Processamento Digital de Sinal

Remoção de ruído de fundo e ruído esporádico

Relatório de Trabalho

Miguel Silva
68552

Índice

Introdução

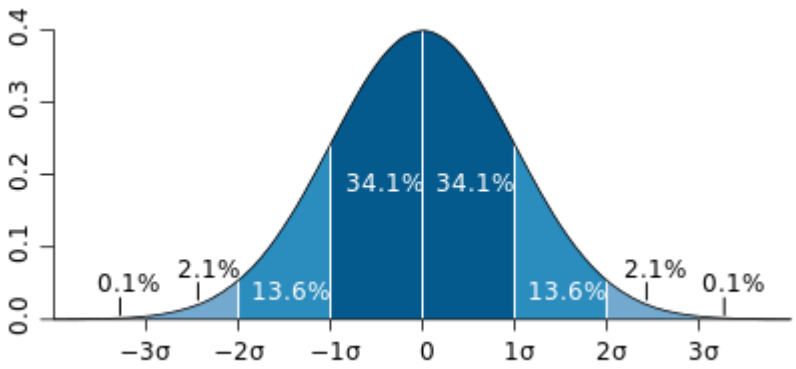
Para a disciplina de Processamento Digital de Sinal (PDS) foi requerido um trabalho que consistia na remoção de ruído numa gravação de uma fala, optimizando assim o sinal, dando relevância apenas aos dados úteis.

De modo a resolver este problema, recorreu-se ao MATLAB para a gravação de um discurso num ficheiro de som e futura computação do ruído.

Assumindo que o ruído segue uma distribuição Gaussiana, fez-se um estudo do modelo de ruído para que possa removido da gravação original, dando origem a dois novos ficheiros de áudio, um com o som filtrado e outro com o ruído apenas.

Fundamentos Teóricos

Distribuição Gaussiana



Esta distribuição, também conhecida como distribuição normal, é das mais importantes em toda a estatística devido ao teorema central do limite que afirma que quando o tamanho da amostra aumenta, a distribuição amostral da sua média aproxima-se cada vez mais de uma distribuição Gaussiana.

Tal como na imagem acima, para qualquer distribuição normal, a probabilidade de encontrar um valor com diferença de um desvio padrão da média é de 68,2%. Logo, para este trabalho, pode assumir-se que a probabilidade de encontrar ruído é de 68,2% e a probabilidade de encontrar uma fala é 100-68,2=31,8%.

Signal-to-Noise Ratio (SNR)

É definido como a relação da potencia de um sinal com informação útil, no nosso caso, a fala, com a potencia de um ruído de fundo não pretendido.

Uma definição alternativa do SNR é a reciprocidade ao coeficiente de variação, isto é, a razão da média e do desvio padrão de uma medida de um sinal.

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

Explicação do código

Para a realização deste trabalho recorreu-se apenas ao uso de uma função desenvolvida especificamente para este trabalho, chamada `noise_remover()`, que irá ser explicada a seguir.

Esta recebe como parâmetros as seguintes variáveis:

- `file` : vector com o sinal de áudio. Gravado com a função `wavrecord()` do MATLAB;
- `fs`: frequência de amostragem do sinal;
- `nlen`: numero de amostras iniciais onde apenas se grava ruído (sem informação útil);
- `spause`: número de amostras entre duas palavras durante um discurso de uma pessoa (geralmente 50) e;
- `espnoise`: número máximo de amostras que um ruído esporádico pode tomar.

E retorna como resultado os seguintes vectores:

- `sbuf`: sinal resultante sem ruído e;
- `nbuf`: ruído removido.

No inicio da função inicializam-se os vectores de retorno a 0 bem como as variáveis `espor`, `pause` e `counter` também a 0. As últimas 3 são apenas usadas como contadores. A variável `espor` conta o número de amostras de modo a conseguir verificar se o que se está a avaliar é um sinal esporádico ou não. A variável `pause` tem um uso semelhante só que é para a verificar se é uma pausa entre palavras ou não. A variável `counter` é uma auxiliar para a contagem de amostras quando se percorre o sinal recebido.

```
espor=0;
pause=0;
counter=0;
sbuf=0;
nbuf=0;
```

Após isto calcula-se o SNR com a formula apresentada no capítulo anterior.

