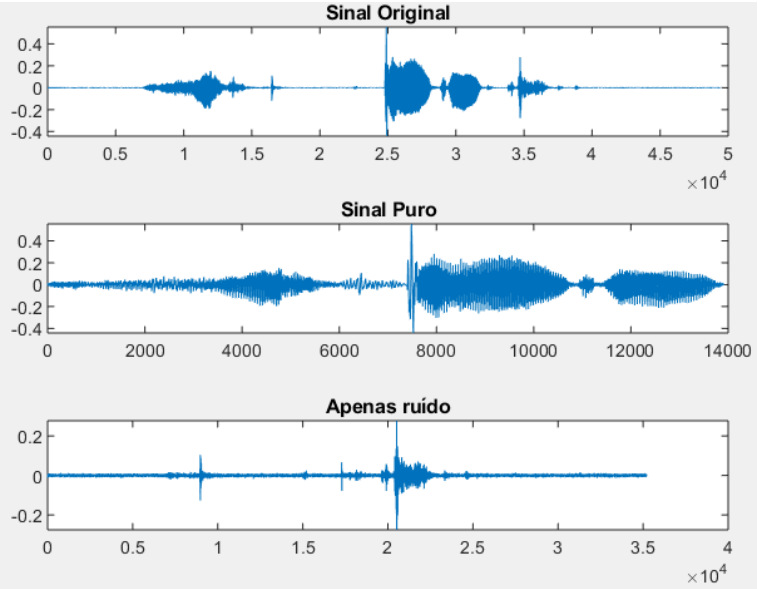

Figura 13: Teste a 20 dBs com melhor threshold, o da tabela

5.7 SNR = 25dBs

Com 25 dBs ouvia-se melhor que em 20, contudo, apenas se conseguia retirar os barulhos esporádicos para thresholds não muito afastados de 2. Acima disso e só partes de palavras eram detetadas. Abaixo e captava-se muito ruído no vetor de fala.
Nota-se um alargamento da margem de valores de threshold que poderiam ser usados para efeitos similares em torno de 2, relativamente a testes anteriores para outros thresholds ótimos correspondentes.

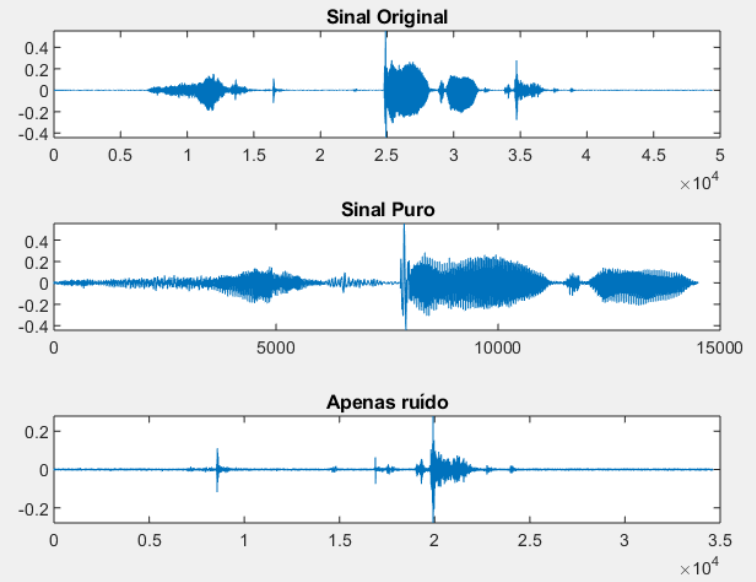

Figura 14: Teste a 25 dBs com melhor threshold, o da tabela

5.8 SNR = 30dBs

Com 30 dBs ouvia-se melhor que em 25, contudo, apenas se conseguia retirar os barulhos esporádicos para thresholds não muito afastados de 4,5. Acima disso e só partes de palavras eram detetadas. Abaixo e captava-se muito ruído no vetor de fala.

