Universidade do Minho
Escola de Engenharia

Processamento Digital de Sinal
ENGENHARIA DE TELECOMUNICAÇÕES E INFORMÁTICA
2020/2021

(Docente: Carlos Manuel Gregório Santos Lima)

9 de maio de 2021

Subamostragem de um sinal de áudio em MatLab

Rui Filipe Ribeiro Freitas - a84121@alunos.uminho.pt

Índice

Lista de figuras

Introdução

No âmbito da Unidade Curricular de PDS (Processamento Digital de Sinal) foi proposto a realização de um trabalho com o objetivo de implementar em MatLab um módulo que permita diminuir a frequência de amostragem por meios digitais evitando a ocorrência de *aliasing*, processo normalmente designado por *Downsampling*.

A realização deste trabalho tem como objetivo desenvolver competências extras em MatLab assim como consolidar os conhecimentos sobre alteração da frequência de amostragem por amostragem discreta de um sinal contínuo e filtros digitais.

Na execução deste trabalho primeiro comecei por realizar uma pesquisa sobre os vários comandos que iria utilizar na elaboração do código em MatLab. Após isso passei para a sua execução utilizando o método *audiorecorder* e *recordblocking*. Através disto foi possível a gravação do áudio a utilizar no processo de decimação. Depois é efetuado um conjunto de comandos utilizando um filtro desejado para resolver o problema proposto. Como o filtro que cada um devia utilizar é atribuído através do resto da divisão do número de estudante por 5, o filtro a utilizar foi o de Chebyshev tipo I ou II visto o meu número ser o 84121. Acabei por optar por utilizar o tipo I visto ser o mais comum.

De modo a ser capaz de cumprir com os objetivos deste trabalho terei de colocar em prática conhecimentos adquiridos ao longo do semestre assim como realizar um estudo sobre as várias matérias a abordar.

Fundamentos

1. Filtros

Os filtros passa-baixo são filtros que permitem a passagem de sinais de baixas frequências, atenuando sinais com amplitudes superiores à frequência de corte. Estes são constituídos por 3 bandas de frequência distintas. A banda passante, que corresponde às frequências do sinal de entrada, a banda de rejeição ou de corte, que corresponde à gama de frequências do sinal de entrada que são rejeitadas e a banda de transição, que corresponde à zona intermédia entre a banda passante e a banda de rejeição (Figura 1).

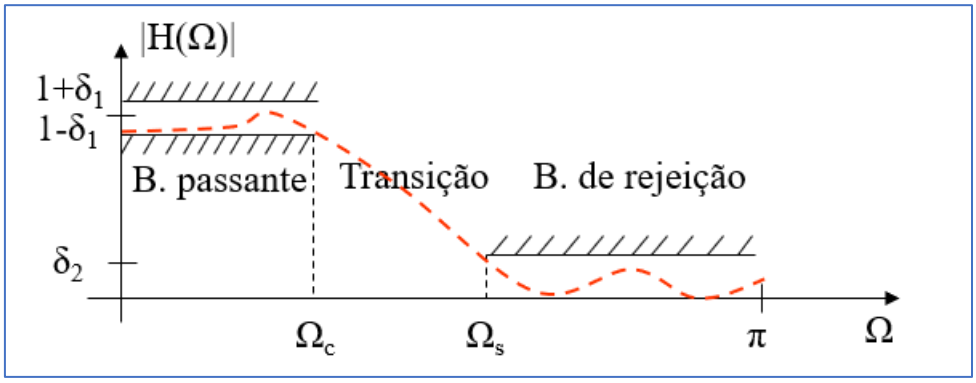

Figura 1 - Filtro passa-baixo.

Os filtros de Chebyshev podem ser de dois tipos: tipo I – com oscilações só na banda passante e tipo II – com oscilações só na banda de rejeição. Os filtros de Chebyshev exibem uma resposta de igual ripple na banda passante e decresce continuamente na banda de rejeição. Nestes filtros pode-se obter uma maior inclinação entre a banda passante e a banda de rejeição, obtendo uma banda de transição mais estreita.

O filtro que será utilizado no desenvolvimento do módulo em MatLab será o Chebyshev tipo I por ser o mais comum, representado pela figura seguinte, com a) ordem par e b) ordem ímpar.