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

Depois calcula-se um limite ao qual é chamado de `threshold`, que nos vai indicar qual é a prababilidade de encontrar ruído ou fala, assumindo uma distribuição gaussiana.

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

Assumindo que nas primeiras amostras apenas se gravou ruído, calcula-se a média e o desvio padrão deste “sub-sinal” que contem apenas informação não desejada. E calcula-se o valor a partir do qual se pode considerar que o sinal é fala e não ruído.

```
    m=mean(file(1:nlen));
    s=std(file(1:nlen));

    aux = m + abs(threshold) * s;
```

Após todas estas inicializações e cálculos, passou-se para a verdadeira computação do sinal, onde se percorre o ficheiro de áudio do inicio ao fim e se avalia se o valor do sinal é maior é maior que o limite calculado anteriormente. Se for maior, trata-se essa amostra como pertencente a uma palavra da fala, caso não seja trata-se como ruído. Caso seja fala, avalia-se sempre se o contador do ruído esporádico é maior que o valor recebido como parâmetro, se for, conta-se como uma palavra, 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.

```
    for i=1:length(file)
        if abs(file(i)) > aux
            espor=espor+1;
            counter=counter+1;
            pause=0;
            if espor > espnoise
                sbuf=cat(1,sbuf,file(i-counter+1:i));
                counter=0;
            end
        else
            pause=pause+1;
            counter=counter+1;
            if pause > spause
                nbuf=cat(1,nbuf,file(i-counter+1:i));
                espor=0;
                counter=0;
                pause=0;
            end
        end
    end
end
```

Para além de passar os parâmetros correctos para a função, faz-se previamente a obtenção de um sinal de áudio com a função `wavrecord()`, e recorre-se à função `plot()` para determinar, aproximadamente o numero de amostras iniciais do ruído. Com o uso da função `sound()`, é possível verificar os resultados da função criada.

