CIM - Trabalho 1

Requantização e reamostragem de sinais de audio e imagem

CIM 2022/2023 **David Rainho** up201906994
Turma: 1MEEC_T02 **Jorge Pais** up201904841

1 Requantização do Sinal Audio

Para o primeiro problema deste trabalho foi fornecido um ficheiro de audio sem compressão de uma música, no qual foram realizadas diferentes operações de quantização sobre as amostras.
Esta operação foi realizada para os números de bits N = 2, 4, 6, 8, 10, 12, 14, 16, reduzindo o número de dígitos que é utilizado para representar os niveis de representação de cada amostra. Para sinais bipolares, é possível realizar a quantização da seguinte maneira, considerando aproximações por arredondamento:

$$\begin{cases} M = 2^{N-1} \\ x_Q[n] = \text{floor}(0.5 + x[n] * M) \end{cases} \Rightarrow x_{requantizado}[n] = x_Q[n]/M$$

Obtendo todos os sinais de requantizados, o sinal de erro de quantização é obtido através da diferença entre estes e o sinal original:

$$e[n] = x_{requantizado}[n] - x_{original}[n]$$

Assim, é possível então obter um valor para a relação sinal-ruído de quantização através das potências do sinal de erro e do sinal requantizado:

$$P_x = \frac{\sum_{n=0}^{N} x[n]^2}{N} \Rightarrow SNR = \frac{P_{requantizado}}{P_{erro}}$$

Realizando todas estas operações no MATLAB, foi possível obter o seguinte gráfico relacionando o SNR com o número de bits de quantização:

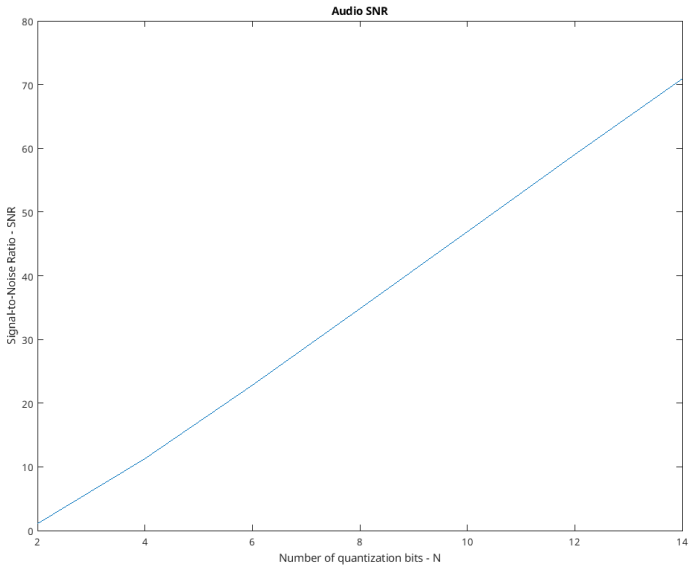


Figura 1: Relação entre o SNR e o número de bits de quantização

Torna-se evidente através da Figura 1, que ao aumentar o número de bits utilizados na requantização a relação sinal-ruído também aumenta. Através da função `polyfit` do MATLAB é possível obter uma função do primeiro grau que melhor aproxima os dados obtidos:

$$SNR = 5.87795N - 11.7575$$

Assim é possível verificar que a cada de bit de quantização N, o SNR aumenta em aproximadamente 6dB. No segundo problema, ao reproduzir cada um dos sinais resultantes, é possível detetar a presença de artefactos até N = 10 bits de quantização a partir desse ponto a diferença é bastante minima. Contudo esta análise é subjetiva, podendo estar afetada por fator externos às operações realizadas, como a qualidade da conversão digital-analógico e dos altifalantes/headphones utilizados durante a reprodução.

2 Requantização do sinal de imagem

De forma semelhante ao sinal áudio, pretende-se realizar a operação de quantização da imagem com N = 2, 3, 4, 5, 6, 7 e 8 bits. Contudo, como a imagem é um sinal unipolar, com representação normalizada $[0; 1[$, deve-se considerar $M = 2^N$.
Recorrendo ao código de MATLAB fornecido, é possível obter os valores SNR (dB) para cada número N de bits e a respetiva reta que melhor se ajusta aos pontos.

