Léna PETRUCCIOLI

61659

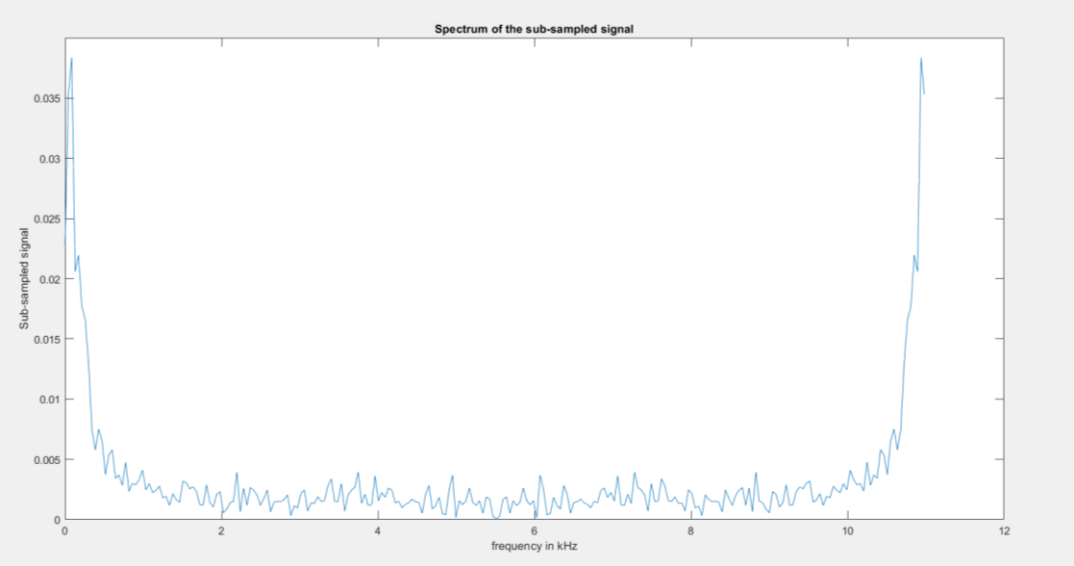
Homework 2

Problem 1

1) To find the maximum frequency in the original signal, I used the fft curb and zoomed on it to find the maximum frequency value that gives us a peak in amplitude, this frequency being below 20 kHz (because the sound is a music and so the maximum frequency can be heard by a human ear). I found the maximum frequency giving an important peak is 3.27 kHz, so it seems to be the maximum frequency of the original signal.

2)

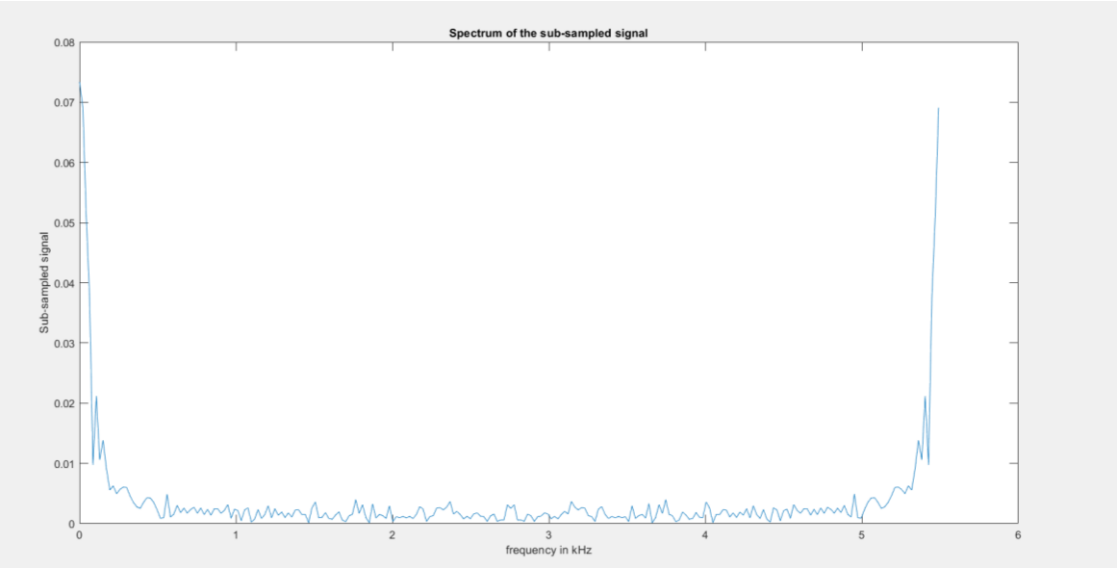
- With k = 4 we divide the sampling frequency by 4 (so we multiply the sampling period by 4). We have $fe_4 = \frac{fe}{4} = \frac{44.1}{4} = 11.025\ kHz$ and the spectrum (in frequencies) of the sub-sampled signal is the following:



We see that there is no aliasing problem since the part of the spectrum around 0 kHz and the one around 11.025 kHz are distinct / separated ones. (It is because 11.025 kHz ≥ 2fmax, fmax being the maximum frequency of the original signal).

