Universidade do Minho – Escola de Engenharia

Mestrado Integrado em Engenharia de Eletrónica Industrial e Computadores

# Algoritmo que permite a separação da fala do ruído

Processamento Digital de Sinal

2015/2016

**Docente: Carlos Lima**

**Discente: Ana Maria da Cunha Rodrigues nº68575**

# Índice

# Introdução

Este projeto foi desenvolvido no âmbito da unidade curricular de Processamento Digital de Sinal, sobre orientação do professor Carlos Lima e tem como objetivo implementar um algoritmo, em MATLAB, que permita eliminar o ruído de um sinal, através da análise do sinal de áudio original, otimizando, assim, o sinal, ficando apenas com a parte que interessa, a fala.

Num primeiro plano, será efetuado um breve resumo com a explicação de alguns conceitos teóricos fundamentais para uma melhor compreensão do problema proposto. E posteriormente, será apresentado o algoritmo desenvolvido, assim como a sua análise e demonstração através da realização de vários testes.

# Fundamentos teóricos

## Ruído branco

O ruído branco é um tipo de ruído produzido pela combinação simultânea de sons de todas as frequências. É por definição aquele que tem a sua potência distribuída uniformemente no espectro de frequência, ou seja, é uma constante. O nome ruído branco advém da analogia com o espectro eletromagnético na faixa de luz. A luz branca contém todas as frequências do espectro visível, daí o nome adotado.

Em estatística, um ruído branco, aplica-se a uma sequência de erros aleatórios, sempre que esta tiver média e variância constantes e sem auto-correlação. O valor da média, por conveniência, será zero, podendo no entanto especificar-se um outro valor. Assim, o ruído branco será temporalmente homogéneo, estacionário e não dependente de instantes anteriores.

Através do Modelo do Ruído, pelo cálculo da média e da variância de um sinal, conseguimos retirar a componente de ruído existente ficando, assim, apenas com o sinal "limpo".

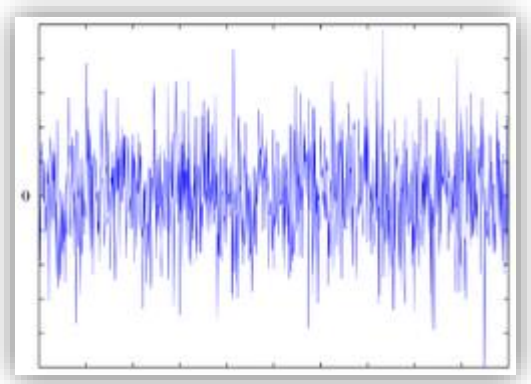


*Figura 1. Forma de onda de um ruído branco*

## *Média*

Em estatística, a média (μ) é o valor que aponta para onde mais se concentram os dados de uma distribuição. Pode ser considerada o ponto de equilíbrio das frequências, num histograma.

## *Variância*

Em estatística, a variância ($\sigma^2$) de uma variável aleatória é uma medida da sua dispersão estatística, indicando quão longe em geral os seus valores se encontram do valor esperado. O seu valor corresponde ao quadrado do desvio-padrão (σ).

## *Distribuição Gaussiana*

Uma distribuição normal ou gaussiana consiste num conjunto significativo de amostras de eventos aleatórios, cuja ocorrência individual não obedece a regras ou padrões que permitam fazer previsões acertadas, mas que, no entanto, a partir da análise do seu conjunto permitem concluir que os eventos tendem a ficar próximos de uma determinada posição, que representa a sua média matemática, μ.

É inteiramente descrita por dois parâmetros: a *média* (μ), valor para o qual mais se concentram os dados de uma distribuição, e o *desvio-padrão*(σ), que representa o valor de dispersão em relação à média. Assim, conhecendo-se estes valores consegue-se determinar qualquer probabilidade em uma distribuição Normal.