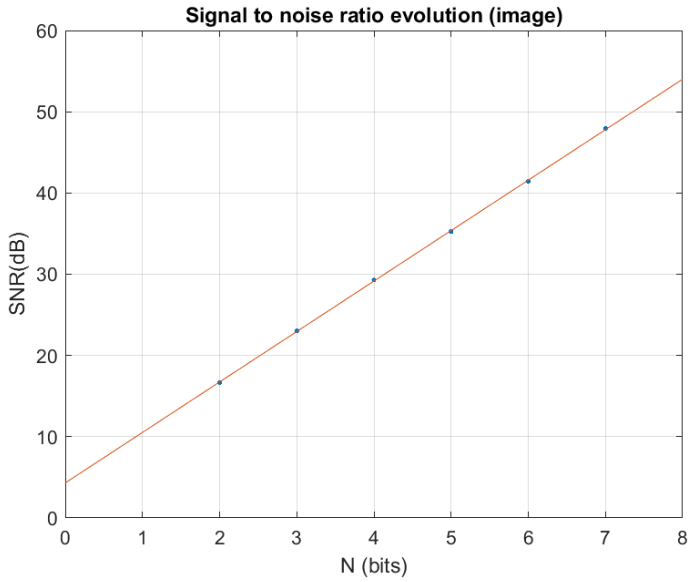


Figura 2: Relação SNR e número de bits N.

A reta obtida, tem declive igual a 6.2117. O valor é próximo do teórico de 6.02.
Após verificação visual da imagem, sujeita a quantização de diferentes valores de N bits, pessoalmente esta não apresenta artefactos notórios para N superior ou igual a 5 bits. Considerando 4 bits, são notórios segmentos grandes em zonas que têm transições mais suaves da cor, por exemplo no braço.

3 Quantização de um sinal sinusoidal

O sinal sinusoidal é considerado um sinal bipolar, logo o processo de quantização é semelhante ao realizado para o áudio. Contudo, teve-se a atenção de garantir que os valores máximos (iguais a 1) ficam associados ao nível superior e assim garante-se o intervalo normalizado $[-1, 1[$.

3.1 i

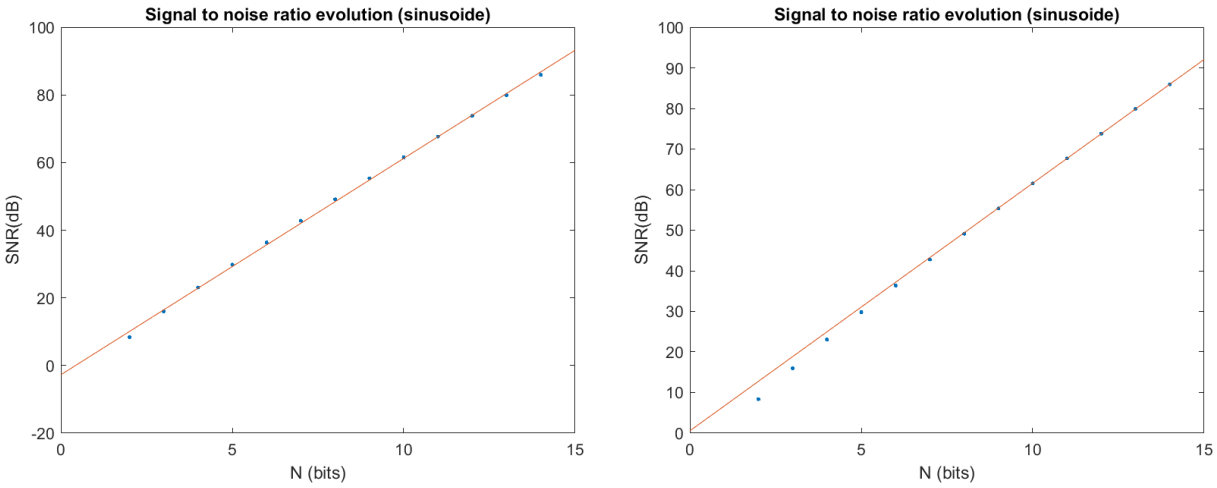
Repetindo o processo já descrito na questão 1, e considerando todos os N=2,3,...,14 bits para a reta que melhor se ajusta, obteve-se a seguinte expressão

$$SNR = 6.3965N - 2.7308,$$

vísivel na figura 3a.
Se forem usados os pontos calculados para N=10,11,...,14 bits, obtem-se a seguinte expressão

$$SNR = 6.0991N + 0.5783,$$

vísivel na figura 3b.