When we hear the sound, it still seems correct.

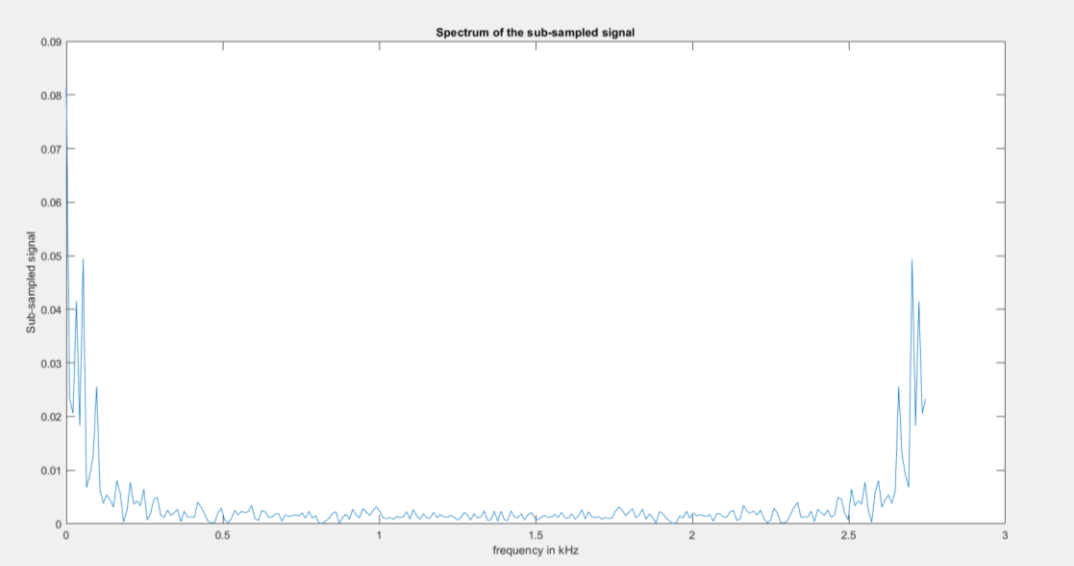
- Now if we take k = 8 we divide the sampling frequency by 8 (so we multiply the sampling period by 8). We have $fe_8 = \frac{fe}{8} = \frac{44.1}{8} \approx 5.513$ kHz and the spectrum (in frequencies) of the sub-sampled signal is the following:



We see that the peaks are not as well defined as in the previous case. 5.513kHz is now not big enough compared to 2fmax, fmax being the maximum frequency of the original signal.

When we hear the sound, it seems correct except at some precise moments where some of the frequencies seem to have been changed.

- Now if we take k = 16 we divide the sampling frequency by 16 (so we multiply the sampling period by 16). We have $fe_{16} = \frac{fe}{16} = \frac{44.1}{16} \approx 2.756$ kHz and the spectrum (in frequencies) of the sub-sampled signal is the following:



We see that there is an aliasing problem, the energy is transmitted in several peaks with low amplitude and not in two single peaks with great amplitude anymore.

The part of the spectrum around 0 kHz and the one around 2.756 kHz are not separated enough (it is because 2.756 kHz < 2fmax, fmax being the maximum frequency of the original signal): they overlap, and we can't find the spectrum of the original signal anymore and so we cannot reconstitute the original signal. When we hear the sound, it is not so good now, with unexpected frequencies.

Conclusion: The aliasing problem happens with a sampling frequency low compared to 2fmax and therefore a sampling period too large.

3) - With 2^4 values used for quantization: there are just a few values to represent the different amplitudes of the original signal and it seems not to be enough. When we hear the sound, we hear it is sliced with notable distinctions between the amplitudes.

- With 2^8 values: the sound we hear is better and there seems to be enough values to represent the different amplitudes of the original signal.

Conclusion: if the quantization level is too low (not enough values) the quantized signal takes less amplitude values than needed to have a signal that sounds like the originl one. Increasing the quantization level enables to get a signal with more precise amplitude values.

4) If we take the original values in the y vector, we have 679578 values that contain 5 digits each. So if we compute the length of this original signal we have 679578 x 5 = 3397890 which is quite big.

Then to run the code I kept a sampling frequence $\frac{fe}{4}$ and a quantization with a level 2^8.

For the uniform coding I get: lstream_total = 1359160 and for the Huffmann one: lHuffman = 971613 which is smaller.