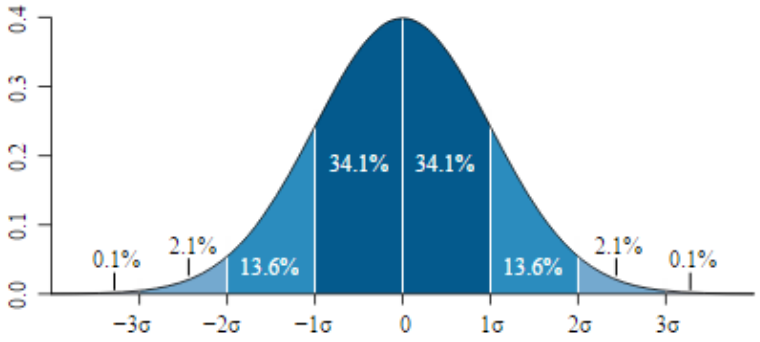



*Figura 2. Curva de distribuição normal*

A distribuição Gaussiana caracteriza-se por uma curva de distribuição de probabilidades simétrica, centrada na média, e como podemos visualizar na fig.2, 68,2% dos valores encontram-se a uma distância da média inferior a um desvio padrão.

Relacionando isto com o trabalho propriamente dito, pretende-se estabelecer um método de forma a poder ser percetível se uma parte do sinal é ou não ruído. Para isso, deve-se analisar uma parte do sinal que se tenha a certeza ser ruído, calcula-se o seu valor médio ($\mu$) e desvio Padrão ($\sigma$). Utilizando as propriedades da distribuição normal assume-se que se:

$$\mu - \sigma < V_R < \mu + \sigma$$

Trata-se de ruído. Assim, assume-se que a probabilidade de encontrar ruído é de 68,2% e a probabilidade de ser fala é de 31,8%

## *Signal-to-Noise Ratio (SNR)*

*Relação sinal-ruído* ou *razão sinal-ruído* é definido como a razão da potência de um sinal e a potência do ruído sobreposto ao sinal. Por outras palavras é uma medida que compara o nível de um sinal desejado, a fala, no nosso caso, com o nível do ruído de fundo. Quanto mais alta for a relação sinal-ruído, menor é o efeito do ruído de fundo sobre a deteção ou medição do sinal.

Um sinal que contenha mais ruído do que o sinal desejado terá um *SNR* baixo, enquanto um que tenha menos ruído terá um *SNR* alto.

$$SNR = \frac{Psinal}{Pruído}$$

Uma definição alternativa do *SNR* é a reciprocidade ao coeficiente de variação, isto é, a razão da média ($\mu$) e do desvio Padrão ($\sigma$) de uma medida de sinal.

$$SNR = \frac{\mu}{\sigma}$$

## *Algoritmo desenvolvido*

Para a realização deste trabalho desenvolveu-se a função `detetor_fala()` que percorre a variável que contém o sinal de áudio, passando os segmentos de amostras que são consideradas como fala para a variável `fala` e os intervalos de silêncio e ruído para a variável `ruido`. Caso existam segmentos de fala na variável `ruido`, é necessário ajustar os parâmetros que a função recebe. A seguir, encontra-se representado uma explicação detalhada desta função.

Os parâmetros que a função recebe são:

- `fich_audio`: vetor que contém o sinal de áudio a analisar;
- `noise_len`: número de amostras desde o início do vetor `fich_audio` que correspondem a ruído;
- `pause_len`: número de amostras permitido entre palavras num discurso, normalmente 50;
- `espor_noise_len`: número máximo de amostras que um ruido esporádico pode tomar.

A função retorna os argumentos:

- `fala`: vetor que contém apenas os segmentos de fala;
- `ruido`: vetor que contém os segmentos de ruído .

No início da função inicializam-se os vetores de retorno e as variáveis necessárias a 0.

```
ruido=0;
fala=0;

con_espor=0;
con_pausa=0;
n_amostras=0;
```

As variáveis `con_espor`, `con_pausa` e `n_amostras` são usadas como contadores. A variável `con_espor` conta o número de amostras para saber se o sinal obtido é esporádico ou não. A variável `con_pausa` conta o número de amostras para saber se o sinal obtido é uma pausa entre as falas ou não. Por último, a variável `n_amostras` é auxiliar para a contagem de amostras quando percorre-se o sinal pretendido .

