Decimação

João Pedro Oliveira Faria

Abril 2020

Conteúdo

1 Introdução

O principal objetivo deste trabalho consiste na implementação de um módulo no Matlab que permita a diminuição da frequência de amostragem de um sinal por meios digitais, com a finalidade de ocupar menos memória bem como exigir menos cálculos de processamento. Este processo denomina-se down-sampling ou decimação. Além disso, para que não ocorra o efeito de "aliasing"é necessário aplicar ao sinal previamente amostrado um filtro passa-baixo. O mesmo irá ser desenvolvido usando um filtro elíptico que obedece às seguintes condições:

- Ripple na banda passante de 40 dB
- Ripple na banda de rejeição de 60 dB
- Largura de banda de transição de 20% da banda passante.
- A frequência de amostragem de 8kHz.

2 Amostragem e Decimação

Um sinal em tempo contínuo tem de ser amostrado a uma determinada frequência para poder ser processado digitalmente, isto é, em certos instantes periódicos de tempo, é obtido uma amostra do

sinal. Após processamento do mesmo, converte-se o sinal de digital para analógico para ser ouvido, por exemplo.

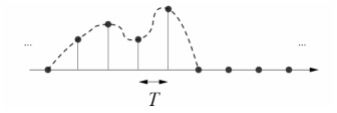

Figura 1: Amostragem.

Contudo, mesmo após obtido um sinal amostrado, este pode conter demasiada informação, para o objetivo a que era proposto. Para tal, faz-se um downsampling, isto é, reduz-se o número de amostras do sinal. A decimação pode considerar só uma a cada duas amostras ou uma a cada três amostras, sendo as restantes zero, tratando-se assim de um fator de decimação, N, de dois e três, respetivamente.

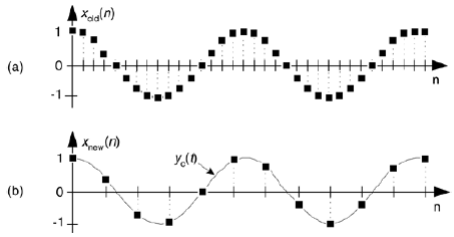

Figura 2: Decimação no tempo discreto.

3 Aliasing

Ao aplicarmos o processo de redução de amostras descrito anteriormente, podemos provocar "aliasing", caso não seja obedecido o teorema de Nyquist, que indica que a máxima frequência do sinal não deve ser superior a metade da frequência de amostragem. Este efeito faz com que sinais com diferentes frequências sejam indistinguíveis entre si quando amostrados sem obedecer ao teorema acima descrito. Consequentemente, em vez de alterarmos a frequência de amostragem podemos aplicar um filtro passa-baixo para eliminarmos o efeito. Contudo, o mesmo é inexequível dado a sua natureza não causal, daí surgirem várias alternativas, cada uma com vantagens e desvantagens, que se aproximam do filtro ideal. Neste projeto, será implementado o filtro elíptico, cujo nome deriva do facto de os seus pólos no plano s formarem uma elipse.

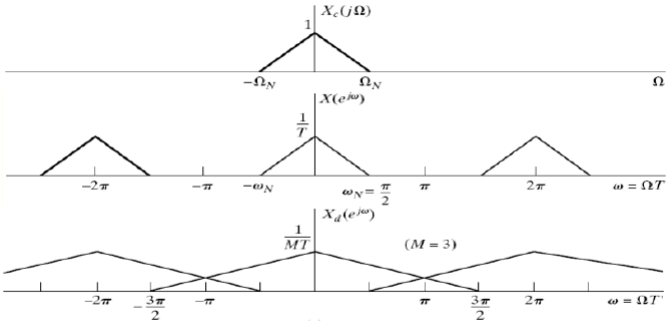

Figura 3: Decimação com aliasing, fator de decimação M=3.

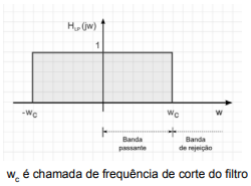

Figura 4: Filtro ideal passa-baixo.

4 Filtro elíptico

A resposta em frequência apresenta "ripples"tanto na banda passante como na de rejeição, como a combinação dos filtros chebysheb tipo 1 e 2, apresentando um roll-off mais acentuado que ambos.