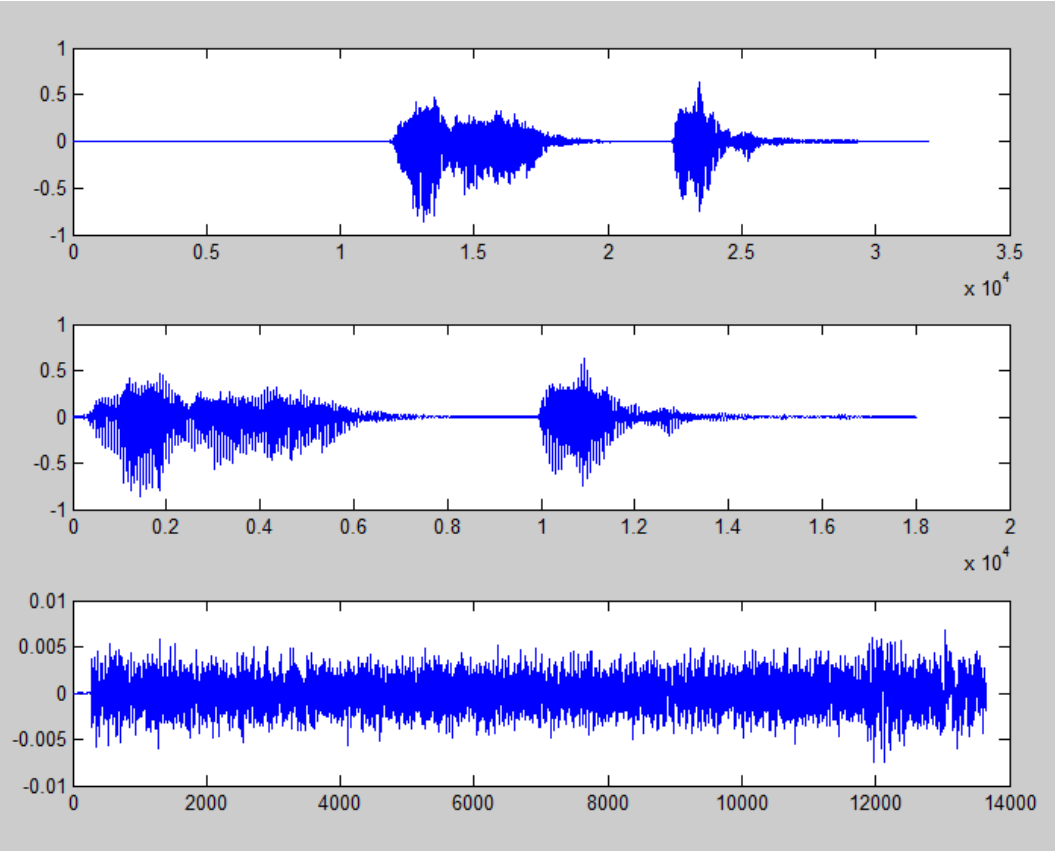
Testes

De modo a testar a função criada, tal como explicado acima, fez-se `wavrecord()` de alguns sinais, seguidos de `plot()` para perceber quais seriam os parâmetros correctos a passar à função `noise_remover()`. Em todos os testes usaram-se as seguintes palavras: "Miguel Silva", sem ruído, com diferentes ruídos de fundo, apenas com ruído esporádico e com ruído de fundo e esporádico.

Em todas as imagens que se seguem, o primeiro gráfico, é o som gravado, o segundo é o som resultante sem ruído e o ultimo é apenas o ruído removido.

Sem ruído

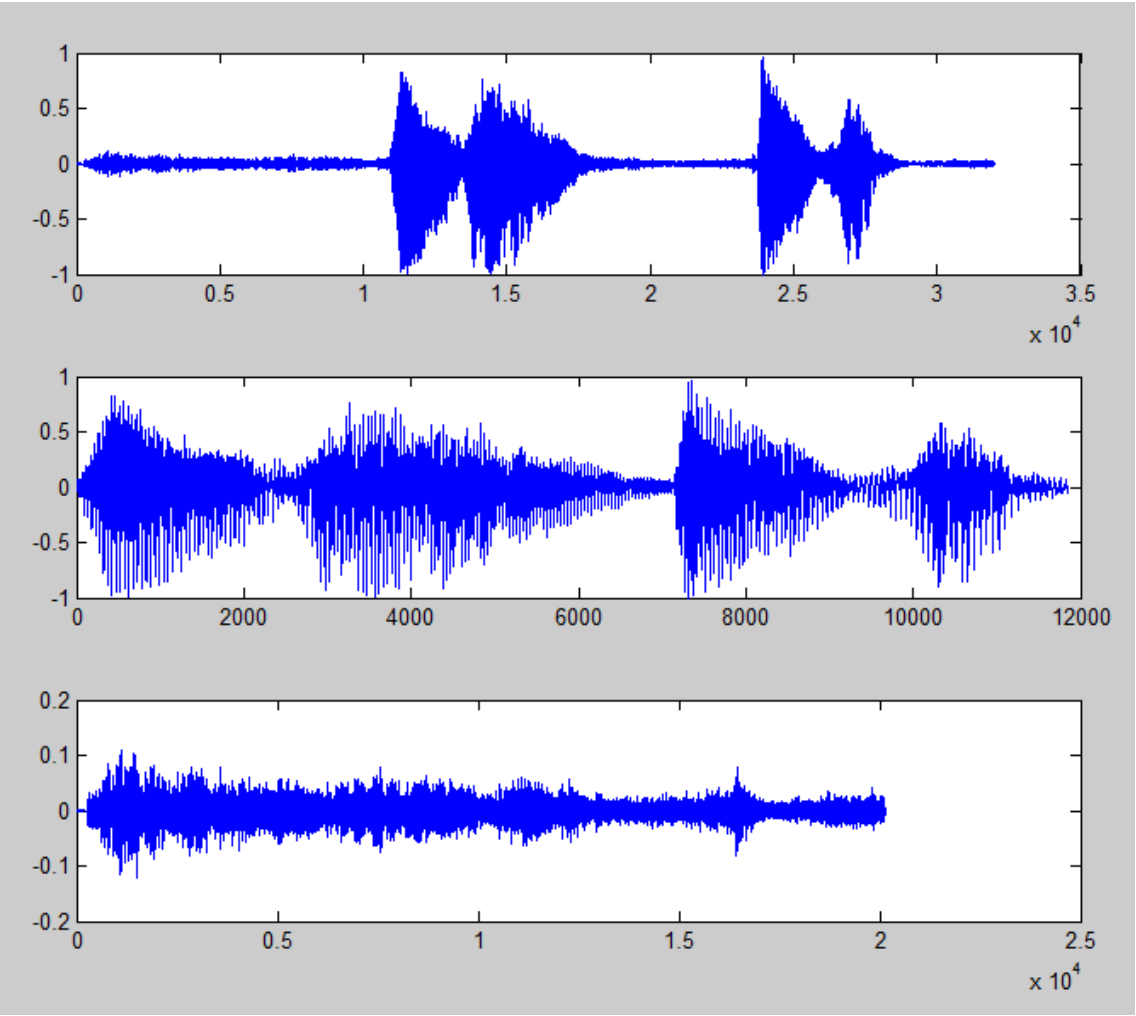
A imagem abaixo demonstra o teste realizado sem ruído. Para facilitar a visualização dos resultados, na gravação do som, deu-se um espaço entre ambas as palavras, de modo a perceber que a função reduz o espaço entre palavras para o valor passado por parâmetro, neste caso para 50 amostras.