Posteriormente às inicializações, procedeu-se ao cálculo do SNR, relação sinal-ruído. Utilizamos este valor para estudarmos qual o melhor valor de threshold para SNR, sendo o threshold um limite, que indica qual é a probabilidade de encontrar ruido ou fala, assumindo uma distribuição gaussiana.

```
SNR=(mean(fich_audio))/(std(fich_audio));

threshold=SNR*10;
if abs(threshold) < 1
    threshold=2;
end
```

Assumindo que as primeiras amostras são compostas de ausência de fala, ou seja, apenas se gravou ruído, calcula-se o valor médio e o desvio-padrão deste "sub-sinal",de modo a de seguida obtermos um modelo do ruído presente no sinal áudio. O próximo passo será então analisar o vetor com o sinal gravado e verificar se o valor será considerado ruido ou fala.

```
media=mean(fich_audio(1:noise_len));
dp=std(fich_audio(1:noise_len));
```

Após todas as inicializações e cálculos, passou-se para ao algoritmo, onde foi criado um ciclo que percorre o ficheiro de áudio e avalia-se o valor do sinal trata-se de um valor considerado fala. Caso seja fala avalia-se se o contador do ruído esporádico é maior que o valor recebido como parâmetro, se for, conta-se como um segmento de fala e a informação é concatenada para o vetor `fala`, caso não seja é removido. O mesmo acontece para a pausa entre palavras, se o contador da pausa for maior que o valor passado por parâmetro, assume-se que é ruído e, neste caso, a informação é concatenada para o vetor `ruido`.

```
    for i=1:length(fich_audio)
        if abs(fich_audio(i)) > (media + abs(threshold)*dp)
           con_espor = con_espor+1;
           n_amostras = n_amostras+1;
           con_pausa=0;
           if(con_espor) > espor_noise_len
               fala=cat(1,fala,fich_audio(i-n_amostras+1:i));
               n_amostras=0;
           end
        else
            con_pausa=con_pausa+1;
            n_amostras = n_amostras+1;
            if con_pausa > pause_len
                ruido=cat(1,ruido,fich_audio(i-n_amostras+1:i));
                con_espor=0;
                n_amostras=0;
                con_pausa=0;
            end
        end
    end
end
```

## Testes

De seguida serão mostrados todos os teste e resultados obtidos usando a função `detetor_fala()`.

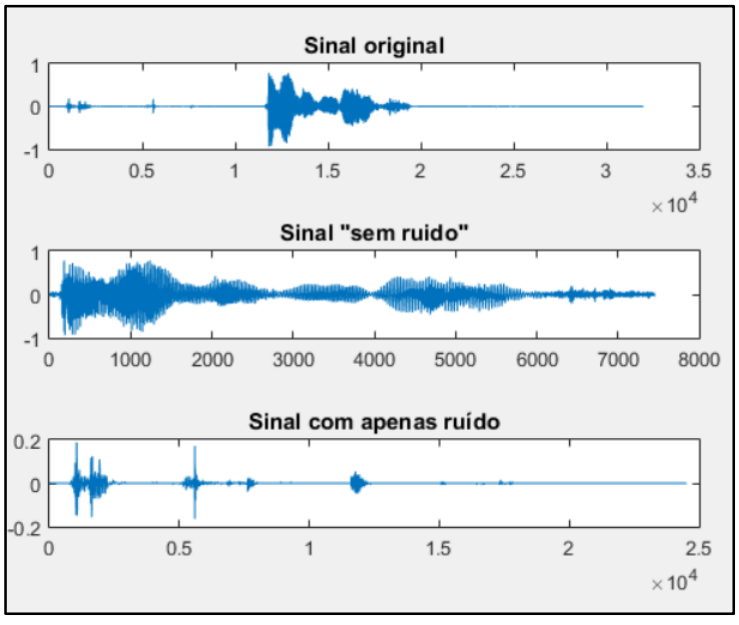
- ### 1º Teste

**Amostra**: "Ana Rodrigues"

Comandos a executar na linha de comandos do *Matlab*:

```
>> x=audiorecorder;
>> recordblocking(x,4);
>> play(x);
>> fich_audio=getaudiodata(x);
>> sound(fich_audio);
>> [fala,ruido]=detetor_fala(fich_audio,5000,50,250);
>> subplot(3,1,1); plot(fich_audio);
>> title('Sinal original');
>> subplot(3,1,2); plot(fala);
>> title('Sinal "sem ruido"');)
>> subplot(3,1,3); plot(ruido);
>> title('Sinal com apenas ruído');
>> sound(fala);
>> sound(ruido);
```

Neste teste, foi gravado, num ambiente sem ruido, a seguinte expressão "Ana Rodrigues", podendo visualizar o sinal gravado, assim como, os sinais resultantes, no esquema abaixo representado.