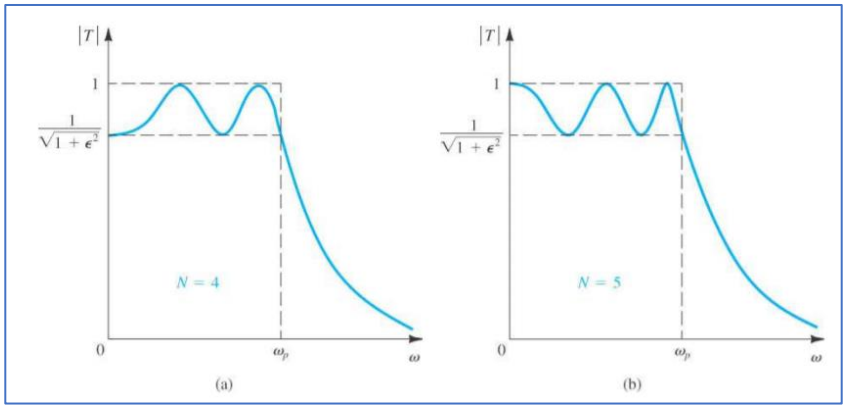

Figura 2 - Filtro de Chebyshev.

2. Decimação

A decimação corresponde à diminuição da frequência de amostragem, tendo só utilidade se um sinal foi amostrado a uma taxa maior que a taxa de *Nyquist*. No entanto, se um sinal foi amostrado à taxa de *Nyquist* e a sua largura de banda foi reduzida por um filtro discreto a sua frequência de amostragem pode ainda ser reduzida por decimação.

Se um sinal x[n] for simplesmente amostrado, iria dar origem a uma forte distorção, sendo necessário remover com um filtro passa-baixo a banda superior do sinal. O processo faz com que sinais de alta frequência sejam sobrepostos, o que provoca o *aliasing*. De forma a combater este fenómeno é necessária a aplicação de um filtro, no meu caso vou utilizar o filtro de Chebyshev, que funciona como um filtro *anti-aliasing*.

O sinal $x_d[n]$ é obtido por decimação de M:1 do sinal x[n]:

$$x_d[n] = x(Mn), \text{ M inteiro}$$

A decimação é uma operação sem perda de informação se $X(e^{jw})$ for zero fora do intervalo $[-\pi/M, \pi/M]$, portanto de banda limitada. Logo, esta operação é normalmente precedida de uma filtragem passa baixo para que esta situação seja garantida. A figura seguinte corresponde ao diagrama de blocos de uma decimação, que é realizada seguida de um compressor.

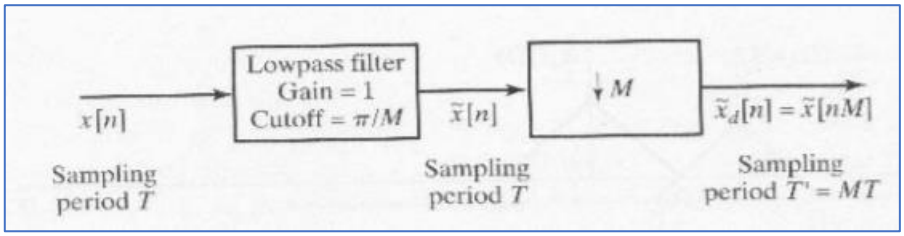

Figura 3 - Diagrama de blocos decimação.

A decimação visa a recolha periódica de amostras de um dado conjunto de dados, tornando a sua variação mais lenta e suave de forma a permitir uma interpretação mais coerente e facilitada dos dados. Na figura seguinte é apresentado um exemplo de uma situação real onde é utilizada a decimação.

