Universidade de Coimbra
Faculdade de Ciências e Tecnologia
Departamento de Engenharia Electrotécnica e de Computadores

Processamento Digital de Sinal – 2019/2020

Trabalho Prático n.1

Áudio, Filtragem e DFT

Pedro Ribeiro 2013136821
Pedro Silva 2011149228

O documento sobre imagem da UC obriga aos novos símbolos.
Ver:
https://www.uc.pt/identidadevisual/

Falta a indicação do nº de grupo, mesmo que esteja no nome do documento.

1 Introdução

Neste trabalho laboratorial, pretendemos fazer uma introdução a processamento digital de sinais, neste caso, um sinal de áudio usando o Matlab como ferramenta de análise e de trabalho. Neste trabalho faremos:

- Uma filtragem de sinal áudio com filtros de 1ª ordem passa-alto em forma directa I e II.
- Calcular e estudar a DFT (*Discrete Fourier Transform*) do sinal áudio.
- Estudar o contúdo espectral do sinal recorrendo a um sonograma do mesmo.

só II.

2 Filtragem passa-alto

Pretendemos estudar o sinal de áudio, amostrado a uma frequência de amostragem de 16Khz, do ficheiro 'pcmtest_10Hz.wav', disponibilizado na plataforma InforEstudante. Recorrendo ao Matlab, traçamos o sinal obtendo Sinal de áudio:

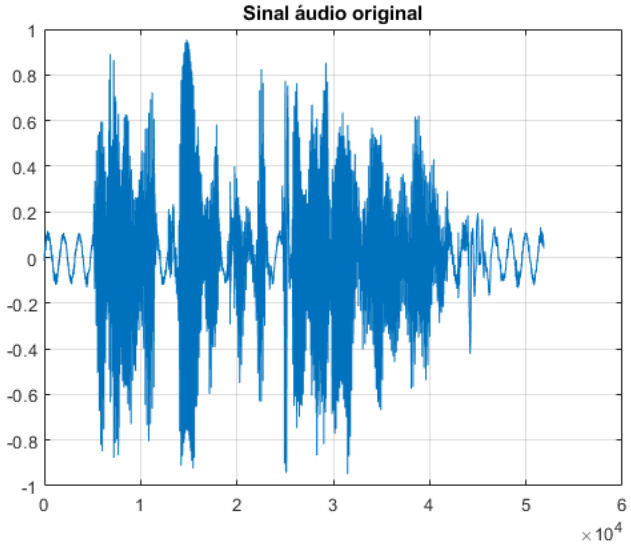


Figure 1: Sinal de áudio

Por inspecção visual do sinal, reparamos na existência duma componente de baixa frequência, de 10 Hz, no sinal. Como próximo passo, vamos filtrar este sinal com um filtro $H(z)$ de primeira ordem, passa-alto, $f_c = 250Hz$ (frequência de corte), com ganho unitário em $\omega = \pi$.

$$H(z) = g\frac{1 - z^{-1}}{1 - az^{-1}}$$

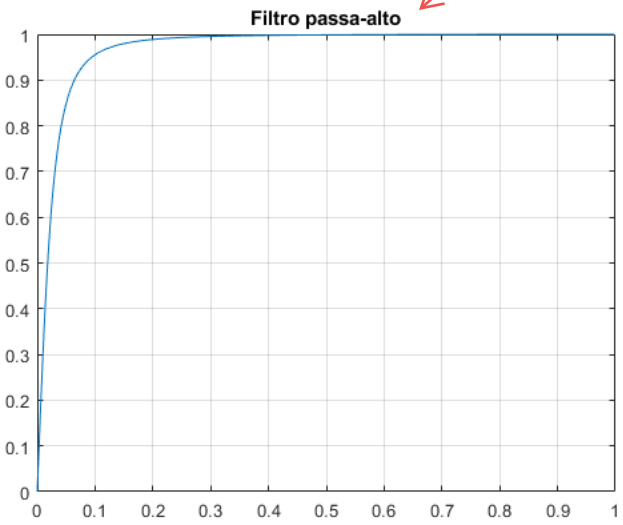
onde $a = e^{-2\pi\frac{f_c}{f_s}}$ e $g = \frac{1+a}{2}$.