Como podemos verificar, o resultado obtido com esta função é o pretendido, pois é eliminado algum ruído existente.

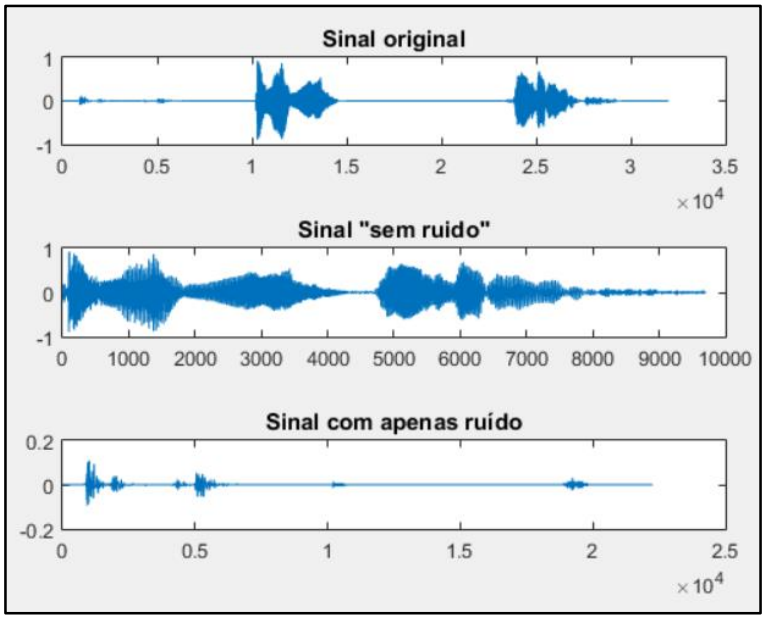
- *2º Teste*

**Amostra**: "Ana" + pausa + "Rodrigues"

Comandos a executar na linha de comandos do *Matlab*:

```
>> x=audiorecorder;
>> recordblocking(x,4);
>> play(x);
>> fich_audio=getaudiodata(x);
>> sound(fich_audio);
>> [fala,ruido]=detetor_fala(fich_audio,5000,50,600);
>> subplot(3,1,1); plot(fich_audio);
>> title('Sinal original');
>> subplot(3,1,2); plot(fala);
>> title('Sinal "sem ruido"';)
>> subplot(3,1,3); plot(ruido);
>> title('Sinal com apenas ruído');
>> sound(fala);
>> sound(ruido);
```

Neste teste, foi gravado, num ambiente sem ruido, a seguinte expressão "Ana Rodrigues" com uma pausa entre as palavras, de modo a perceber se a função reduz o espaço entre palavras para o valor passado por parâmetro. Pode-se visualizar o sinal gravado, assim como, os sinais resultantes, no esquema abaixo representado.




Como podemos verificar, o algoritmo está a funcionar pois é eliminada a pausa entre as palavras, assim como, algum ruído existente.