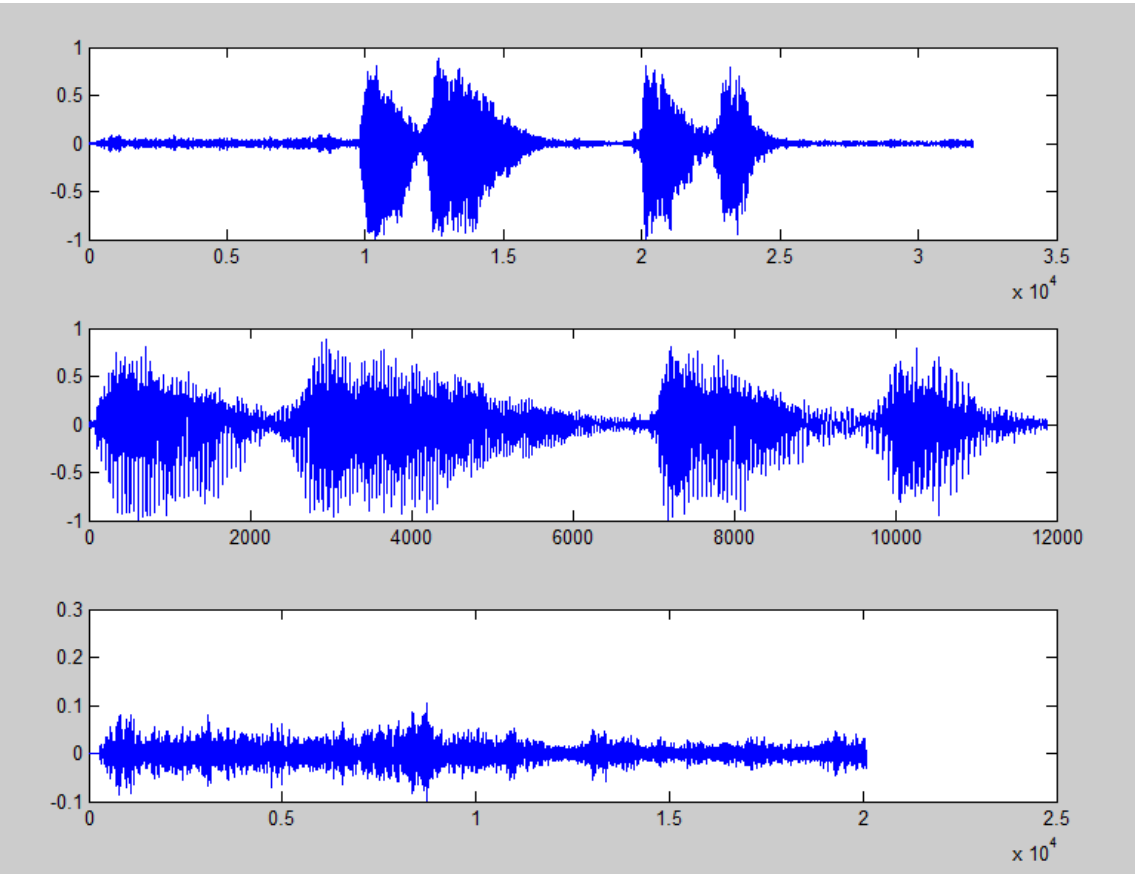


Ruído de fundo

Aqui usaram-se dois ruídos diferentes. Na primeira imagem usou-se como ruído de fundo, um aspirador doméstico e na segunda usou-se o barulho da chuva.

Em ambos o resultado foi o esperado, ambas as palavras sem ruído.