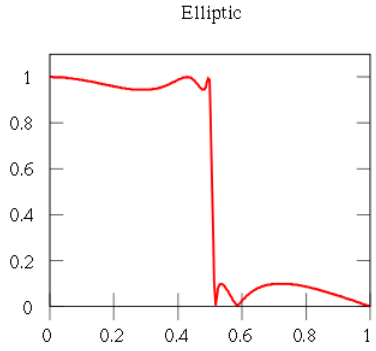

Figura 5: Filtro elíptico resposta em frequência (amplitude).

- O declive da banda de transição é N*20dB por Década.

$$G_n(\omega) = \frac{1}{\sqrt{1 + \epsilon^2 R_n^2(\xi, \omega/\omega_0)}}$$

Figura 6: Resposta em frequência do filtro elíptico.

- A fase pouco se assemelha a uma reta.
- É uma generalização dos filtros Butterworth e Chebyshev.
- Possui zeros além de pólos.
- Possui um par(es) de zeros imaginários perto da frequência corte que aumenta o declive banda transição.

Os vários parâmetros que se encontram na equação da figura 6 designam-se:

- ϵ fator de ripple, influencia a amplitude dos "ripples"em ambas as bandas passante e rejeição.
- R_n, ordem nth da função racional elítica, que define a o gráfico do filtro, recorrendo aos seguintes parâmetros:
- ξ, fator seletividade, afeta o ripple na banda de rejeição.
- ω_o, a frequência de corte, havendo na sua vizinhança um acentuado declive de ganho.