(a) (b)

Figura 3: Aproximação de primeira ordem aos pontos com os pontos (a) N=2,3,...,14 bits e (b) N=10,11,...14 bits.

Consoante se considera o conjunto de bits superiores ($10 \leq N \leq 14$), mais se aproxima a reta da relação teórica esperada (SNR = 6,02N+1,78). Isto acontece, pois o erro tende a apresentar uma distribuição uniforme, caso se assuma na dedução teórica. Este ficará mais claro nos seguintes pontos.

3.2 ii

A correlação é considerada pouco significativa, quando tem módulo inferior a 0.1. Verifica-se essa condição para N=8 bits (ou superior), sendo a amplitude máxima do erro aproximadamente 0.004.

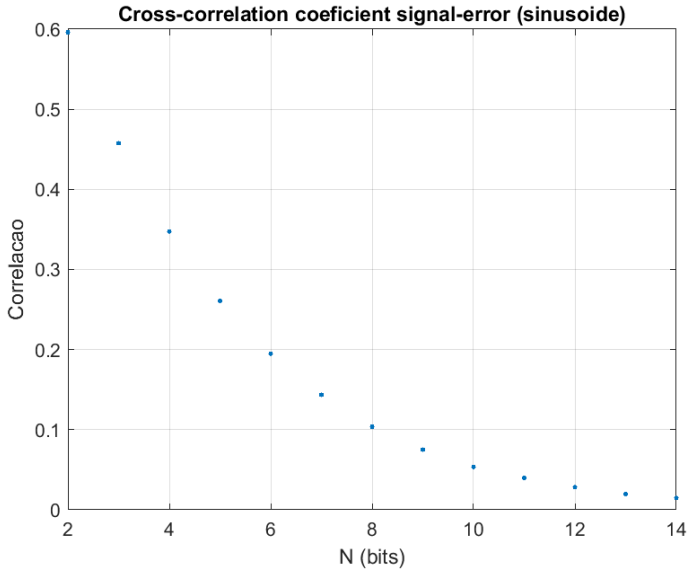
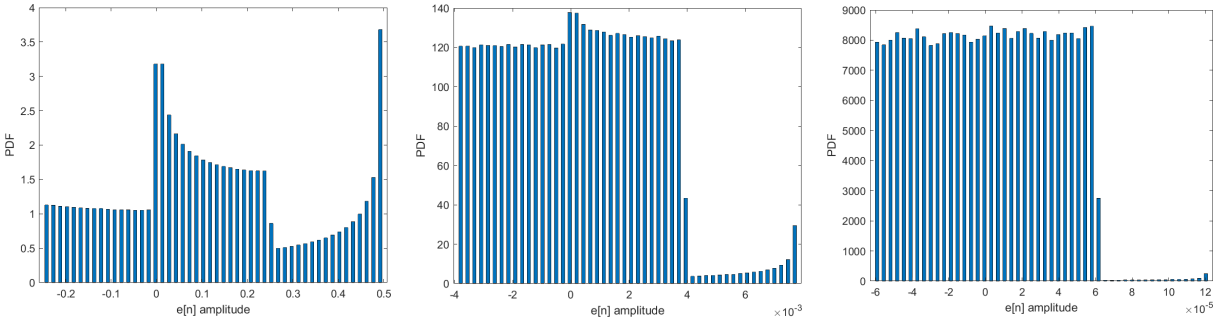


Figura 4: Correlação Cruzada em função do número de N bits.

3.3 iii

Dos gráficos na figura 5, visualiza-se a evolução da PDF do erro, conforme se aumenta o número de bits N.



