Mestrado integrado em Engenharia Eletrónica e Computadores

Universidade do Minho
Escola de Engenharia

Diminuir a frequência de amostragem por meios digitais

Nome: Ângela Sofia Faria Meireles

Numero: A78343

Ano letivo: 2017/2018

Unidade Curricular: Processamento Digital de Sinal

Docente: Carlos Manuel Gregório Santos Lima

Índice:

Introdução:

Foi nos propostos implementar em Matlab um módulo que permitisse diminuir a frequência de amostragem por meios digitais evitando a ocorrência de ""aliasing", processo normalmente designado por "Downsampling". O propósito deste procedimento é compactar ao máximo o sinal de modo a ocupar menos memória e exigir menos cálculos em processamento de sinal como por exemplo, filtragem. Este processo de decimação obtido pela amostragem do sinal já digitalizado requer uma ação de filtragem passa-baixo para evitar o "aliasing", processo que deve ser efectuado recorrendo a um dos métodos de filtragem estudado. Os métodos de filtragem estudos foram: Butterworth, Chebyshev (tipo I ou II , escolher um destes), Elíptico, Janela de kaiser, Parks-McClellan.

O áudio deve ser recolhido e sintetizado no Matlab através das funções respetivas. Pretende-se por isso implementar uma função cujos parâmetros de entrada são um segmento de áudio amostrado a 8 Khz e um número que é o fator de subamostragem ou período do trem de impulsos (N) do amostrador discreto. A função deve devolver áudio amostrado a 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 (apenas ouvindo) da perda sucessiva de conteúdo espectral do sinal. Para tal foi usado o filtro janela de kaiser.

Esclarecimento dos conceitos usados:

Os sinais de tempo discreto apresentam valores definidos somente em determinados instantes do tempo. Geralmente eles provêm de sinais no tempo contínuo que são medidos ou gravados em intervalos de tempo regularmente espaçados. Por ser impossível introduzir dados contínuos nos computadores digitais, qualquer sinal ou dado necessita ser representado por um conjunto de números para processamento posterior. Se pretendermos trabalhar com sinais de tempo discreto, primeiramente devemos amostrar o seu equivalente de tempo contínuo. A **teoria da amostragem** tem um papel muito importante no desempenho de qualquer sistema de processamento digital de sinais. A **teoria da amostragem** de sinais estabelece as condições básicas para que um sinal de tempo contínuo seja representado unicamente por suas amostras tomadas em instantes de tempo regularmente espaçados.

A **teoria da amostragem** é a forma para se obter um sinal x[n] de tempo discreto a partir de um sinal x(t) de tempo continuo. Sabe-se da teoria de Fourier que a transformada da função trem de impulsos é dada pela equação, em que o valor de $\Omega_a = 2\pi/T_a$

$$\sum_{n=-\infty}^{\infty}\delta(t-nT_a) \leftrightarrow \frac{2\pi}{T_a}\sum_{k=-\infty}^{\infty}\delta(\Omega-k\Omega_a)$$

A transformada de Fourier do produto de x(t) e p(t) é dado pela convolação dos mesmos, ficando a seguinte equação:

$$X_a(\Omega) = \frac{1}{2\pi} X(\Omega) * P(\Omega)$$

Sendo Xa(Ω) a transformada de Fourier do sinal amostrado, a equação fica da seguinte forma:

$$X_a(\Omega) = \frac{1}{T_a}\sum_{k=-\infty}^{\infty} X(\Omega) * \delta(\Omega - k\Omega_a) \quad \Longrightarrow \quad X_a(\Omega) = \frac{1}{T_a}\sum_{k=-\infty}^{\infty} X(\Omega - k\Omega_a)$$