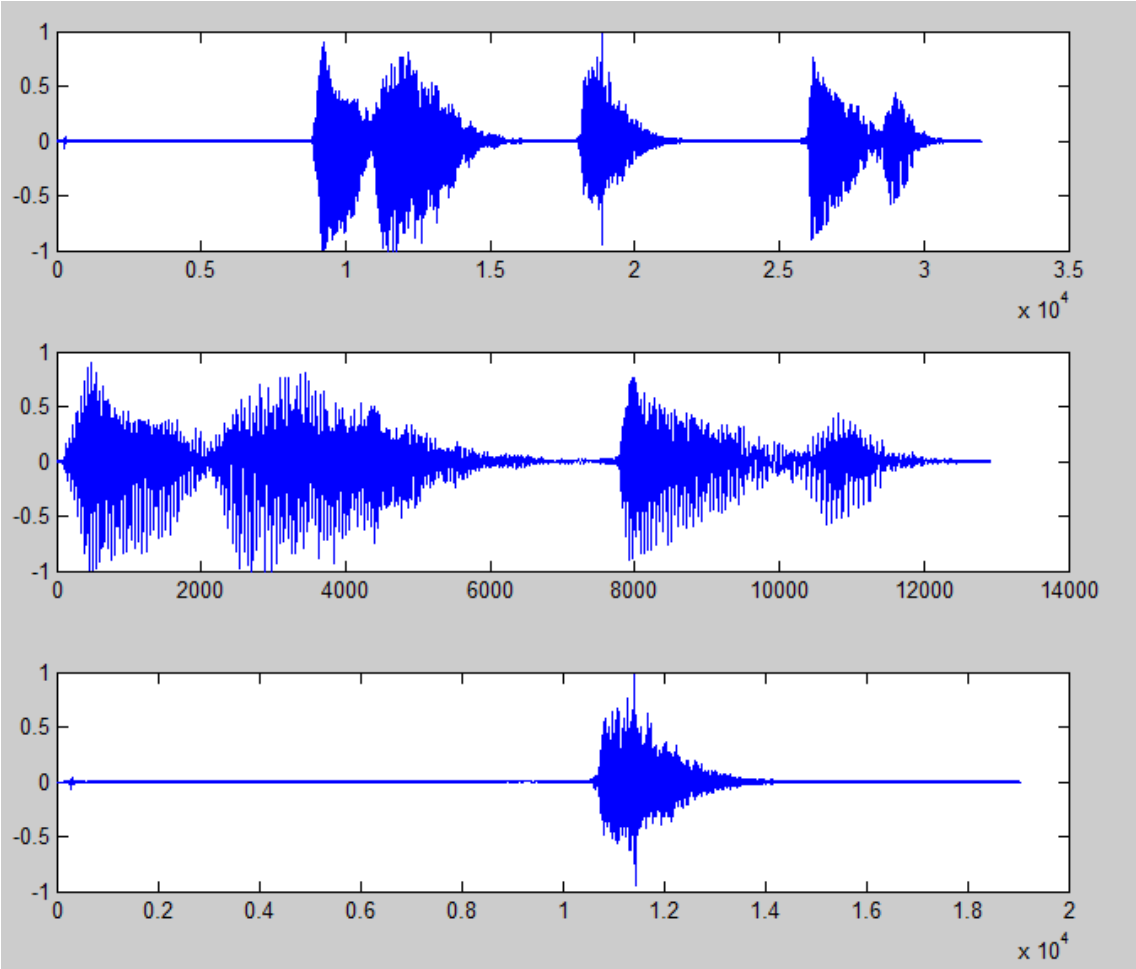

1
0.5
0
-0.5
-1
0
0.5
1
1.5
2
2.5
3
3.5
x 10^4
1
0.5
0
-0.5
-1
0
2000
4000
6000
8000
10000
12000
0.2
0.1
0
-0.1
-0.2
0
0.5
1
1.5
2
2.5
x 10^4


1
0.5
0
-0.5
-1
0
0.5
1
1.5
2
2.5
3
3.5
x 10^4
1
0.5
0
-0.5
-1
0
2000
4000
6000
8000
10000