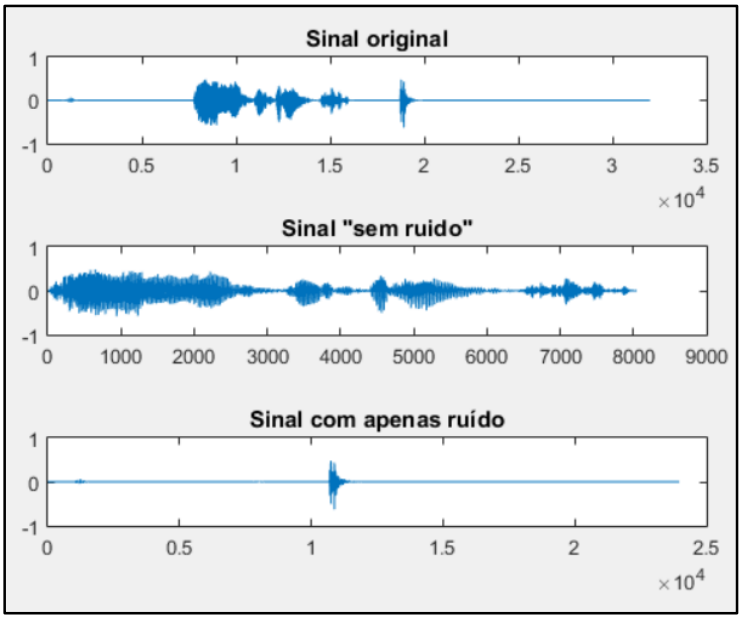
- *3º Teste*

**Amostra**: "Ana Rodrigues" + objeto a cair

Comandos a executar na linha de comandos do *Matlab*:

```
>> x=audiorecorder;
>> recordblocking(x,4);
>> play(x);
>> fich_audio=getaudiodata(x);
>> sound(fich_audio);
>> [fala,ruido]=detetor_fala(fich_audio,5000,50,1150);
>> subplot(3,1,1); plot(fich_audio);
>> title('Sinal original');
>> subplot(3,1,2); plot(fala);
>> title('Sinal "sem ruido"';)
>> subplot(3,1,3); plot(ruido);
>> title('Sinal com apenas ruído');
>> sound(fala);
>> sound(ruido);
```

Neste teste, foi gravado a seguinte expressão "Ana Rodrigues" + ruido esporádico. Um ruído esporádico é um som ou barulho de curta duração com valores que se assemelham ao de uma palavra, como por exemplo um objeto a cair. No esquema abaixo é possível visualizar o sinal gravado, assim como, os sinais resultantes.




Como podemos verificar, o resultado obtido com esta função é o pretendido, pois o ruido foi eliminado.

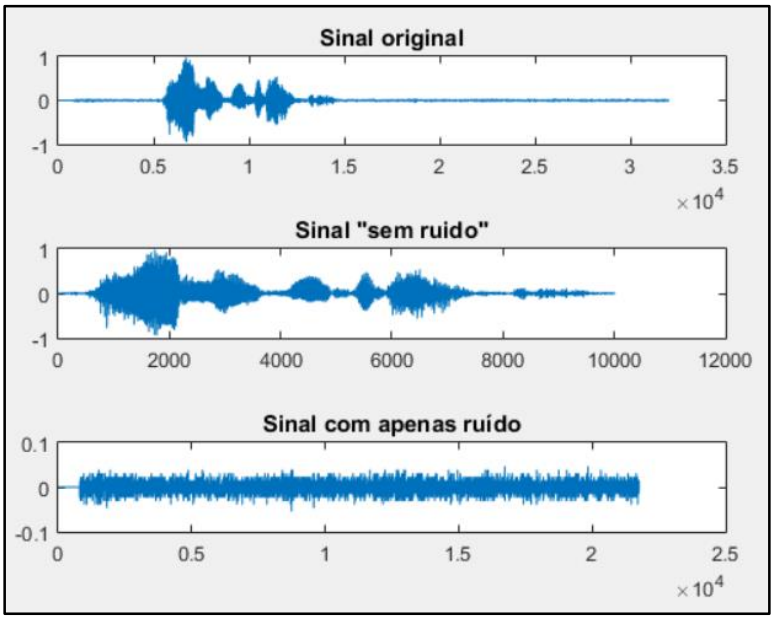
- *4º Teste*

**Amostra**: "Ana Rodrigues" + som do secador

Comandos a executar na linha de comandos do *Matlab*:

```
>> x=audiorecorder;
>> recordblocking(x,4);
>> play(x);
>> fich_audio=getaudiodata(x);
>> sound(fich_audio);
>> [fala,ruido]=detetor_fala(fich_audio,5000,50,1150);
>> subplot(3,1,1); plot(fich_audio);
>> title('Sinal original');
>> subplot(3,1,2); plot(fala);
>> title('Sinal "sem ruido"';)
>> subplot(3,1,3); plot(ruido);
>> title('Sinal com apenas ruído');
>> sound(fala);
>> sound(ruido);
```

Neste teste, foi gravado a seguinte expressão "Ana Rodrigues" + ruido de fundo, sendo que foi utilizado um secador como ruido de fundo. No esquema abaixo é possível visualizar o sinal gravado, assim como, os sinais resultantes.