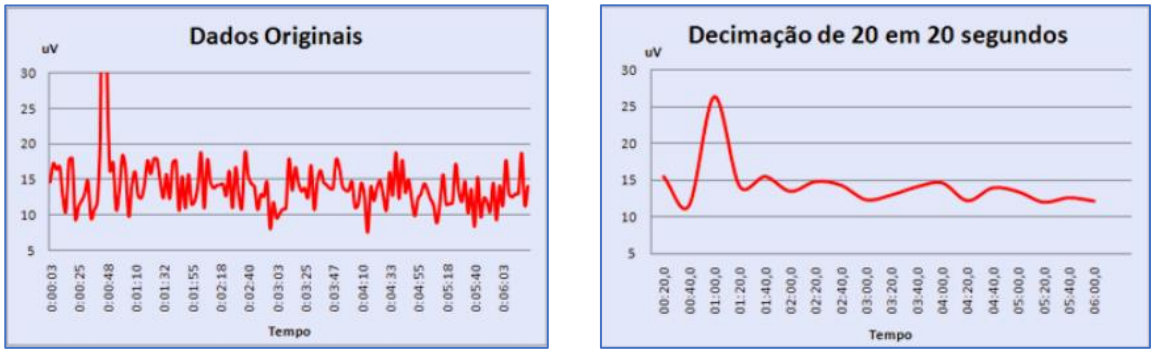

Figura 4 - Exemplo de uma decimação.

3. Problema proposto

Foi proposta a implementação de uma função realizada em MatLab cujos parâmetros de entrada são um segmento de áudio amostrado a 8 kHz e um fator de subamostragem ou período do trem de impulsos(N) do amostrador discreto. A função deve devolver áudio amostrado a $\frac{Fs}{N}$. Esta função deve ser chamada recursivamente para diferentes valores de N colocados em ordem crescente e deve ser ouvido o áudio para aferição subjetiva da perda sucessiva de conteúdo espectral do sinal.

As especificações mínimas do filtro devem ser:

1- Ripple na banda passante de 40 dB.
2- Ripple da banda de rejeição de 60 dB.
3- Largura de banda de transição de 20% da banda passante.

Este problema será implementado com auxílio do programa MatLab, fazendo chamada recursiva da função de modo a gerar os diferentes valores do coeficiente de subamostragem, verificando sempre o áudio no fim, para se poder identificar a perda sucessiva do conteúdo espectral.

Implementação

Em primeiro lugar é realizada uma gravação de áudio de 5 segundos de forma a gerar um sinal de entrada que, mais tarde, será usado no processo de decimação e filtragem.

```
%% Gravação de áudio

micro = audiorecorder(8000,8,1);     % Criação do objeto audiorecorder
disp("Start speaking.");
recordblocking(micro,5);             % Gravação de 5 segundos
disp("End of recording.");
audiorec = getaudiodata(micro);      % Armazenar os dados num array
plot(audiorec);                      % Gráfico dos dados
sound = play(micro);                 % Ouvir a gravação
disp('Properties of Sound:');
get(sound);                          % Propriedades do som gravado
audiowrite('som.wav',audiorec,8000);
fourier = fft(audiorec);
[audiorec,Fs]=audioread('som.wav');
pause(5);                            % Pausa para ouvir o som original
```

Figura 5 - Excerto do código para a criação do áudio.

Após obter o sinal de áudio são definidos os parâmetros a serem utilizados, de maneira a originar o filtro com as especificações pedidas no enunciado. Com os dados fornecidos do *ripple* na banda passante, do *ripple* na banda de rejeição e da largura da banda de transição sabendo a da banda passante conseguimos obter os valores necessários para fazer a projeção do filtro de Chebyshev tipo I. Os valores configurados no MatLab estão presentes a seguir:

```
%% Parâmetros para a realização do filtro

plot(audiorec);
Ny = Fs/2;                  % Frequência de Nyquist
Rp = 40;                    % ripple na banda passante
Rs = 60;                    % ripple na banda de rejeição
OmegaP = (Ny/N);
OmegaS = 1.2*OmegaP;
Wp = OmegaP/Ny;             % Frequência da banda passante
Ws = OmegaS/Ny;             % Frequência da banda de rejeição
Ts = 1/8000;                % Ts -> Período de amostragem
```

Figura 6 - Excerto do código para a projeção do filtro.

Com os parâmetros da figura anterior realizamos a projeção do filtro de Chebyshev tipo I (figura 8) através do método *cheb1ord*. Com os valores de N e Wn foi realizada a criação filtro através do comando *cheby1* onde obtemos os valores do numerador e do denominador da função de transferência do filtro analógico. O comando usado para isto em MatLab está presente na figura 7. Após isso realizamos uma transformação bilinear para que, recorrendo à função filter se possa utilizar o filtro no sinal original.

```
%% Filtro de Chebyshev tipo I

[n,Wn] = cheb1ord((2/Ts)*tan((Wp)/2),(2/Ts)*tan((Ws)/2),Rp,Rs,'s');

%% Criação do filtro e transformação bilinear

[B,A] = cheby1(n,Rp,Wn,'s');                    % Criação do filtro
[Num,Den] = bilinear(B,A,1);                    % Transformação bilinear

%% Filtragem do sinal e decimação

filteredSignal = filter(Num,Den,audiorec);      % Sinal filtrado
decimado = downsample(audiorec,2);              % Decimação
filteredSignal2 = filter(Num,Den,decimado);     % Sinal decimado
fourier2 = fft(decimado);
fourier3 = fft(filteredSignal);
fourier4 = fft(filteredSignal2);
```

Figura 7 - Excerto do código para a criação do filtro.