Observando o comportamento de Xa(Ω) e com o auxílio da figura 1 observamos que as cópias de x(t) podem se sobrepor. A copias não devem interferir uma com as outras, para que isso não aconteça, o espetro deve se limitado em frequência máxima. ΩM é a frequência máxima do sinal a ser amostrado. Sobreposição é evitada se a frequência de amostragem for no mínimo o dobro da frequência máxima do sinal;

$$\Omega_M \le \Omega_a - \Omega_M \quad \Longrightarrow \quad \Omega_a \ge 2\Omega_M$$

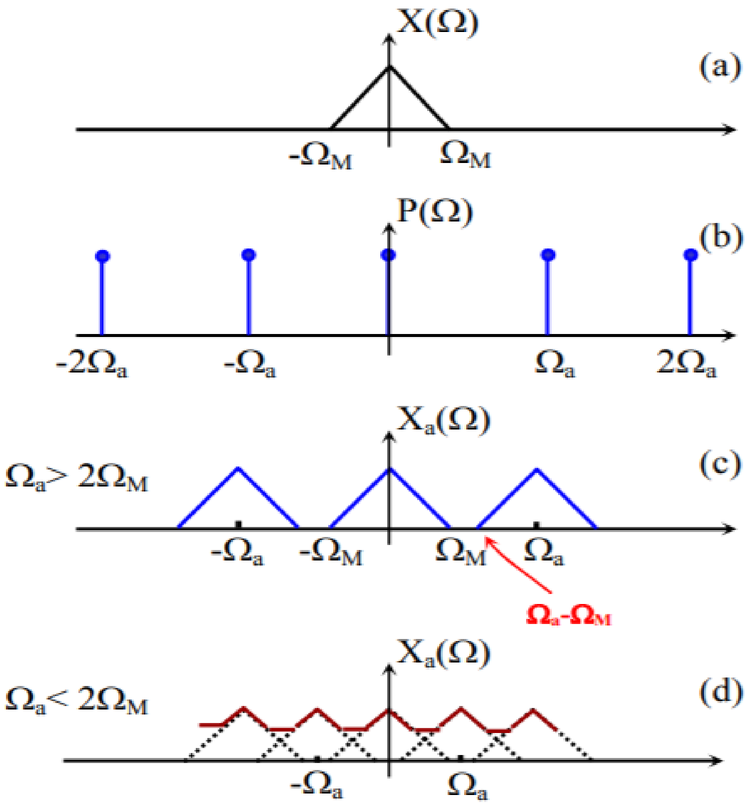


Figura 1- Efeito da amostragem no domínio da frequência

Se as condições estipuladas em cima não forem satisfeitas as cópias do espectro de x(t) vão sobrepor-se, causando a perda no formato do espectro original, este fenômeno tem o nome de aliasing. Assim sendo o espectro original desaparece e não pode ser mais recuperado a partir do sinal amostrado, se as condições forem satisfeitas o espectro original é mantido e pode ser recuperado através de uma filtragem por um filtro passa baixo ideia com uma frequência de corte Ωa/2.

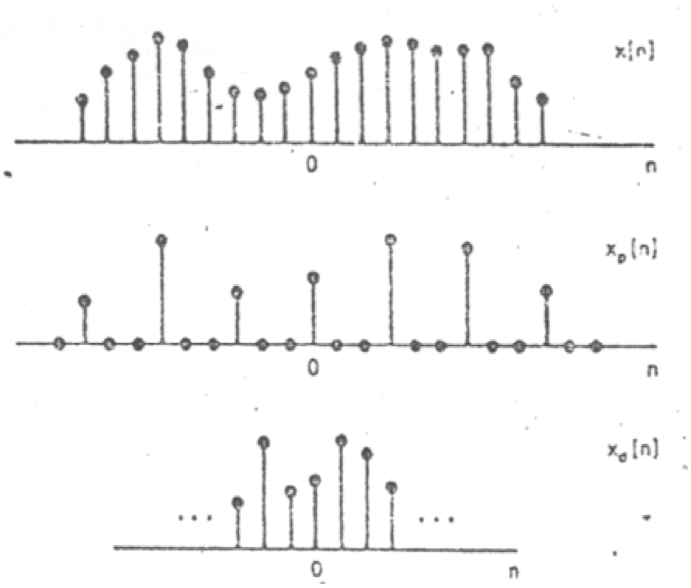
Assim sendo o teorema da amostragem pode ser descrito pela seguinte equação,