(a) (b) (c)

Figura 5: Evolução da PDF do erro para número de bits (a)N=2, (b)N=8 e (c)N=14.

No gráfico 5a, a PDF do erro apresenta um formato que indica existir alguma informação do sinal sinusoidal contida no erro, tal como concluído na alínea ii).

Avaliar a distribuição do erro, nesta situação (N=2), é mais simples se pensarmos no sinal sinusoidal, dividido em 4 secções (em amplitude).

- Erros positivos de valor aproximadamente igual a 0.5, ocorrem para “amplitude"próxima do ponto de inflexão da arcada positiva (=1), onde a sinusoide é “mais lenta";

- Na arcada negativa, também com transição “lenta", o ponto de inflexão encontra-se muito próximo de -1 (nível mais baixo), logo espera-se uma grande concentração de erros de valor próximo de 0. Além destes, existem os restantes erros aproximadamente nulos, junto aos níveis -0.5, 0 e 0.5.

Esta simples análise heurística da PDF permite compreender o porquê de o erro apresentar a correlação elevada para N reduzido. Tentando repetir a análise para o gráfico 5b, observa-se que algumas das características não são tão evidentes, sendo mesmo impossível de o fazer para o gráfico 5c. Assim, justifica-se a tendência da correlação obtida em 4, diminuir com o aumento do número de bits N.

Por fim, relembre-se na alínea i), que a relação SNR(N) tende para a reta teoricamente calculada nas aulas, pressupondo que a PDF do erro é uniforme, que é verdade se considerarmos o número de bits N elevado (superior a 10).

4 Reamostragem

Finalmente, o ultimo problema deste trabalho consistia nas reamostragens dos sinais audio e de imagem, através das operações de downsample (decimação), upsample e repetição de amostras.

4.1 Sinal Audio

Em primeiro lugar foram realizadas estas operações sobre o sinal audio. A operação de decimação foi realizada para fatores de F = 2 e 4, eliminando uma em cada duas amostras e 3 em cada 4 amostras, respetivamente. Ouvindo os sinais audio resultantes é possível verificar que algum do conteúdo espectral de alta frequência dos sinais foi perdido. Isto é de esperar dado que a operação de decimação será equivalente a amostrar o sinal original a uma frequência menor.

Relativamente à sobreamostragem, é possível verificar que ao adicionar zeros entre as amostras originais que o sinal não parece perder alguma qualidade durante a reprodução, porém existe uma perda de volume relativamente ao original devido à perda de potência devido ao zero-padding entre as amostras. Isto será de esperar dado que a sobreamostragem terá o mesmo efeito que aumentar a frequência a que o sinal original foi amostrado, e como não existe conteúdo espectral acima da frequência máxima, o sinal irá soar igual.

Quanto à repetição de amostras, verifica-se praticamente o mesmo do que durante a sobreamostragem mas com uma potência de sinal resultante maior resultando em mais volume. Do ponto de vista da amostragem, o mesmo se aplica, sendo que apenas foi utilizado um método de interpolação em amostras diferente.

4.2 Sinal de Imagem

De forma similar ao audio, as mesmas operações foram aplicadas à imagem. Primeiramente a decimação foi realizada que no caso da imagem, ao eliminar alguns dos pixeis fez com que a sofresse uma redução da resolução para 256x256 (F=2) e 128x128(F=4).

Figura 7: Imagem após decimação, F=2 à esquerda e F=4 à direita

Relativamente à sobreamostragem, foi possivel verificar que no caso de F = 2 que a imagem perdeu alguma luminosidade, dado que para cada pixel, foram introduzidos 3 pixeis de valor zero em torno desse. Ao realizar isto para F = 4, a imagem ficou completamente escura, dado que em cada pixel existem 15 pixeis vizinhos completamente escuros.