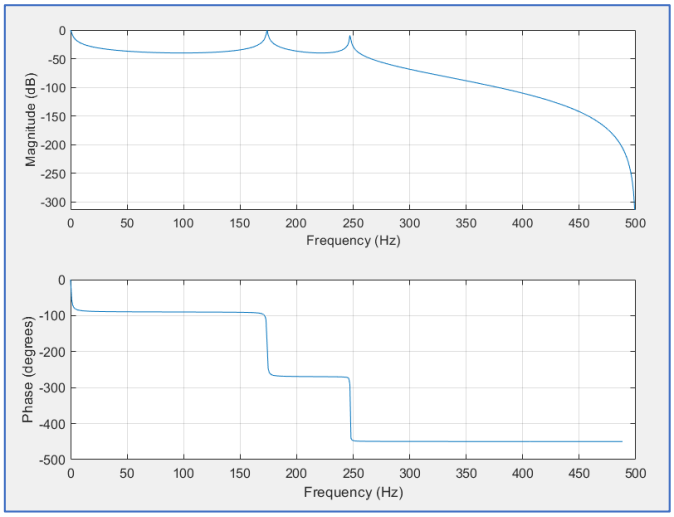

Figura 8 - Filtro de Chebyshev.

Testes e discussão de resultados

Após termos aplicado o filtro tanto no áudio original como no áudio decimado passamos à implementação de uns gráficos de modo a podermos observar melhor as diferenças entre os sons e podermos ver o efeito do filtro. A figura 10 corresponde à comparação entre o som original e o som filtrado e na figura 9 entre o som decimado e o som decimado filtrado. Ambos os gráficos foram tirados com um fator de subamostragem de 5.

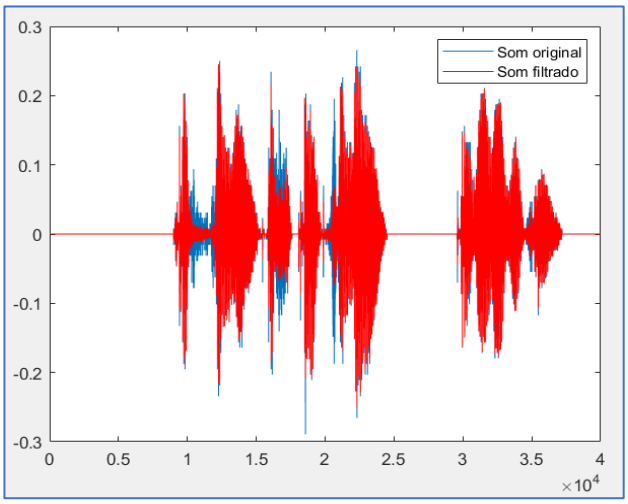

Figura 10 - Comparação som original com som filtrado.

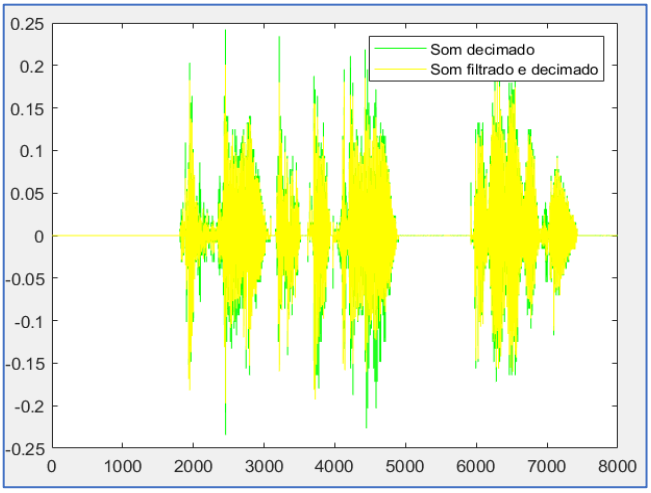

Figura 9 - Comparação som decimado com som decimado e filtrado.