$$\Omega_a = \frac{2\pi}{T_a} \geq 2\Omega_M$$

ΩM é a frequência de Nyquist, também conhecida como sendo a frequências mínima de amostragem.

É comum que antes do processamento amostragem filtra-se o sinal (filtro anti-aliasing) com o objetivo de minimizar a energia do sinal acima da frequência de Nyquist (limitar o sinal em banda), para assim minimizar o efeito do aliasing que ocorre durante a conversão analógico-digital.

Ao modificar a frequência de amostragem de uma sequência x[n], reduzindo ou aumentando por um determinado fator. A decimação corresponde a uma diminuição da frequência de amostragem, tendo só utilidade efetiva se um sinal foi amostrado a uma taxa maior que a 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 ser reduzida por decimação.



$$x_d[n] = x[nN] = x_p[nN]$$

Observando pelo gráfico em cima, se o N=3 uma vez que a cada 3 pontos do sinal x[n] apenas 1 é selecionado para xd[n]. O valor de N determina o fator de subamostragem usado na decimação, quanto maior este for, mais se reduz o numero de amostras usado para representar o sinal.

O filtro aplicado ao sinal antes de ser feita a amostragem foi o filtro Janela de Kaiser. Os filtros em geral têm a função primária de selecionar, com pouca ou nenhuma atenuação, determinadas componentes de frequência e rejeitar ou remover todas as outras componentes de um sinal. Tais filtros são chamados de filtros seletivos em frequência. Eles são utilizados em aplicações tais como: redução de ruído, enriquecimento de sinais, equalizadores gráficos em sistemas de áudio, e muito mais.

A **janela de Kaiser** conduz a um projeto otimizado de filtros FIR. No projeto de filtros por janelas, o ideal seria utilizar uma janela tal que no domínio da frequência dela fosse maximamente concentrada em torno de w = 0. A janela que mais se aproxima desta condição ideal, é a janela de Kaiser, é também uma tentativa de resolver o compromisso entre a largura do lobo principal e atenuação do lobo secundário.

I0() é uma função de Bessel modificada do 1º tipo e de ordem zero e α=M/2. α e β(forma) permitem escolha separada de forma/largura da janela.

$$w[n]=\begin{cases}\dfrac{I_0\left[\beta\left(1-\left[\dfrac{n-\alpha}{\alpha}\right]^2\right)^{1/2}\right]}{I_0(\beta)}; & 0\le n\le M\\ 0; & outros\ \ casos\end{cases}$$

$$\beta=\begin{cases}0.1102(A-8.7); & A>50\\ 0.5842(A-21)^{0.4}+0.07886(A-21); & 21\le A\le 50\\ 0.0; & A<21\end{cases}$$

$$M=\frac{A-8}{2.285\Delta\Omega} \qquad \Delta\Omega=\Omega_s-\Omega_p \qquad \Omega_c=\frac{\Omega_s+\Omega_p}{2}$$

As formulas colocadas em cima são os parâmetros para a realização do filtro janela de kaiser. Utiliza-se em A o "ripple" correspondem-te, tendo em atenção que a síntese de filtros pelo método das janelas origina "ripples" iguais em ambas as bandas, pelo que se deve escolher o maior valor absoluto, β o coeficiente que determina a forma da janela, M o coeficiente que determina a largura dos lobos e também a ordem do filtro. ΔΩ representa a banda de transição normalizada, sendo Ω_s a frequência normalizada que delimita o inicio da banda de rejeição e Ω_p a frequência normalizada que delimita o fim da banda passante. Ω_c é a frequência de corte normalizada. Tendo calculado estes 4 paramentos o nosso filtro fica completo. Neste caso o filtro foi calculado metendo os valores no Matlab. Foi nos dado o valor do "ripple" na banda passante de 40 db, o valor do "ripple" na banda de rejeição de 60 Db e ainda a largura de banda de transição de 20% da banda passante.