5 Código Matlab

```
%SOUND RECORDING
Fs = 8000;
sound_object = audiorecorder(Fs,8,1)                %% Parameters: 8000Hz, 8 bits, 1 channel
recordblocking(sound_object, 3);                    %% Records for 5 secs
sound_array = getaudiodata(sound_object);           %% Converts object to plottable sound
figure
plot(sound_array);                                  %% Plot 8kHz original sound signal
title('Sound signal');
figure
freqz(sound_array);                                 %% Bode diagram of sound signal
title('Bode diagram of sound signal');
sound(sound_array);                                 %% Original sound replay


for N=2 : 8                                         %% Recursive-like for
%%FILTER
Wp = 2*tan((pi/N)/2);                               %% Passband edge frequency
Ws = 2*tan((1.2*pi/N)/2);                           %% Stopband edge frequency
Rp = -20*log(0.99);                                 %% Passband Ripple 40 db
Rs = 60;                                            %% Stopband Ripple 60 db
[n, Wp] = ellipord(Wp, Ws, Rp, Rs, 's');            %% Outputs a low-pass n ord elliptic filt.
[b, a] = ellip(n, Rp, Rs, Wp, 's');                 %% Design an eliptic filter
[num, den] = bilinear(b, a, 1);                     %% Using bilinear method
```

```
for N=2 : 8                                                                    %% Recursive-like for
%%FILTER
Wp = 2*tan((pi/N)/2);                                                          %% Passband edge frequency
Ws = 2*tan((1.2*pi/N)/2);                                                      %% Stopband edge frequency
Rp = -20*log(0.99);                                                            %% Passband Ripple 40 db
Rs = 60;                                                                       %% Stopband Ripple 60 db
[n, Wp] = ellipord(Wp, Ws, Rp, Rs, 's');                                       %% Outputs a low-pass n ord elliptic filt.
[b, a] = ellip(n, Rp, Rs, Wp, 's');                                            %% Design an eliptic filter
[num, den] = bilinear(b, a, 1);                                                %% Using bilinear method


%%FILTERED SIGNAL
filtered_signal = filter(num,den,sound_array);                                 %% Filtered signal
freqz(filtered_signal);                                                        %% Filtered downsampled signal
title('Bode diagram of filtered signal');
sound(filtered_signal, Fs);                                                    %% Filtered sound replay


%%DOWNSAMPLING SIGNAL
downsampled_signal = func_downsampling(sound_array, N);                        %% Downsampling reduces numb samples
figure
plot(downsampled_signal);                                                      %% Plot 8/N kHz downsampled signal
title(sprintf('8 kHz downsampled sound signal, N=%d', N));
figure
freqz(downsampled_signal);                                                     %% Bode diagram of downsampled signal
title(sprintf('Bode diagram of downsampled sound signal, N=%d', N));
sound(downsampled_signal, Fs/N);                                               %% Downsampled sound repla
```

```
%%FILTER/DOWNSAMPLING SIGNAL
filtered_downsampled_signal = filter(num,den,downsampled_signal);          %% Filtered downsampled signal
sound(filtered_downsampled_signal, Fs/N);                                  %% Filtered/downsampled sound replay
filtered_downsampled_signal = upsample(filtered_downsampled_signal, N);    %% Upsampled so it is easy to compare
plot(filtered_downsampled_signal);                                         %% Plot filtered 8/N kHz downsampled signal
title(sprintf('Filtered Downsampled signal, N=%d', N));
figure
freqz(filtered_downsampled_signal);                                        %% Bode diagram of downsampled signal
title (sprintf('Bode diagram of downsampled filtered signal, N= %d', N));


%%Comparison between Original/DS/Filtered_downsampled
figure
plot(sound_array, 'b-');
hold on
plot(filtered_signal, 'r-');
plot(downsampled_signal, 'y-');
plot(filtered_downsampled_signal, 'g-');                                   %% Plot original and dec/filt 8 kHz signal
hold off
title(sprintf('Original and Filtered signal, N=%d', N));
figure
freqz(num,den);                                                            %% Bode diagram of eliptic filter
title (sprintf('Bode Diagram of Low-pass eliptic filter, N=%d', N));
```

```
%% N point FFT for aliasing analysis
L=length(downsampled_signal);
NFFT = 2^nextpow2(L);
decimated_signal_fft = abs(fft(downsampled_signal, NFFT));
freq = Fs / 2 * linspace(0, 1, NFFT / 2 + 1);
figure;
subplot(2,1,1);
plot(freq, decimated_signal_fft(1 : NFFT / 2 + 1));
title(sprintf('Espectro do Sinal Decimado e não filtrado com N= %d', N));
xlabel('frequency (Hz)');
ylabel('Amplitude');
L=length(filtered_downsampled_signal);
NFFT = 2^nextpow2(L);
decimated_signal_fft = abs(fft(filtered_downsampled_signal, NFFT));
freq = Fs / 2 * linspace(0, 1, NFFT / 2 + 1);
subplot(2,1,2);
plot(freq, decimated_signal_fft(1 : NFFT / 2 + 1));
title(sprintf('Espectro do Sinal Decimado e filtrado com N= %d', N));
xlabel('Frequencia (Hz)');
ylabel('Amplitude');
```

```
%% Aliasing effect w/ FFT with zero-padding
%% Better Frequency Resolution
```

```
end
```

pds_trabalho1.m | func_downsampling.m

```
function y=func_downsampling(signal,N)
    y = zeros(1, 8000);

for i=1 : (length(signal)/N)
    y(i)=signal(N*i);                %% resampling N to N samples
end
```

6 Testes e Resultados

Os excertos de som foram amostrados recorrendo ao uso das seguintes funções de Matlab:

-audiorecord, que foi implementada para captar o som à frequência proposta de 8kHZ, 8bits, definida em 1 canal, gerando um objeto de som;
-recordblocking, que apenas grava durante 3 segundos, neste teste, o som;
-getaudiodata, que converte o objeto som num array de doubles para poder ser manipulável, por outras funções Matlab, o sinal captado;
Quanto ao resto das funções, existe documentação na forma de comentários associada ao código.
Foi utilizada a música "Afraid to Shoot Strangers"dos Iron Maiden dado que possui a voz do cantor, bem como uma guitarra e bateria para diversificar o espetro de frequências do sinal entrada.
Após discretização do sinal, o mesmo foi filtrado, decimado, por diferentes fatores de decimação (N), variando de 2 a 8, e decimado/amostrado bem como foi ainda efetuado um teste de aliasing (análise FFT de n-pontos) para mostrar como o filtro passa-baixo usado elimina este efeito para todos os N.
Por último, foi analisado os diagramas de bode dos diferentes filtros gerados, via método bilinear, para diferentes N, e o seu efeito final no som, que foi convertido para contínuo para comparar não só visualmente, mas auditivamente o mesmo ao fim de cada uma das etapas, sendo estas reportadas de seguida no relatório.