Como podemos verificar, o resultado obtido com esta função é o pretendido, embora alguma parte do ruído ainda aparecer no sinal "sem ruído".

## *Conclusão*

O trabalho realizado correu bem visto que no final foi atingido o objetivo traçado. Com recurso a simples processos estatísticos e a uma função do *MATLAB*, foi possível retirar as partes indesejadas a um sinal de áudio de forma e reduzir ao seu tamanho e ficar apenas com o que interessa ao utilizador.

Logo à partida surge como vantagem neste processo, o facto de poupar memória visto que foi truncado o sinal. Para além disso, é muito mais agradável de trabalhar com um sinal que tem apenas o necessário e não ser preciso andar a procura do conteúdo que verdadeiramente interessa.

Após a finalização de trabalho importa realçar também que a sua implementação permitiu adquirir conhecimentos acerca da ferramenta de desenvolvimento usada, *MATLAB.*

## Anexo

```
function [fala,ruido] = detetor_fala(fich_audio, noise_len, pause_len,
espor_noise_len)
% detetor_fala -> função que analisa o sinal de áudio e separa o ruido
% da fala através da criação de 2 vetores
% Argumentos de entrada:
%   fich_audio: vetor com o sinal áudio a analisar
%   noise_len: número de amostras iniciais onde se grava ruído
%   pause_len: duração do tempo de pausa permitido no segmento de voz
%   (amostras), normalmente 50
%   espor_noise_len: número máximo de amostras que um ruido esporádico
pode tomar
% Argumentos de saída:
%   ruido: vetor que guarda o segmento de ruido (ruido removido)
%   fala: vetor que guarda o segmento de fala sem ruido
    %Inicializações
    ruido=0;            %vetor segmento de ruido inicializado a 0
    fala=0;             %vetor segmento de fala inicializado a 0
    con_espor=0;         %variável contador do número de amostras para
                         %saber se o sinal obtido é esporádico ou não
   con_pausa=0;         %variável counter do num de amostras para saber
                         %se o sinal é uma pausa entre as falas ou ñ
    n_amostras=0;        %variável auxiliar contador do num de amostras

    SNR=(mean(fich_audio))/(std(fich_audio)); %SNR=media/desvio_padrao
    threshold=SNR*10;          %threshold calcula um limite, que indica
    if abs(threshold) < 1      %qual é a probabilidade de encontrar
        threshold=2;          %ruido ou fala
    end

    media=mean(fich_audio(1:noise_len));     %calculo da média
    dp=std(fich_audio(1:noise_len));         %calculo do desvio-padrão
                                            %do ruído inicial
    %Algoritmo da função
    for i=1:length(fich_audio) %percorre o fich de áudio e compara-se
        if abs(fich_audio(i)) > (media + abs(threshold)*dp)
           con_espor = con_espor+1;        %se o valor do sinal é
           n_amostras = n_amostras+1;      %maior ao limite calculado
           con_pausa=0;                    %é um segmento de fala
           if(con_espor) > espor_noise_len
               %se o contador de amostras de ruido esporádico for
               %maior que o valor recebido por parâmetro conta-se como
               %um segmento de fala, senão é removido
               fala=cat(1,fala,fich_audio(i-n_amostras+1:i));
               n_amostras=0;
           end
        else                                %senão é um segmento de ruido
            con_pausa=con_pausa+1;
            n_amostras = n_amostras+1;
            if con_pausa > pause_len
            %se o contador da pausa for maior que o valor recebido por
            %parâmetro assume-se que é ruido
                ruido=cat(1,ruido,fich_audio(i-n_amostras+1:i));
                con_espor=0;
                n_amostras=0;
                con_pausa=0;
            end
        end
    end
end
```