O filtro H(z) tem um ganho em DC ($\omega = 0$) nulo e o valor -1 em $\omega = \pi$.

$$H(e^{jw}) = g\frac{1 - e^{jw}}{1 - ae^{jw}} \Leftrightarrow H(e^{j0}) = g\frac{1-1}{1-a} = 0$$
$$\Leftrightarrow H(e^{j\pi}) = g\frac{1+1}{1+a} = g\frac{2}{1+a} = -1$$

+1

(módulo da resposta em frequência)



freq. normalizada (onde pi radianos corresponde a 1.
Se for complicado mexer na figura, devem indicar isso na "caption" da figura.

Figure 2: Filtro Passa-alto H(z)

Após a filtragem do sinal com H(z), usando o comando filter do Matlab, obtemos o sinal descrito em Sinal filtrado.

É possível observar que a componente de 10Hz do sinal áudio foi atenuada e pode ser desprezada mas não foi totalmente eliminada.

Seria bom colocar uma figura semelhante à Fig 1. se possível (como é o caso) ver a diferença.

3 Filtro forma directa II

Vamos agora filtrar o sinal com a forma directa II de H(z), descrito pelo diagrama Diagrama do filtro H(z) em forma directa II. Este diagrama pode ser descrito por duas equações:

O que é? Onde está?

$$w[n] = x[n] + a.w[n-1]$$
$$y[n] = g.(w[n] - w[n-1])$$

Este filtro é implementado pelo seguinte bloco de código em Matlab:

```
y2 = zeros(size(x));
w1 = 0;  %%
for n = 1:Nx
    w = x(n) + a * w1;
    y2(n) = g * (w - w1);
    w1 = w;
end
```

Para obter cada amostra na saída, isto é, para cada valor de y2, são necessárias duas operações de multiplicação e duas de adição. Típicamente uma operação de multiplicação demora mais ciclos de relógio do processador comparativamente a operações de adição, com um custo computacional típico de um ciclo de relógio. Assim, estimamos de grosso modo que a cada oito ciclos de relógio obtemos um valor de y2. Observando que y2 tem 51840 valores, estimamos que foram necessários cerca de 414000 ciclos de relógio para a filtragem deste sinal, um custo computacional não desprezável.

Não será bem verdade (a questão da multiplicação mais rápida que a adição) mas o nº de ciclos depende do processador. Num DSP não é assim. Num caso como este, deverão indicar uma referência que suporte a afirmação (não deve ser uma mera opinião pessoal).

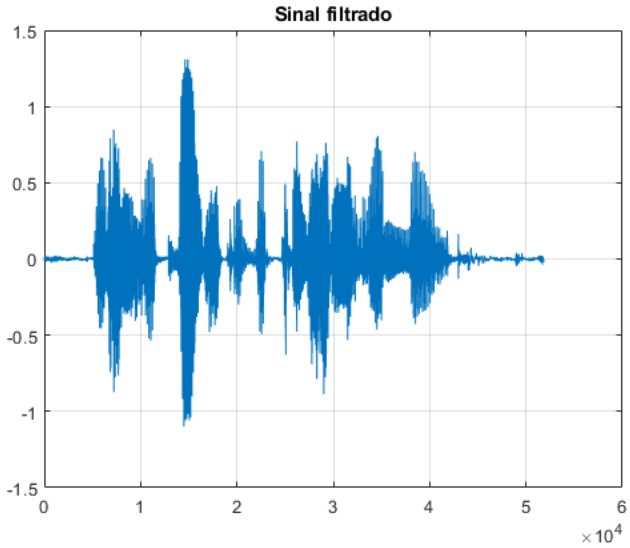
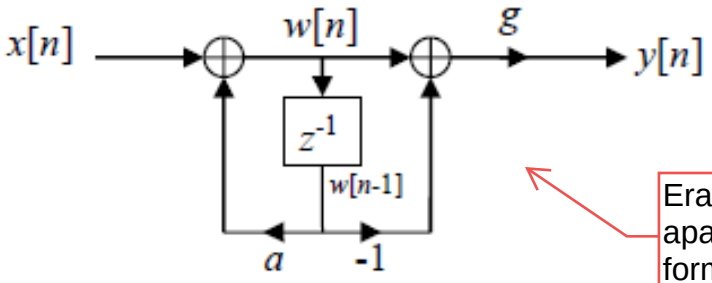


Figure 3: Sinal filtrado



Era mais claro se esta figura aparecesse quando indicam a forma direta II.