O A vai tomar o valor de 60 db, como explicado em cima, Ω_s toma o valor de $1{,}2\frac{\pi}{N}$, Ω_p vai tomar o valor de $\frac{\pi}{N}$ e Ω_c que é a frequência de corte toma o valor de $\frac{\pi}{N}$.

Tendo calculado os valores em cima, também conseguimos calcular o valor de β utilizando a formula 0.1102(A – 8.7) por A > 50.

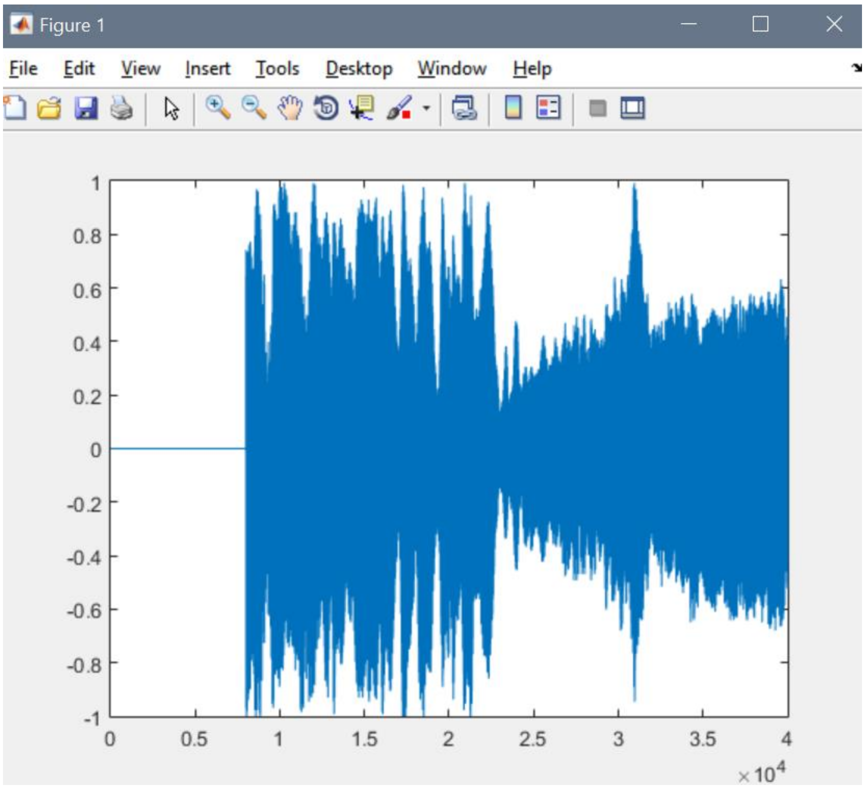
Estes cálculos foram efetuados em Matlab em parceria com a recursividade que fazia que o valor de N, fator de subamostragem, toma-se um valor entre 1 e 10.