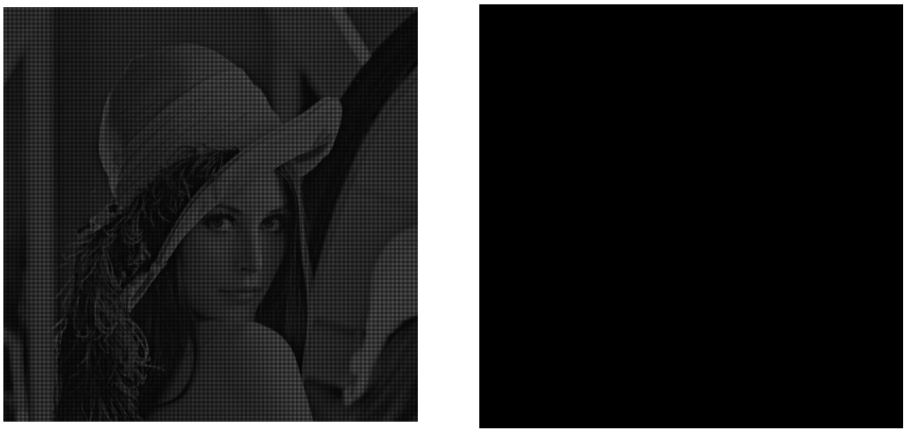


Figura 8: Imagem após sobreamostragem, F=2 à esquerda e F=4 à direita

Através do sinal sobreamostrado, as amostras foram repetidas e obtiveram-se as seguintes imagens. Para F=2 a imagem resultante ficou com resolução 1024x1024 e para F=4 a imagem fica com resolução 2048x2048.

Figura 9: Imagem após repetição de amostras, F=2 à esquerda e F=4 à direita

A Código MATLAB

testesqz.m

```
1
2  % M.EEC045 - CODIFICACAO DE INFORMACAO MULTIMEDIA
3  %
4  % first home assignment (audio part)
5  %
6  % due date: February 24, 2023
7  %
8  % Anibal Ferreira
9
10 % AUDIO PART
11
12 [x,FS]=audioread('sting22.wav');
13 % [x,FS,NBITS]=wavread('sting22.wav'); % old Matlab versions
14 NBITS=16;
15 %sound(x,FS,NBITS); % NOTA: x values are already in the range [-1, 1]
16 sound(x(1:1*FS), FS, NBITS); % Only play the song for 2 seconds
17 samples=[0:length(x)-1];
18 figure(1)
19 plot(samples/FS, x);
20 xlabel('Time (s)');
21 ylabel('Amplitude');
22 title('sting22.wav');
23
24 N = 2:2:NBITS;  % Values for requantization
25 M = 2.^(N-1);   % Number of quantization levels (bipolar)
26
27 for i = 1:length(N) % Requantize
28     fprintf("Number of bits: %d \n", N(i));
29     x_Q = floor(0.5+x.*M(i)); % Round-up approximation
30     x_R = x_Q./M(i);
31
32     error = x_R-x;
33     Px = sum(x_R.^2)/length(x_R);
34     Perror = sum(error.^2)/length(error);
35     SNR(i) = 10*log10(Px/Perror);
36     fprintf("SNR = %d dB\n", SNR(i));
37
38     sound(x_R, FS, NBITS); % listen to each result!
39     pause
40     %sound(error, FS, NBITS);
41     %pause
42 end
43
44 figure(2)   % plot the SNR
45 fprintf("Model parameters: \n");
46 fprintf("%d\n", polyfit(N(1:(length(N)-1)), SNR(1:(length(N)-1)), 1));
47 plot(N, SNR);
48 title("Audio SNR");
49 xlabel("Number of quantization bits - N");
50 ylabel("Signal-to-Noise Ratio - SNR");
51 pause
52 % IMAGE PART
53
54 % reads and displays image
55 A=imread('lena512.bmp');
56 figure(2)
57 imshow(A,[0 255]); % displays original image
58 A=single(A)/255.0; % convert to float and normalizes [0, 1.0]
59 title('lena512.bmp');
60
61 % insert code here
62 Ar=A; % this is temporary, to be replaced by the new code
63
64 N = 2:2:NBITS;  % Values for requantization
65 M = 2.^N;   % Number of quantization levels (unipolar)
66
67 for i = 1:length(N)
68     fprintf("Number of bits: %d \n", N(i));
69     A_Q = floor(0.5+A.*M(i));
70     A_R = A_Q./M(i);
71
72     error = A_R - A;
73     Px = sum(A_R.^2)/length(error);
74     Perror = sum(error.^2)/length(error);
```

```
    SNR(i) = 10*log10(Px/Perror);
    fprintf("SNR = %d dB\n", SNR(i));

    Ar=uint8(A_R*255.0); % convert to "uint" format
    figure(3)
    imshow(Ar,[0 255]); % displays modified image
    title('modified Lena, N');

    pause
end

figure(4)   % plot the SNR
fprintf("Model parameters: \n");
fprintf("%d", polyfit(N(1:(length(N)-1)), SNR(1:(length(N)-1)), 1));
plot(N, SNR);
title("Image SNR");
xlabel("Number of quantization bits - N");
ylabel("Signal-to-Noise Ratio - SNR");
```