6.1 N = 2

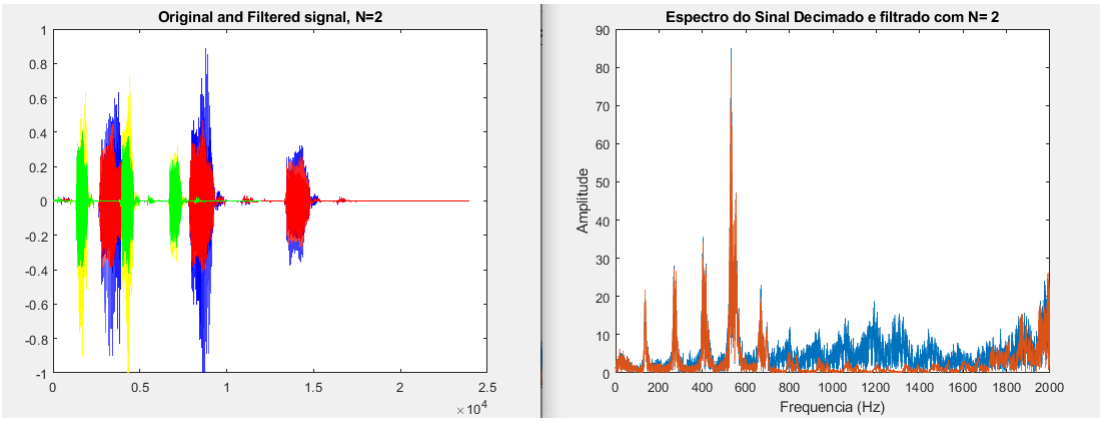

Figura 7: Na esquerda, o sinal original (azul) com o sinal filtrado (vermelho) assim como o sinal subamostrado (amarelo) sem zeros e sinal filtrado e subamostrado (verde) sem zeros. Na direita, a FFT dos sinais subamostrado a azul e subamostrado e filtrado a vermelho. Nota-se uma diminuição de amplitude do sinal azul para o sinal vermelho, na análise espetral, dado a atenuação do aliasing efetuada pelo filtro elítico.

Para N=2, tanto para o sinal filtrado como para o sinal não filtrado se assemelham ao sinal original, sendo estes mais graves um bocado que o inicialmente captado. Voz, guitarra, bateria são facilmente distinguíveis. Contudo, no sinal não filtrado, nota-se uma leve distorção na voz e na guitarra.

6.2 N = 3

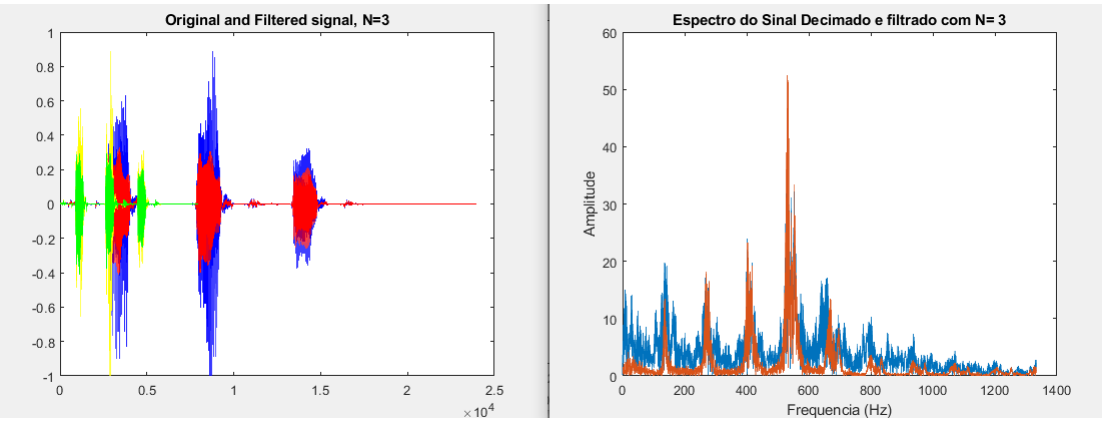

Figura 8: Na esquerda, o sinal original (azul) com o sinal decimado, sem zeros (amarelo) e sinal filtrado e subamostrado (verde). Na direita, a FFT dos sinais decimado, com aliasing (cima) e decimado e filtrado (baixo), sem aliasing.