Conclusion: The uniform code provides a longer streamlength than the Huffman code (which takes the occurrences into account to reduce the number of bits used). The Huffman code is then better to compress the information we have. Both compression methods enable us to use less information as we needed to code the original signal (y: decimal numbers with 5 digits each).

Problem 2

1) To demodulate the signal, we need to remove the oscillations introduced when we put the signal on a carrier frequency.

The wave we have before demodulation is under this form:

$\sqrt{SNR}[Ak\cos(2\pi f_0\, t) - Bk\, sin(2\pi f_0 t)] + noise$ where Ak is the inphase component and Bk the quadrature component. Ak is in fact $Acos(\varphi)$ and Bk is in fact $Asin(\varphi)$ where A is the amplitude and φ the phase both given by the 16QAM constellation map. So A and φ directly enable us to find back the binary symbols transmitted.

- ➔ We just have to find A and φ through and Ak and Bk. To find back Ak we multiply the whole signal by $2\cos(2\pi f_0\, t)$ and to find back Bk we multiply it by $-2\sin(2\pi f_0 t)$. Since

we have energy transmitted around 0 MHz and around $2f_0$ so 1600 MHz, we need to finally remove oscillations around $2f_0$ with a low-pass filter.
We apply the low-pass filter with a frequency $\frac{1}{\text{duration of a symbol}}$ and then we can find back Ak and Bk, then A and φ.

2) We have a signal put on a carrier frequency $f_0 = 800$ MHz. The processing starts with the inphase component: it is shown that the spectrum we obtain after the first step has energy around 0 GHz and around 1.6 GHz (which corresponds to $2f_0$).

Then the low-pass filter enables to remove the oscillations around 1.6 GHz and so the signal in time domain is clearer: we have the principal oscillations (the ones for the different binary symbols) and a little bit noise, but we can now read the values for Ak.

For the quadrature, the same two steps are applied, and we are able to read the values for Bk on the time representation of the signal.

3) The symbol duration is 2 μs since it is the time during which the value of both Ak and Bk are constant. Ak and Bk change every 2 μs, A and φ can be found for each binary symbol.

The couple values of inphase and quadrature & the corresponding binary symbols from the constellation are the following:

(-1, 3) → 0100

(3, 3) → 1000

(-1, -3) → 0110

(-1, 1) → 0101

(-1, -3) → 0110

(1, 3) → 1100

(-1, -3) → 0110

(1, 3) → 1100

(-1, -3) → 0110

(1, -1) → 1111

(-3, 3) → 0000

When we use the ASCII converter after having grouped the bits by 8, we find the name “Hello” for the song. This one is sung by Adele.

Problem 3

2) We have to find the frequency between two subcarriers to have orthogonal subcarriers so to have $\Delta f = \frac{1}{T}$ between two neighbour subcarriers, with T being the duration of an OFDM symbol transmitted. We have a number of subcarriers equal to N = 128.

Since the T duration is $T = \frac{N}{Fs}$ and since the frequency between two neighbour subcarriers must be $\frac{1}{T}$, we have:

$\Delta f = \frac{1}{T} = \frac{Fs}{N} = \frac{1.92*10^6}{128} = 15000$ Hz = 15 kHz.

3) The data rate takes into account the number of information bits transmitted during one symbol and the number of subcarriers dedicated to transmitting the message so:

We have 4 bits transmitted in a symbol (because we use 16QAM) but only 3 out of the four 4 bits are information bits, the last one is considered a parity bit. The subcarriers dedicated to transmitting the message are the 1 to 24 ones, so 24 symbols can be transmitted at the same time: the useful data rate is then (24 x 4 x (3/4)) / T = 24 x 3 x 15000 = 1 080 000 b/s = 1.08 Mb/s.

4) We read the inphase and quadrature components amplitudes on the two spectrums at the end. Here are the couples found for inphase and quadrature components and the corresponding binary symbols:

(-1, 1) : 0101;	(-1, -3) : 0110;	(-3, 3) : 0000;	
(1, 3) : 1100;	(-1, 1) : 0101;	(3, -3) : 1010;	(1, 3) : 1100;
(3, 1) : 1001.	(-3, -1) : 0011;	(-3, 3) : 0000;	(-3, 3) : 0000;
(1, 3) : 1100;	(3, 1) : 1001;	(3, -3) : 1010;	(1, 3) : 1100;
(3, 1) : 1001;	(-3, -1) : 0011;	(-3, 3) : 0000;	(-3, 3) : 0000;
(1, 3) : 1100;	(-3, -1) : 0011;	(-3, -1) : 0011;	(3, 1) : 1001;
(3, -3) : 1010			

5) We remove the redundancy bits so the last one for each group of four bits. We have the following binary stream:
010011000110010101110100001000000110100101110100001000000110001001100101

If we group the bits by 8 and we use the online ascii converter we get: “Let it be”

This song is sung by the Beatles.