Para além dos gráficos elaborados anteriormente também foram implementados 2 tipos de gráficos que permitem uma melhor comparação entre os vários sons. O primeiro que corresponde aos gráficos das transformadas de *Fourier* onde é possível observar o que fica no sinal após este ser filtrado (Figura 11) e o segundo que corresponde aos espetros dos sinais sonoros onde que se consegue observar a "limpeza" do ruído por parte do filtro (Figura 12).

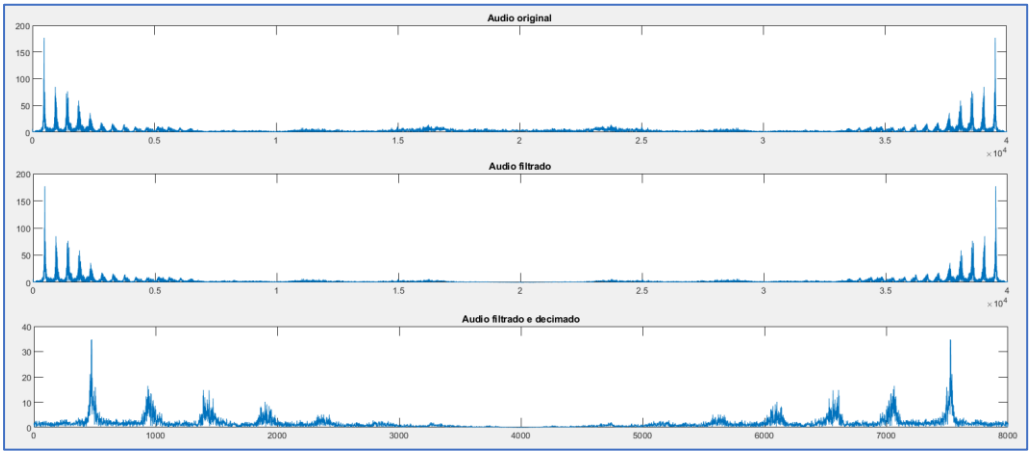

Figura 11 - Gráfico das transformadas de Fourier.

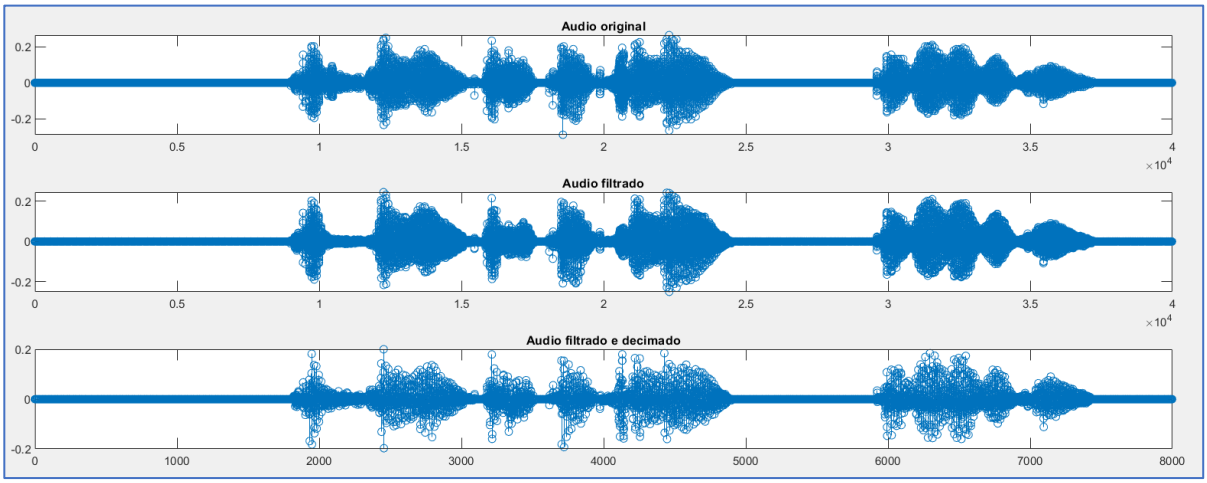

Figura 12 - Gráficos dos espetros dos sinais sonoros.