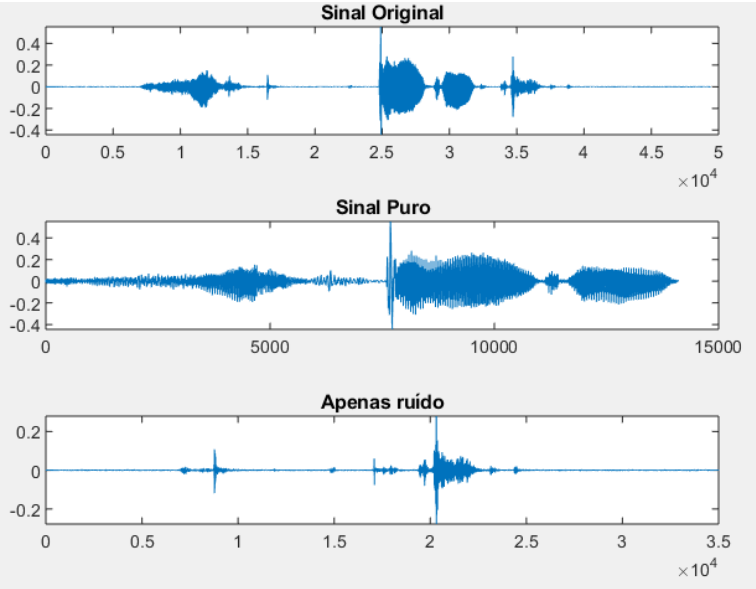

Figura 15: Teste a 30 dBs com melhor threshold, o da tabela

5.9 SNR = 35dBs

Com 35 dBs ouvia-se melhor que em 30, contudo, apenas se conseguia retirar os barulhos esporádicos para thresholds superiores a 5, sendo 6 o melhor threshold. Muito acima e apenas se notavam partes de palavras.

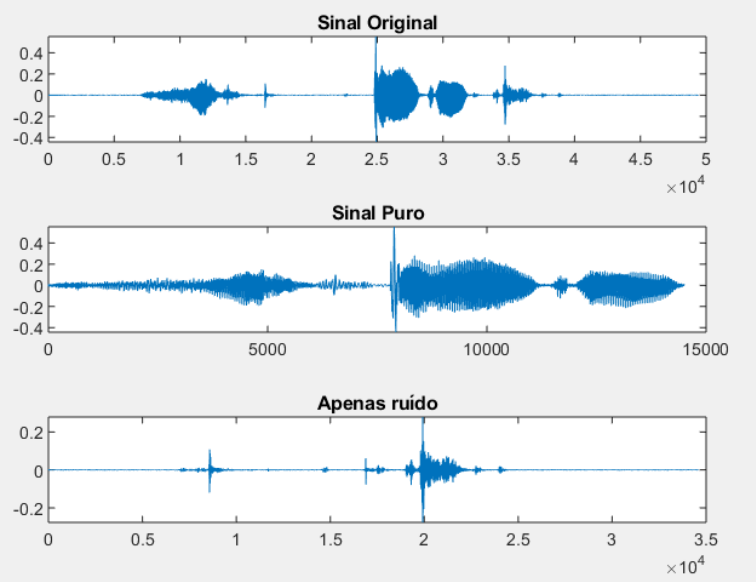

Figura 16: Teste a 35 dBs com melhor threshold, o da tabela

5.10 SNR = 40dBs

Com 40 dBs ouvia-se melhor que em 35, contudo, apenas se conseguia retirar os barulhos esporádicos para thresholds superiores a 5, sendo 12 o melhor threshold. Na primeira imagem podemos verificar que para 0,6 threshold, o programa pouco faz de útil, mas na figura abaixo ele claramente consegue fazer a distinção pretendida ruído fala.