Para N=3, ambos os sinais são mais graves, contudo, agora é o som filtrado o menos percetível, sendo o mais grave. Com algum esforço ainda se consegue ouvir sons que representam a voz e a guitarra para o filtrado; quanto ao não filtrado ainda se ouve, com dificuldade, os três diferentes timbres com alguma clareza. Ouve-se, ainda que um pouco baixo, um barulho fundo.

6.3 N = 4

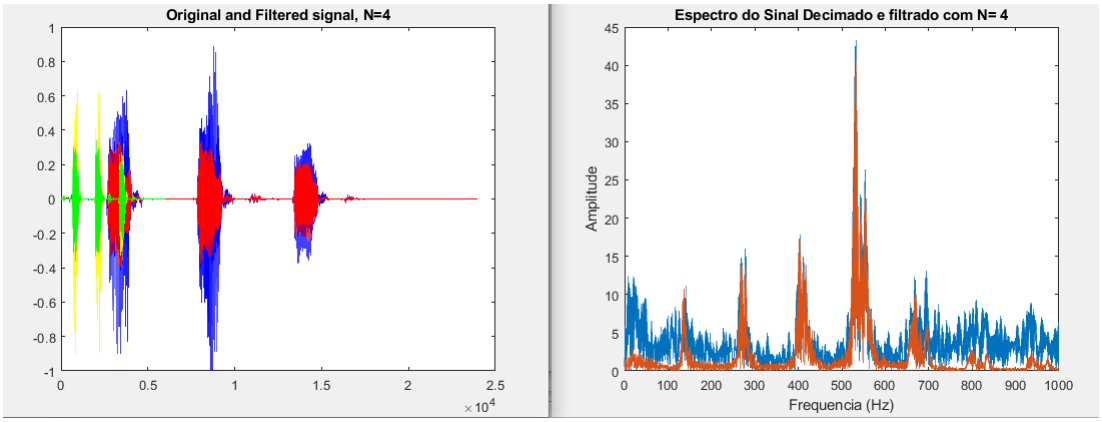

Figura 9: Na esquerda, o sinal original (azul) com o sinal decimado, sem zeros (amarelo) e sinal filtrado e subamostrado (verde). Na direita, a FFT dos sinais decimado, com aliasing (cima) e decimado e filtrado (baixo), sem aliasing.

Para N=4, do som não filtrado ouve-se sons que se podem assemelhar aos timbres mencionados(voz, guitarra, bateria). Do som filtrado, só os batuques dados na bateria é que são minimamente percetíveis. Sinais mais graves, ouve-se um barulho fundo geral.