Relativamente a este último gráfico (Figura 12) conseguimos observar uma remoção grande do ruído do sinal através do desaparecimento dos pontos mais isolados. Quando é realizada a filtragem no som decimado, que corresponde ao terceiro gráfico, observamos mais pontos isolados, no entanto, observamos também uma menor amplitude do sinal e uma grande perda de informação.

De modo a realizar este trabalho de forma eficiente e fiável foi efetuado um *loop* em MatLab que incrementava o valor do fator de subamostragem (N) em 4, ou seja, na primeira vez era efetuado todo o processo com um fator de 1, na segunda com um fator de 5 e por aí em diante. Como a entrada era um sinal de áudio, todas as vezes que se corria o script guardava um áudio de 5 segundos e a única variável de entrada que alterava era o N. O som filtrado vai se tornando mais "suave" em relação ao ruído, no entanto, este torna-se mais abafado e impercetível à medida que o fator de subamostragem vai aumentando.

De seguida apresento 2 resultados distintos para um N de 4 e um N de 8. Através destes resultados conseguimos observar graficamente a diminuição da amplitude do áudio relativa à perda de informação através do processo de decimação assim como a desconfiguração do mesmo quando este é filtrado e decimado.

N = 4

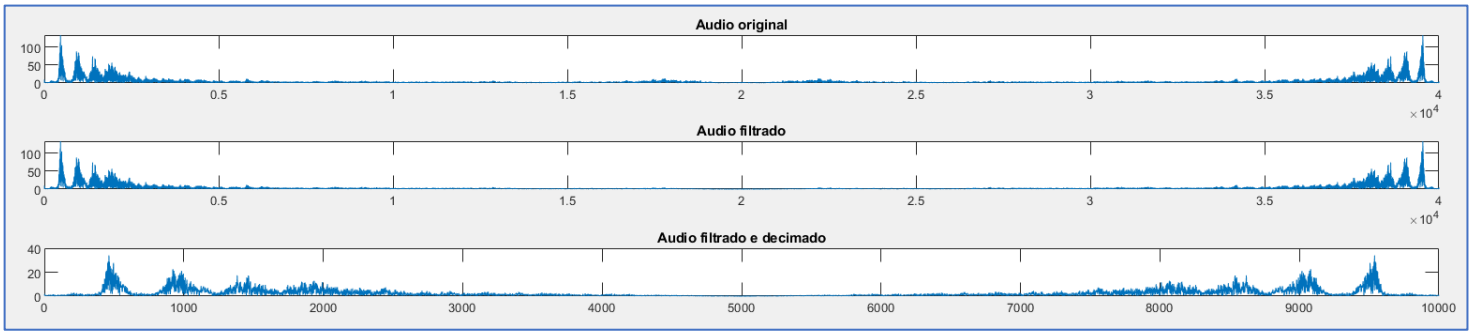

Figura 13 - Transformada de fourier com N = 4.

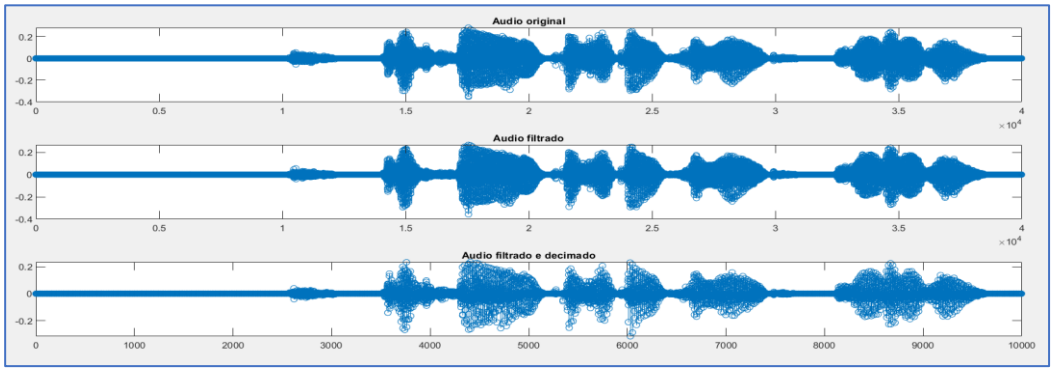

Figura 14 - Espetros dos sinais com N = 4.