Figure 4: Diagrama do filtro H(z) em forma directa II

Vamos agora fazer uma comparação do resultado final dos dois filtros, demonstrada na figura Comparação dos dois filtros implementados. De notar a ordem dos valores do eixo da cota deste gráfico: $1.5e^{-15}$. Como os cálculos estão a ser efectuados ~~num processador~~, existe um desvio devido às operações em vírgula flutuante. O Matlab descreve esta exactidão como $2.2204e^{-16}$. Portanto, os desvios entre o resultado dos dois filtros podem ser atribuídos à realidade das operações em vírgula flutuante efectuadas.

com precisão fixa

Ok. Não está mal, embora se possa aprofundar a questão. O problema é a representação dos números ("double precision" na norma IEEE 754) com 52 bits de mantissa.
A multiplicação de duas mantissas resulta em 52+52=104 bits, mas tem de caber em 52 bits, por arredondamento.
Ver "doc eps" e no fim "Floating-Point Numbers".

4 DFT

sim, os erros de arredondamento.

Vamos agora gerar um sinal sinusoidal simples com frequência de 1 KH descrito, em tempo contínuo como z(t):

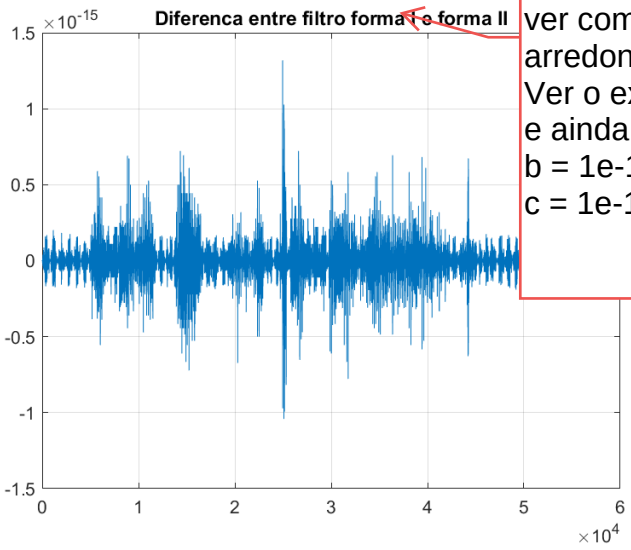
$$z(t) = 0.1sin(2\pi f_o t)$$

Em tempo discreto teremos:

$$z[n] = z(nT) = z(\frac{n}{f_s}$$

)

Onde f_s é a frequência de amostragem do sinal, T o período do sinal indexando cada um das 51840 valores do sinal áudio.



Não é. É a diferença entre a implementação de filtragem intrínseca do Matlab e da atual com o sistema na forma direta II.
O Matlab usa forma direta II transposta, mas não é essa a questão: tem a ver com a precisão finita e erros de arredondamento.
Ver o exemplo1: e = 1 - 3*(4/3 - 1)
e ainda (ordem das operações):
b = 1e-16 + 1 - 1e-16
c = 1e-16 - 1e-16 + 1

Figure 5: Comparação dos dois filtros implementados

A próxima operação a fazer é a soma dos dois sinais: o sinal disponibilizado no Inforestudante 'pcmtest_10Hz.wav' e este sinal por nós gerado. Por audição dos sinais, observamos que o sinal resultante é a sobreposição dos dois sinais.

Propomo-nos agora a calcular a DFT (*Discrete Fourier Transform*) das primeiras 500 amostras deste sinal resultante. Como sabemos que a frequência de amostragem $f_s = 16KHz$, não ~~nós~~ é interessante estudar o espectro deste sinal a frequências superiores a 8KHz, devido ao teorema da amostragem:

> Nyquist-Shannon Sampling Theorem: If a function $x(t)$ contains no frequencies higher than B hertz, it is completely determined by giving its ordinates at a series of points spaced $\frac{1}{2B}$ seconds apart.

nos? retirar?
não é possível!

kHz
(kapa é minúsculo)

Aspas (inglês) e referência:
De onde vem a citação? Wikipédia?