Resultados:

Gravação do som

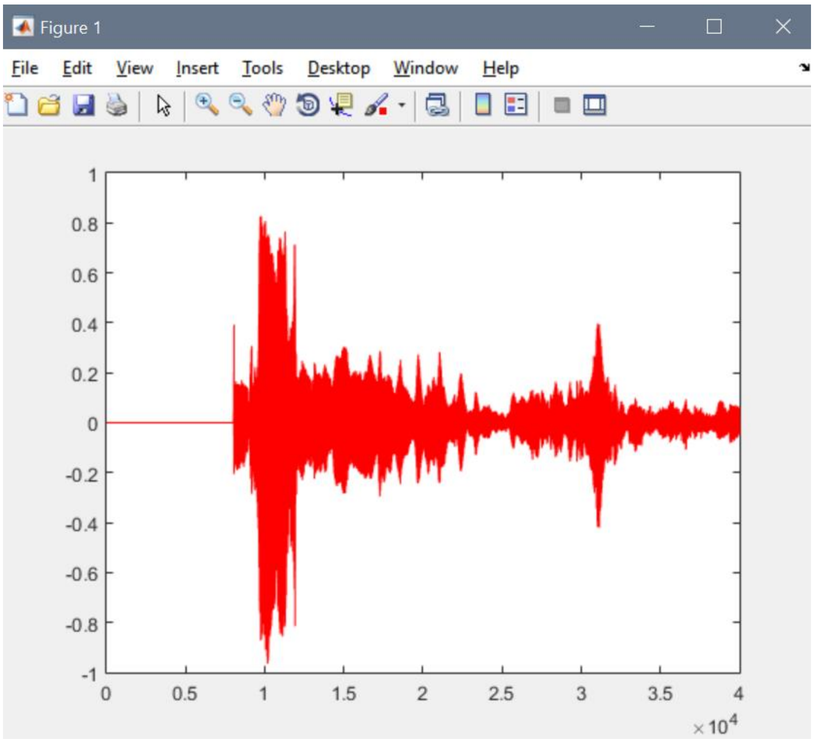
Foi gravado um áudio durante 5 segundos de alguém a falar num ambiente com ou sem ruido. A gravação foi efetuada a 1 canal e a 8 bits. O áudio deve ser recolhido e sintetizado no Matlab usando as funções wavrecord, record, sound, mas na versão do Matlab em qual realizei o trabalho não podia ser usada nenhuma dessas funções, então foi necessário arranjar uma função paralela a essa que fizesse a mesma coisa, a função usada foi *audiorecorder*.

O gráfico do áudio gravado é o seguinte, sem lhe ser aplicado o filtro e sem sofre amostragem:



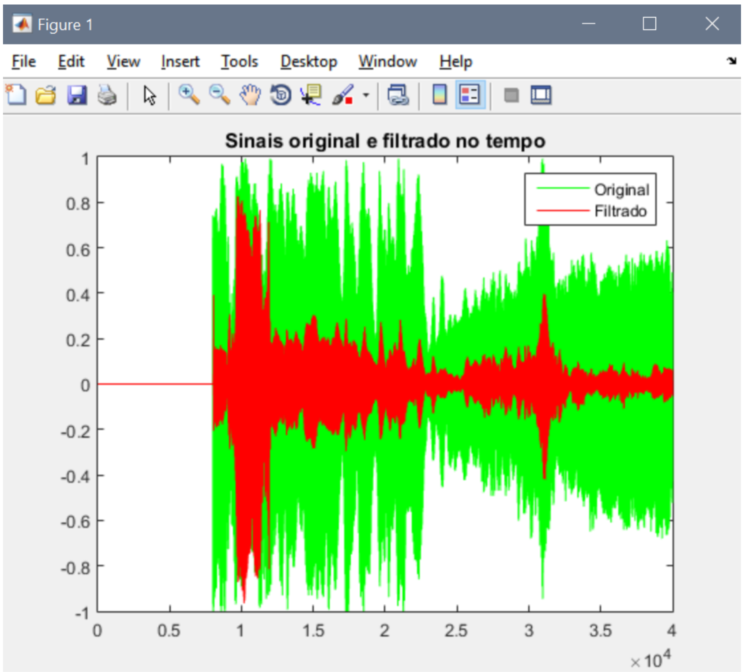
Depois de aplicarmos o filtro e amostramos o sinal para diferentes valores de subamostragem (N).