N = 8

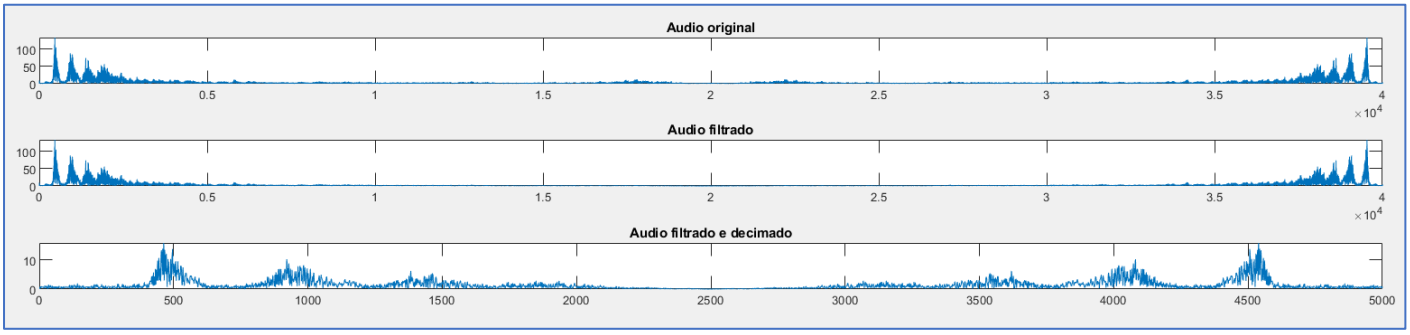

Figura 15 - Transformada de fourier com N = 8.

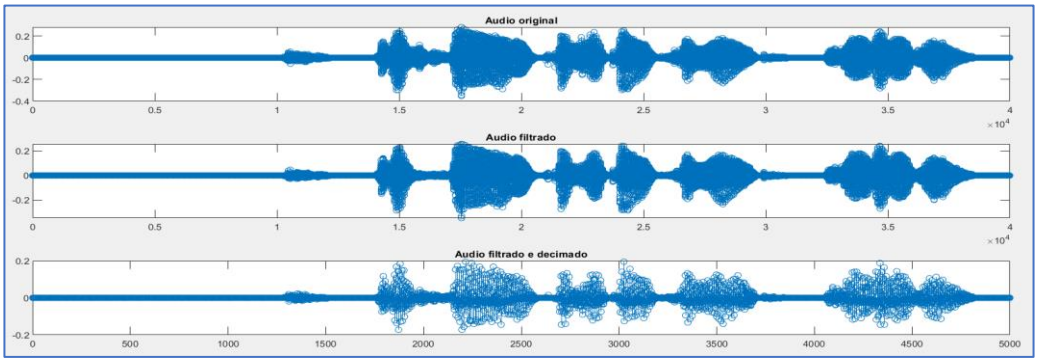

Figura 16 - Espetro dos sinais com N = 8.

Conclusão

Através da realização deste trabalho foi possível consolidar tanto conhecimentos sobre a matéria lecionada relativa a filtros e decimação assim como uma melhor familiarização com o ambiente de desenvolvimento MatLab.

Assim, após a realização dos testes demonstrados anteriormente consegui observar o trabalho desenvolvido em MatLab de modo a chegar a resultados que eram fiáveis na realização deste trabalho.

Relativamente às conclusões retiradas do desenvolvimento deste trabalho, no que toca ao filtro utilizado pode-se concluir que aquando da gravação do sinal, este é guardado com a presença de algum ruído. Através da utilização do filtro, este ruído é descartado, não na totalidade, mas uma boa parte, pelo que é possível ouvir no resultado final um som mais limpo.

No que toca à decimação, à medida que o fator de subamostragem é incrementado, verifica-se uma maior perda de valores no array guardado. Isto leva a uma perda considerável na informação do som gravado que é sentido aquando da reprodução do sinal sonoro.

Quando é efetuado a filtragem e a decimação de um sinal e é incrementado o fator de subamostragem conseguimos observar 2 fatores importantes na aferição dos resultados. Quanto à utilização do filtro este leva a uma perda de amplitude do sinal e relativamente à utilização da decimação, quanto maior o fator de subamostragem maior é a perda de informação do som. Isto leva a que quando ouvimos o resultado final, como o ouvido humano não está preparado para certas frequências de sinal, o som pareça mais abafado e incompreensível.