sineProcessing.m

```
samples = 0:(1E5 - 1);
x = sin(0.22*samples);

N = 2:2:14;     % Number of quatization bits
M = 2.^(N-1);   % number of quatization levels (bipolar)

for i = 1:length(N)
    fprintf("Number of bits: %d\n", N(i));

    x_Q = floor(0.5+ x.*M(i));

    index = find(x_Q==M(i)); % E necessario retirar o ultimo nivel de quantizacao
    B = maxk(unique(x_Q),2);
    x_Q(index) = B(2);

    x_R = x_Q./M(i);

    error = x - x_R;

    Px = sum(x_R.^2)/length(x_R);
    Perror = sum(error.^2)/length(error);
    SNR(i) = 10*log10(Px/Perror);
    fprintf("SNR = %d dB\n", SNR(i));

    corr_matrix = corrcoef(x, error);
    CORR(i) = corr_matrix(1,2);

    E(i,:) = error;
end

figure(1)   % plot the SNR
fprintf("Model parameters: \n");
fprintf("%d \n", polyfit(N(1:(length(N)-1)), SNR(1:(length(N)-1)), 1));
plot(N, SNR);
title("SNR vs Number of bits");

figure(2);
plot(N, CORR);
title("Error/Original signal correlation")

figure(3);
[H X] = hist(x, 50); equalize = 50/(max(x) - min(x));
bar(X, H/sum(H)*equalize, 0.5);
ylabel('FDP'); xlabel('x[n] amplitude');

pause
for i = 1:length(N)
    figure(4);
    [H X] = hist(E(i,:), 50); equalize = 50/(max(E(i,:)) - min(E(i,:)));
    bar(X, H/sum(H)*equalize, 0.5);
    ylabel('FDP'); xlabel('error amplitude');
    pause
end
```

resampling_audio.m

```
% Audio part
[x, Fs] = audioread("sting22.wav");
```

```

%sound(x, Fs); pause

for f = 2:2:4
    fprintf("F = %d\n", f);
    x_down  = downsample(x, f); % Decimate signal, downsample
    x_up    = upsample(x, f);   % Upsample the signal, this can be used also in
    conjunction with the filter to get
    x_repeat= filter(ones(1,f), 1, x_up); % all ones impulse response, since the x_up
    samples are spaced out by zeros

    % Now the can listen to the signals (note the playback frequency!!!)
    fprintf("Playing the downsampled signal now!\n"); sound(x_down, Fs/f); pause; clear
    sound;
    fprintf("Playing the upsampled signal now!\n"); sound(x_up, Fs*f); pause; clear sound
    ;
    sound(x, Fs); pause; clear sound;
    fprintf("Playing the repeated samples signal now!\n"); sound(x_repeat, Fs*f); pause;
    clear sound;
end
```

resampling_images.m

```
A = imread("lena512.bmp");
figure(1); imshow(A);

for f = 2:2:4
    fprintf("F = %d\n", f);
    x_down = downsample(A, f); x_down = downsample(x_down.', f).'; % ALSO transpose
    figure(2); imshow(x_down);

    x_up = upsample(A, f); x_up = upsample(x_up.', f).';
    figure(3); imshow(x_up, [0 255]);

    x_repeat = filter2(ones(f, f), x_up);
    figure(4); imshow(x_repeat, [0 255]);
    pause;
end
```