Aplicar o filtro para um valor de N=3:

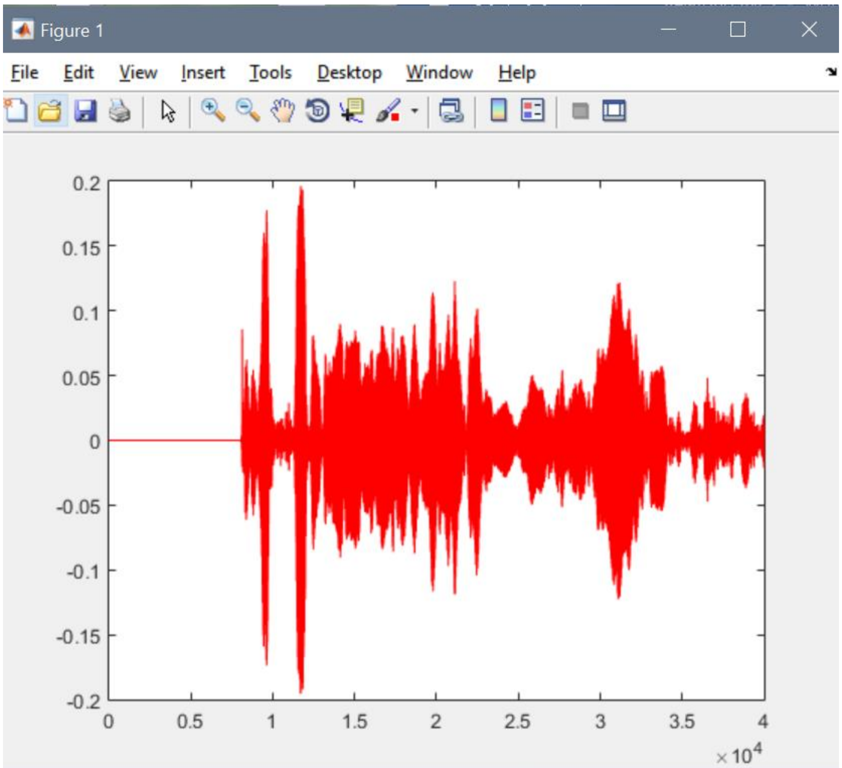


Depois de aplicar o filtro e de aplicar o fator de subamostragem, nota-se uma brusca diminuição das frequências, tendo em consideração que se usou um N pequeno.

Mas para as diferenças serem mais rapidamente notadas vamos juntar os dois sinais, o sinal original com o sinal filtrado no tempo.