12000
0.3
0.2
0.1
0
-0.1
0
0.5
1
1.5
2
2.5
x 10^4

Ruído Esporádico

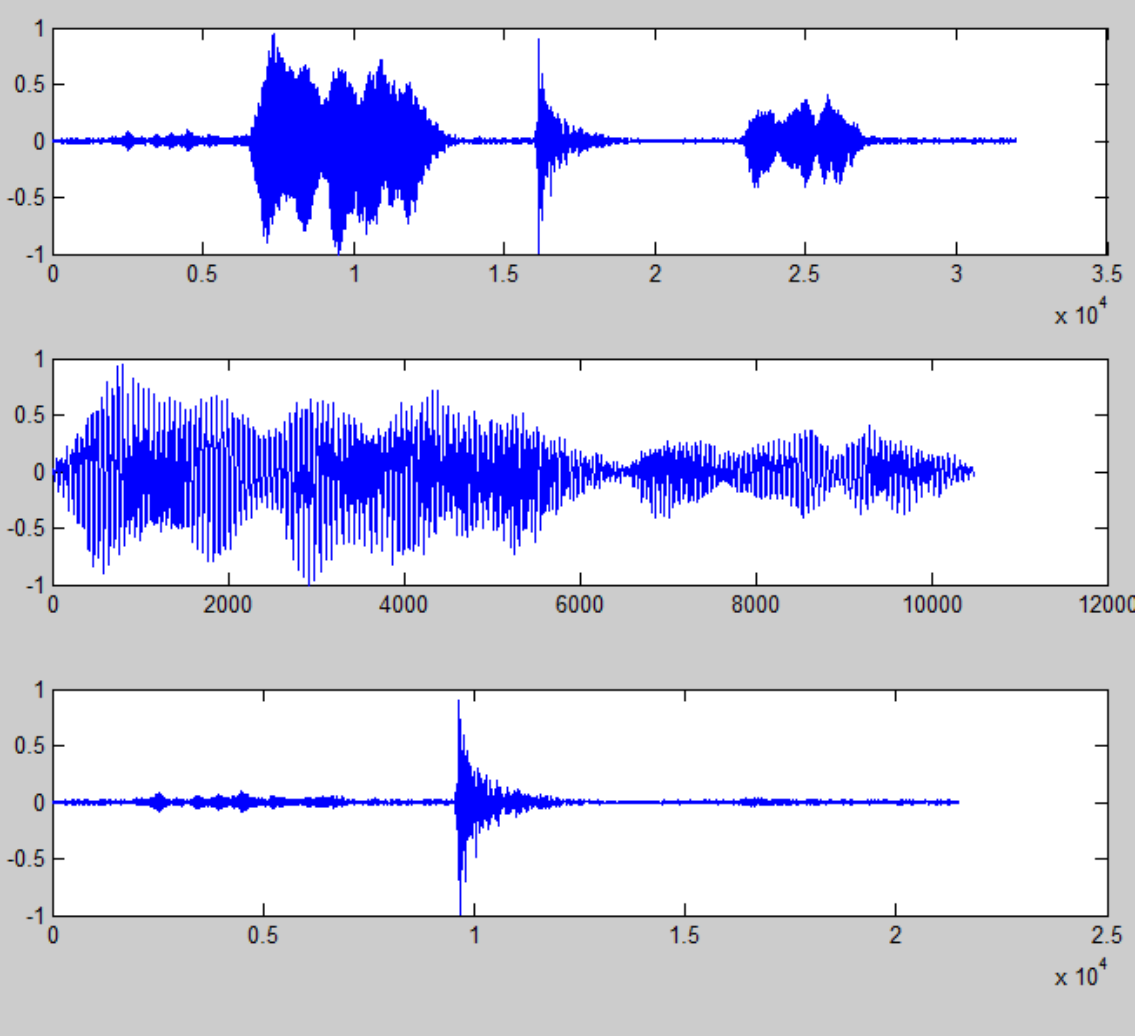
Um ruído esporádico é um som ou um barulho de curta duração com valores que se aproximam ao de uma palavra, como por exemplo o cair de algo ou o bater uma palma.