6.4 N = 5

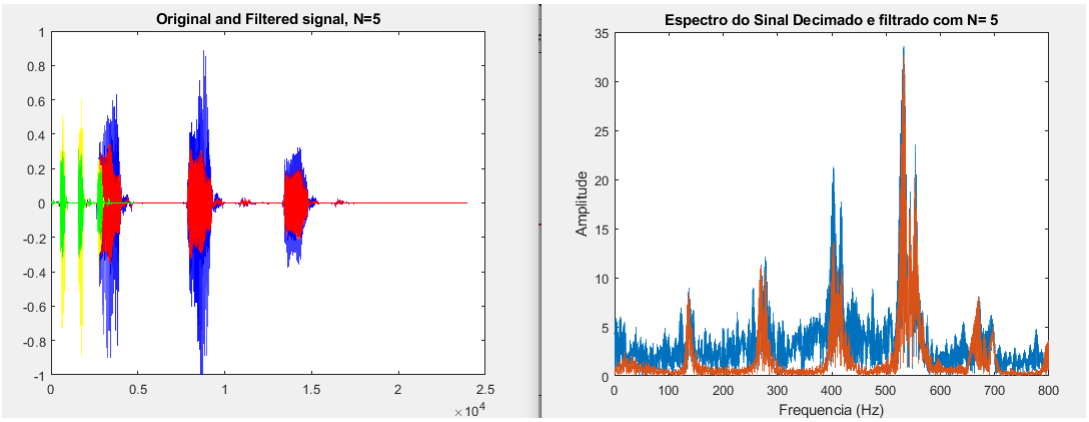

Figura 10: Na esquerda, o sinal original (azul) com o sinal decimado, sem zeros (amarelo) e sinal filtrado e subamostrado (verde). Na direita, a FFT dos sinais decimado, com aliasing (cima) e decimado e filtrado (baixo), sem aliasing.

Para N=5, o som não filtrado encontra-se ainda mais distorcido, já não se percebe a guitarra e a letra é difícil de perceber. O outro não é mais que um barulho de fundo.

6.5 N = 6

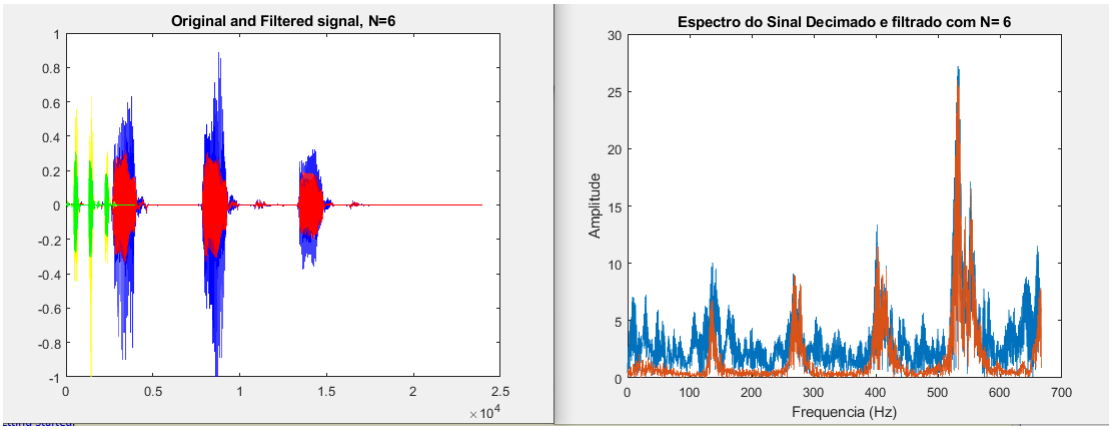

Figura 11: Na esquerda, o sinal original (azul) com o sinal decimado, sem zeros (amarelo) e sinal filtrado e subamostrado (verde). Na direita, a FFT dos sinais decimado, com aliasing (cima) e decimado e filtrado (baixo), sem aliasing.

Para N=6, apenas a voz permanece, sendo muito difícil perceber a letra. Em ambos os sinais, ouve-se um barulho vindo das profundezas, maioritariamente para o pré-filtrado. Mal se ouve a bateria.