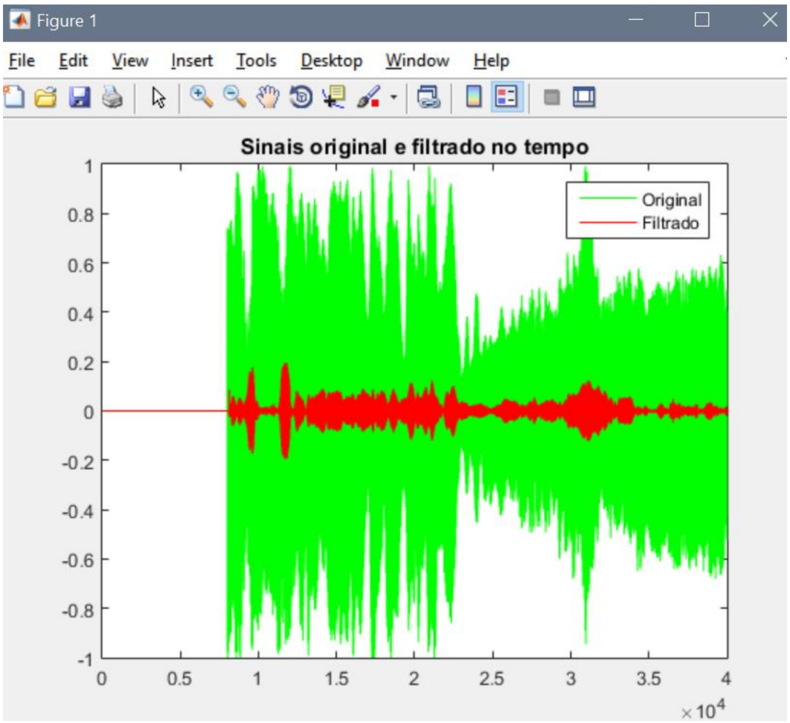


Aplicar o filtro para um valor de N=7:

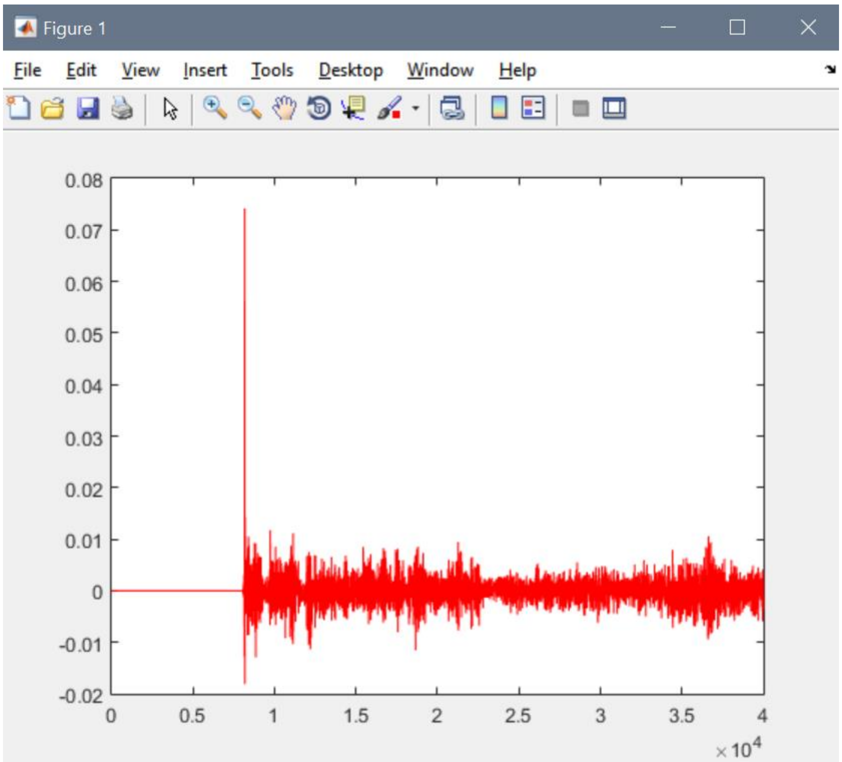


Depois de aplicar o filtro e de aplicar o fator de subamostragem, nota-se alguma diminuição das frequências, tendo em consideração que se usou um N um bocadinho maior.

Mas para as diferenças serem mais rapidamente notadas vamos juntar os dois sinais, o sinal original com o sinal filtrado no tempo.