No seguinte teste usou-se um barulho destes entre ambas as palavras, e o resultado obtido foi positivo, apenas as palavras da fala, podendo observar-se o ruído removido no terceiro gráfico.



Ruído de fundo e ruído esporádico

Num último teste, juntou-se o barulho do vento no fundo e o bater de um telemóvel na mesa. Mais uma vez os resultados obtidos foram bons tal como se pode ver na próxima imagem.


1
0.5
0
-0.5
-1
0
0.5
1
1.5
2
2.5
3
3.5
x 10^4
1
0.5
0
-0.5
-1
0
2000
4000
6000
8000
10000
12000
1
0.5
0
-0.5
-1
0
0.5
1
1.5
2
2.5
x 10^4

Conclusão

Após a finalização de trabalho é importante referir que foi possível adquirir conhecimentos na ferramenta de desenvolvimento MATLAB que não foi muito abordada até ao momento no decorrer do curso.

Superadas as dificuldades deste IDE diferente do qual se está habituado, foi necessário superar as dificuldades do tema, que apesar de muito interessante, tem um grau de dificuldade acrescido devido ao nível de abstracção que é necessário ter para se poder analisar um som.

No entanto, todas as dificuldades foram superadas até que se obteve os resultados acima descritos, ajudando a perceber o tratamento que levam os sons que ouvimos provenientes de emissores para as massas, rádio e televisão por exemplo.

Anexos

Função noise_remover()

```
function [sbuf, nbuf] = noise_remover( file, fs, nlen, spause,
espnoise )

    espor=0;
    counter=0;
    pause=0;
    sbuf=0;
    nbuf=0;

    SNR=(mean(file)/std(file));

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

    m=mean(file(1:nlen));
    s=std(file(1:nlen));

    aux = m + abs(th) * s;
    for i=1:length(file)
        if abs(file(i)) > aux
            espor=espor+1;
            counter=counter+1;
            pause=0;
            if espor > espnoise
                sbuf=cat(1,sbuf,file(i-counter+1:i));
                counter=0;
            end
        else
            pause=pause+1;
            counter=counter+1;
            if pause > spause
                nbuf=cat(1,nbuf,file(i-counter+1:i));
                espor=0;
                counter=0;
                pause=0;
            end
        end
    end
end
```