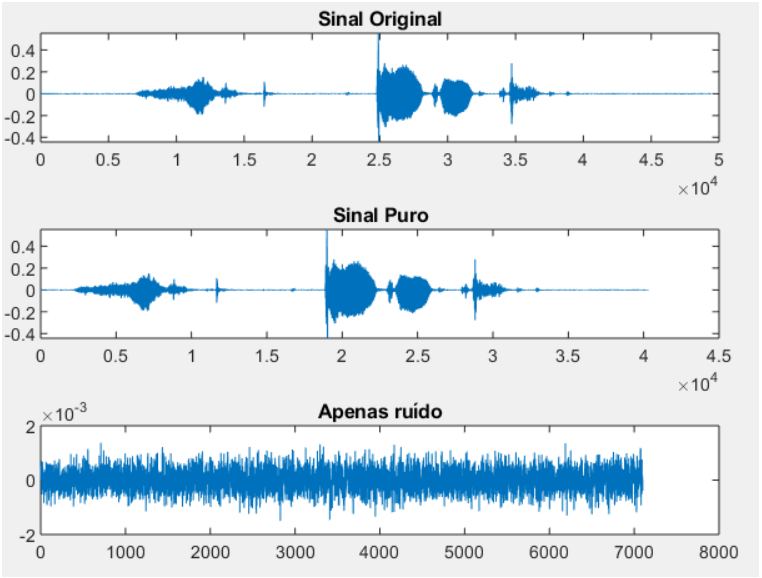

Figura 17: Teste a 40 dBs com o threshold de 0.55

6 Resultados: Tabela de ajuste automático de threshold

SNR	Threshold
0	0.55
5	0.6
10	0.8
15	1
20	1.5
25	2
30	4.5
35	6
40	12

Foi efetuada uma interpolação da tabelad de pares acima mencionada para determinação de thresholds de SNR's intermédios.
O algoritmo de threshold automático funciona conforme na imagem podemos verificar a separação de voz em ("teste", estalido, "teste", estalido, palma) dos restantes ruídos.

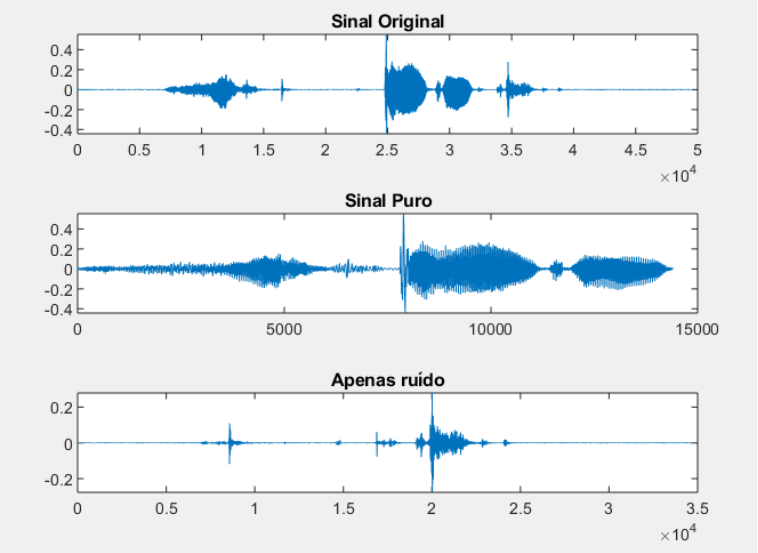

Figura 18: Teste a 40 dBs com melhor threshold, o da tabela (12 de threshold)

```
potencia_sinal = mean(sinal.^2);            %potencia do sinal
potencia_ruido = mean(sinal(1:(Fs/5)).^2);  %potencia do ruido, considerando que a primeira quinta

SNR = 10*log10(potencia_sinal/potencia_ruido); %calculo do SNR pela formula em dBs

disp('SNR:')
disp(SNR)

if (SNR <= 0)
    disp ('O ruido é em potência superior à fala.');
    threshold = 0.55;
elseif (SNR >= 0 && SNR <5)
    threshold = SNR*((0.6-0.55)/5)+0.55; %Calculo da reta do intervalo de 0 a 5
elseif (SNR >= 5 && SNR <10)
    threshold = SNR*((0.8-0.6)/5)+(0.6-((0.8-0.6)/5)*5); %Calculo da reta do intervalo de 5 a 10
elseif (SNR >=10 && SNR <15)
    threshold = SNR*((1-0.8)/5)+(0.8-((1-0.8)/5)*10); %Calculo da reta do intervalo de 10 a 15
elseif (SNR >=15 && SNR <20)
    threshold = SNR*((1.5-1)/5)+(1-((1.5-1)/5)*15); %Calculo da reta do intervalo de 15 a 20
elseif (SNR >=20 && SNR <25)
    threshold = SNR*((2-1.5)/5)+(1.5-((2-1.5)/5)*20); %Calculo da reta do intervalo de 20 a 25
elseif (SNR >=25 && SNR <30)
    threshold = SNR*((4.5-2)/5)+(2-((4.5-2)/5)*25); %Calculo da reta do intervalo de 25 a 30
elseif (SNR >=30 && SNR <35)
    threshold = SNR*((6-4.5)/5)+(4.5-((6-4.5)/5)*30); %Calculo da reta do intervalo de 30 a 35
elseif (SNR >=35 && SNR <40)
    threshold = SNR*((12-6)/5)+(6-((12-6)/5)*35); %Calculo da reta do intervalo de 35 a 40
elseif (SNR >=40)
    threshold = 12; %Calculo da reta para SNR superior ou igual a 40dBs
```