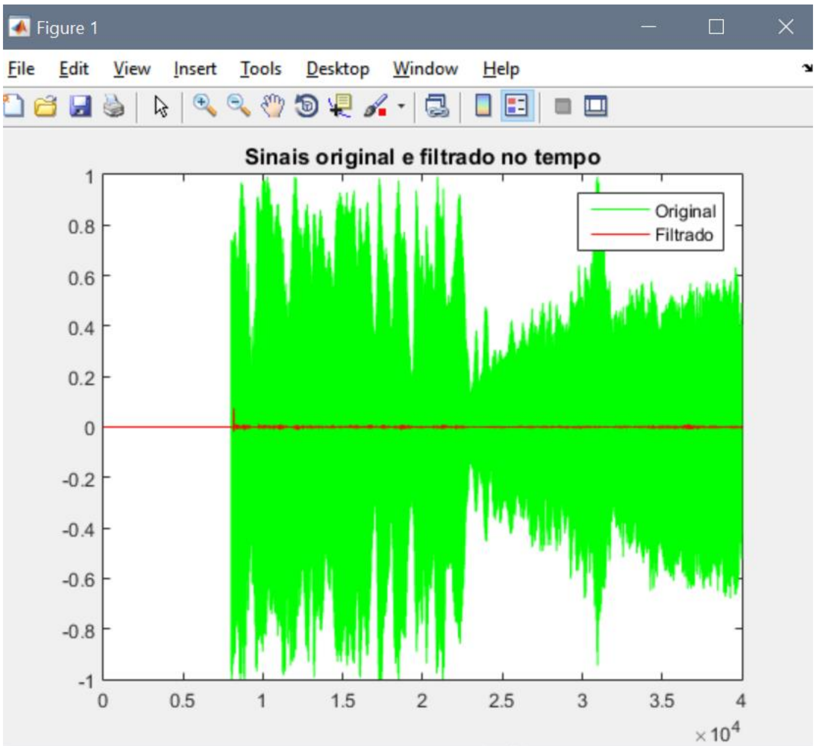


Por ultimo aplicamos o filtro para um valor de N=9:



Depois de aplicar o filtro e de aplicar o fator de subamostragem, nota-se na grande diminuição das frequências, tendo em consideração que se usou um N maior.

Mas para as diferenças serem mais rapidamente notadas vamos juntar os dois sinais, o sinal original com o sinal filtrado no tempo.



Conclusão:

Através da realização deste trabalho foi possível consolidar, de uma forma geral, todos os conhecimentos, em particular da ferramenta de desenvolvimento (Matlab), abordados nas aulas da unidade curricular de Processamento Digital de Sinal.

Assim após os testes e da observação dos gráficos obtidos podemos concluir que a medida que o N, facto de subamostragem, se altera e aumenta provoca uma diminuição da frequência, provando assim que quanto maior o N, mais é reduzido o numero de amostras para representar o sinal.

A medida que o N é aumentado, por observação dos gráficos verificamos então que as frequências altas são filtradas ficando assim só com frequências mais baixas, podemos então dizer que o objetivo, diminuir a frequência de amostragem, foi cumprido. O ultimo teste a ser feito, é ouvir o áudio depois de ser filtrado, e o que acontece é que todo o ruido que existe a volta desapareceu e só ficou mesmo o som de alguém a falar.

Bibliografia:

- Aulas e apontamentos de Processamento digital de sinal ano letivo 2017/2018;
- http://www.mathworks.com.

Anexos:

Código Matlab

função.m

```
function funcao
fs = 8000;
recObj = audiorecorder(fs,8,1);
recordblocking(recObj, 5);
play(recObj);
myRecording = getaudiodata(recObj);
plot(myRecording);
N=3;
wc=pi./N;
ws=1.2*pi./N;
M=ceil((60-8)/(2.285*(ws-wc)))
beta=0.1102*(60-8.7)
hd=ideal_lp(wc,M);
w=(kaiser(M,beta))';
h=hd.*w;
audio = myRecording;
y=decimacao(N,audio);
f=filter(h,1,myRecording);
plot(audio,'g');
hold on
plot(f,'r');
title('Sinais original e filtrado no tempo')
legend('Original','Filtrado');
hplayer = audioplayer(f, fs/N);
play(hplayer);


end
```

Ideal_lp.m

```
function hd=ideal_lp(wc,M)
alpha=(M-1)/2;
n =(0:1:(M-1));
m=n-alpha+eps;
hd=sin(wc*m)./(pi*m);
end
```

decimacao.m

```
function y=decimacao(N,audio)
y=0;
for i=1; length(y)/N
  y(i)=audio(i*N)

end
```