6.6 N = 7

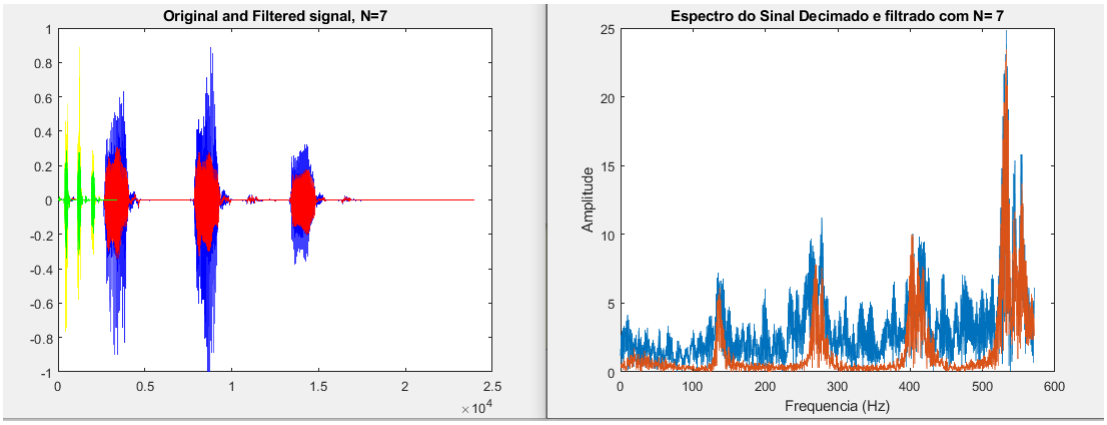

Figura 12: Na esquerda, o sinal original (azul) com o sinal decimado, sem zeros (amarelo) e sinal filtrado e subamostrado (verde). Na direita, a FFT dos sinais decimado, com aliasing (cima) e decimado e filtrado (baixo), sem aliasing.

Para N=7, muito mal se percebe a voz do som não filtrado. Continua um barulho de fundo no outro sinal. Não se ouve bateria.

6.7 N = 8

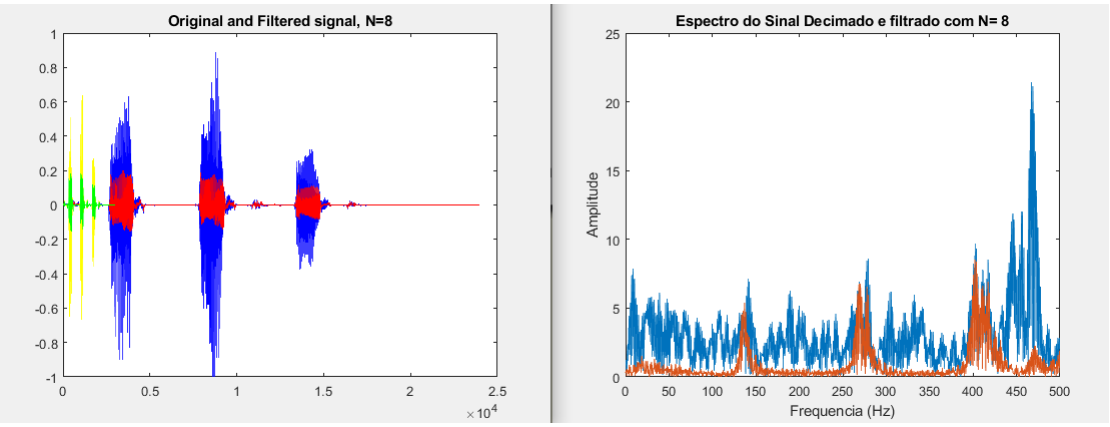

Figura 13: Na esquerda, o sinal original (azul) com o sinal decimado, sem zeros (amarelo) e sinal filtrado e subamostrado (verde). Na direita, a FFT dos sinais decimado, com aliasing (cima) e decimado e filtrado (baixo), sem aliasing.

Para N=8, ambos os sinais nada se assemelham à música. Sendo o filtrado, mais grave.

7 Conclusão

Analisados os resultados, pode-se verificar que o filtro elíptico, ao filtrar frequências muito elevadas, ajuda a evitar o fenómeno do aliasing (menos distorção do sinal, menos ruído) conforme verificado nas últimas duas imagens de cada subsecção dedicada a cada N, contudo, como visto no diagrama de bode deste, e do sinal filtrado, a banda passante tende a encurtar com o aumento de N, levando à perda de cada vez mais informação.
Nota-se que os filtros elípticos são os que apresentam maior roll-off em detrimento das outras versões de filtro, permitindo um corte mais eficiente das amostras, embora, apresentam mais "ripple", mais instabilidade, mais variações de ganho em ambas as bandas de transição e de rejeição.
Em suma, existe um balanço a ser efetuado de forma a escolher um filtro que permita a compactação do sinal, para ser mais facilmente processado e ocupar menos memória sem, no entanto, perder demasiada informação relevante. Cabe ao projetista pensar nisso e desenvolver filtros, que não têm de ser elípticos, e ajustá-los da maneira mais conveniente à sua aplicação.

8 Github

O projeto irá ser colocado num repositório online para acesso do código, ficheiros latex, imagens, entre outros ficheiros.
https://github.com/Joaovsky/PDS