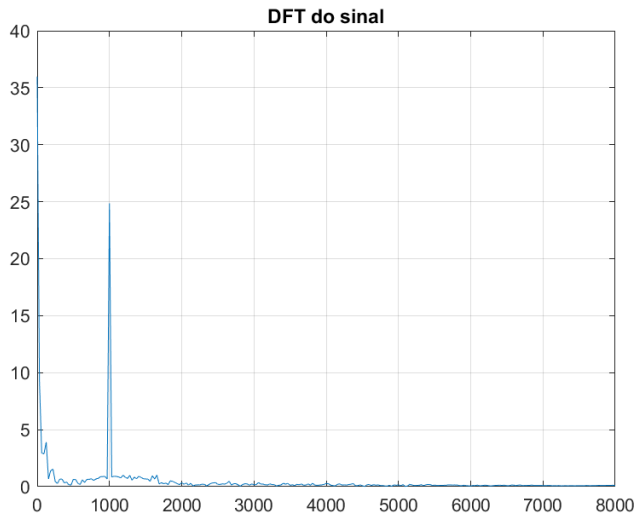


Figure 6: DFT do sinal resultante da sobreposição de dois sinais

A figura DFT do sinal resultante da sobreposição de dois sinais demonstra a DFT do sinal. De notar que estudámos frequências até 8KHz (metade da frequência de amostragem $f_s = 16KHz$). Observamos a presença de um máximo absoluto à frequência de $1Khz$, devido ao sinal sinusoidal por nós gerado e introduzido.

Agá é em maiúscula

5 Sonograma do sinal

Devido à variação temporal do sinal, o seu conteúdo espectral também varia. Para observarmos o espectro do sinal ao longo do tempo, podemos calcular DFTs a um ritmo constante por forma a obtermos o sinal a três dimensões: tempo, frequência e amplitude. Para observarmos estas três dimensões, recorreremos a um sonograma.

Para este propósito, consideraremos *frames* de 500 amostras e DFTs com $N_{FFT} = 512$ e usaremos o comando fft do Matlab para o cálculo da DFT recorrendo a um algoritmode FFT (*Fast Fourier Transform*). As *frames* deverão avançar a cada 160 amostras ($f_s/100 = 160$). Calculando a DFT de cada *frame*, podemos implementar o sonograma do sinal.

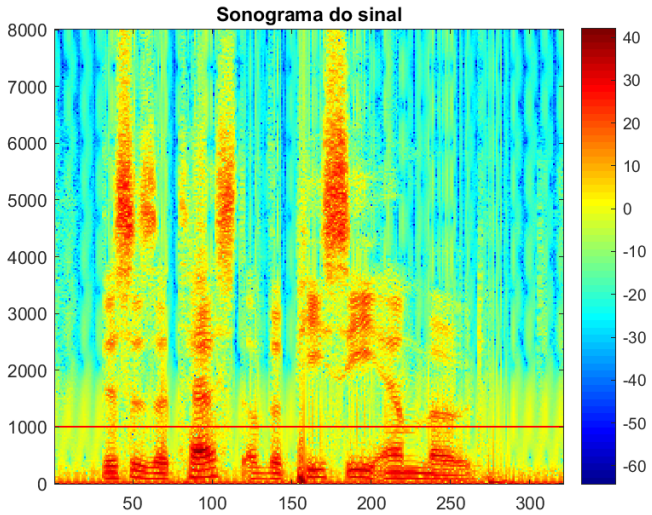


Figure 7: Sonograma de 'pcmtest_10Hz.wav' sobreposto com um tom sinusoidal com frequência de 1KHz

o s

Observando Sonograma de 'pcmtest_10Hz.wav' sobreposto com um tom sinusoidal com frequência de 1KHz, vemos uma linha horizontal vermelha à frequência de 1KHz, confirmando a presença do sinal sinusoidal por nós introduzido. Observamos também os tons associados às partes vozeadas e fricativas do locutor, representadas no sonograma como as manchas vermelhas a baixas frequências e a frequências de aproximadamente 5KHz, respectivamente.

Relatório está bom. Está cuidado de acordo com os requisitos pretendidos. A melhorar apenas a capa (eventualmente) e seguir as observações indicadas.