Figura 19: Função aplicação de threshold automático

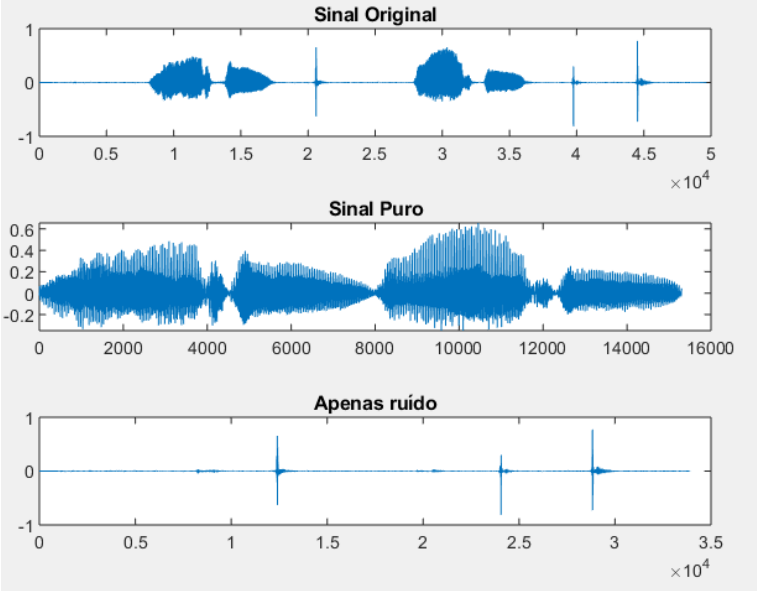

Figura 20: Sinal detetado de 28.3 dB atribuído automaticamente a um threshold de 3.68

7 Conclusão

Com este relatório conseguiu-se elaborar um algoritmo robusto de deteção e remoção de silêncios entre falas, ruído de fundo, ruído esporádico e ainda extração de sinal de fala, com qualidade aceitável, de sons com SNR negativos até -10 dBs.

Concluiu-se aquilo que fora já mencionado na secção do threshold, em que este aumenta com o aumento do SNR do sinal de entrada, dado que podemos alargar o nosso intervalo de confiança porque o peso do ruído passa a ser cada vez menor no sinal.

Além disso, cimentaram-se conceitos relacionados com a UC enunciados na abordagem teórica que, em conjunto com fundamentos de estatística, programação e teoria probabilística foram postos à prova numa tentativa (e eventual) resolução de um problema bem real onde, num mundo em que cada vez mais se efetuam transmissões de informação digitalmente, há equitativamente um aumento de interesse no estudo de técnicas e expansão de conhecimento na área de processamento digital de sinal.
Para acesso aos ficheiros de áudio, imagens e do matlab, consulte o repositório online:

`https://github.com/Joaovsky/PDS/tree/master/PDS_tp2`

8 Referências Bibliográficas

https://en.wikipedia.org/wiki/Additive_white_Gaussian_noise

https://en.wikipedia.org/wiki/Signal-to-noise_ratio

http://ganeshtiwaridotcomdotnp.blogspot.com/2011/08/silence-removal-and-end-point-d
html

Saha G., Chakroborty S., Senapati S., A New Silence Removal and Endpoint Detection Algorithm for Speech and Speaker Recognition Applications

El-Fattah, M. A., Dessouky, M. I., Diab S. M. and El-samie, F. E., speech enchancement using an adaptive wiener filtering aproach
Lima, C., slides