6 Código Matlab

```
%LAB 1 PDS:  udio , Filtragem e DFT
close all; clc; clear; mkdir('Imagens'); delete Imagens/*.*
%Abrir e ler um ficheiro de  udio
[x,fs]=audioread('pcmtest_10Hz.wav'); % l  sinal e freq. de amostragem.
% x: sinal normalizado (entre -1 e 1) numa coluna (mono) ou em 2 colunas (stereo)
%fs: frequ ncia de amostragem (n  amostras por segundo)
Nc=size(x,2) %n mero de canais/colunas de x: 1=mono, 2=stereo.
Nx=size(x,1) %n mero de amostras/linhas de x (comprimento de x[n])
fig = figure
plot(0:Nx-1,x)%gr fico do sinal (linhas retas entre pontos. Verifique...
%soundsc(x,fs) % reproduz o sinal  udio    frequ ncia de amostragem fs. Se
    variar a
title('Sinal  udio  original')
grid on
% taxa de amostragem o sinal pode ficar demasiado agudo ou demasiado grave.
print(fig, '-dpng', 'Imagens/sinalaudiooriginal');
%Filtra  o de um sinal com filtro passa-alto 1a ordem, fc = 250hz, gain =
%1, w = pi

fc=250; a=exp(-2*pi*fc/fs); g=(1+a)/2; % exp(-wc*n*T)=an (invar. impulsional)
num = g*[1,-1]; den = [1,-a]; % Polin mios numerador e denominador de H(z).
[H,w]=freqz(num,den); % H(exp(jw)) em 512 pontos, w=[0,pi[
fig = figure
plot(w/pi,abs(H)) % Plot do m dulo em freq. normalizada (pi rad corresponde a 1).
title('Filtro passa-alto')
grid on
print(fig, '-dpng', 'Imagens/filtropassalto');
fig = figure
plot(w/pi*fs/2,abs(H)) % Plot em Hz (pois pi rad corresponde a fs/2 Hz).
title('Filtro passa-alto em Hz')
grid on
print(fig, '-dpng', 'Imagens/filtroemHz');
y1=filter(num,den,x);
fig = figure
plot(0:Nx-1,y1)%Verifique que a oscila  o de baixa frequ ncia desapareceu.
title('Sinal filtrado')
grid on
print(fig, '-dpng', 'Imagens/sinalfiltrado');
%Filtro com forma directa II
y2=zeros(size(x)); %espa o para sa da y2[n]; mesmo formato da entrada
w1=0; %condi  o inicial nula.
for n=1:Nx
 w=x(n)+a*w1;
 y2(n)=g*(w-w1);
 w1=w;
end

%vamos comparar y1 com y2
fig = figure;plot(y1-y2), title('Diferenca entre filtro forma I e forma II'), grid
     on
print(fig, '-dpng', 'Imagens/comparacaodosfiltros');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Parte 2: DFT
z = 0.1 * sin(2*pi*1000/fs*(0:Nx-1)');

Nfft = 512;
k = (0:Nfft/2); %k: 257  ndices  de interesse da DFT. Y0(k+1) s o os valores da
    DFT.52
f = k*fs/Nfft; %f: 257 frequ ncias dos   bins   da DFT, desde f=0 at  f=fs/2.
```

```
y = x + z;
Y0=fft(y(1:500),Nfft);
fig = figure
plot(f,abs(Y0(1:257)))
title('DFT do sinal')
grid on
print(fig, '-dpng', 'Imagens/DFTsinal');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Parte 3: Sonograma do sinal
%% Sonograma do sinal com N=500; Nfft=512; M=fs/100
Nfft=512; N=500; M=fs/100;
Nt = floor((Nx-N)/M)+1; %n  de tramas do sinal
Y=zeros(Nfft,Nt); %espa o para o resultado.
i=1; j=N; %i:j s o os  ndices  da 1  trama.
for m=1:Nt
 trama = y(i:j);
 Y(:,m)= fft(trama,Nfft);
 i=i+M;
 j=j+M;
end
fig = figure;
imagesc(1:Nt,f,db(Y(1:257,:))), axis xy, colorbar, title('Sonograma do sinal')
colormap(jet)
print(fig, '-dpng', 'Imagens/sonograma');
```